

OBJECT ORIENTED ANALYSIS AND DESIGN – CS6502

**V.S.B ENGINEERING COLLEGE, KARUR
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

Academic Year- 2018 - 2019

TWO MARK QUESTIONS WITH ANSWERS

SUB NAME: OBJECT ORIENTED ANALYSIS AND DESIGN

SUB CODE: CS6502

SEMESTER / YEAR / DEPT: V / III / CSE

UNIT-I**Introduction to OOAD****1. What is the critical ability of an object oriented system?**

A critical ability of Object Oriented development is to skilfully assign responsibilities to software objects. It is one activity that must be performed either while drawing a UML diagram or programming and it strongly influences the robustness, maintainability, and reusability of software components.

2. What is meant by object oriented system development methodology?

Object oriented system development methodology is a way to develop software by building self-contained modules or objects that can be easily replaced, modified and reused.

3. What is meant by analysis and design?

Analysis emphasizes an investigation of the problem and requirements, rather than a solution. For example, if a new online trading system is desired, Analysis answers the following questions: How will it be used? & what are its functions?

"Analysis" is a broad term, and it is referred as requirements analysis (an investigation of the requirements) or object-oriented analysis (an investigation of the domain objects).

Design emphasizes a conceptual solution (in software and hardware) that fulfils the requirements, rather than its implementation. For example, a description of a database schema and software objects. Design ideas often exclude low-level or "obvious" details obvious to the intended consumers. Ultimately, designs can be implemented, and the implementation (such as code) expresses the true and complete realized design.

Useful analysis and design have been summarized in the phrase do the right thing (analysis), and do the things right (design).

What is OOAD?**4. What is object oriented analysis and design?**

During object-oriented analysis there is an emphasis on finding and describing the objects or concepts in the problem domain. For example, in the case of the flight information system, some of the concepts include Plane, Flight, and Pilot.

During object-oriented design (or simply, object design) there is an emphasis on defining software objects and how they collaborate to fulfil the requirements. For example, a Plane software object may have a tailNumber attribute and a getFlightHistory method.

5. Define –Domain Model

Domain Model is defined as a visualization of the concepts or mental models of a real-world

domain and it is also called a conceptual object model. Object-oriented analysis is concerned with creating a description of the domain from the perspective of objects. There is an identification of the concepts, attributes, and associations that are considered noteworthy. The result can be expressed in a domain model that shows the noteworthy domain concepts or objects. It can be noted that a domain model is not a description of software objects.

What is UML?

6. What is UML?

(M/J – 12)

The Unified Modeling Language (UML) is a visual language for specifying, constructing and documenting the artifacts of systems.

The word visual in the definition is a key point - the UML is the de facto standard diagramming notation for drawing or presenting pictures (with some text) related to software primarily OO software.

7. What are the three ways to apply UML?

The three ways to apply UML are:

- 1) UML as sketch – Informal and incomplete diagrams (often hand sketched on whiteboards) created to explore difficult parts of the problem or solution space, exploiting the power of visual languages.
- 2) UML as blueprint – It is a relatively detailed design diagrams used either for,
 - (i) Reverse engineering to visualize and better understanding existing code in UML diagrams
 - (ii) Code generation (forward engineering)

If reverse engineering, a UML tool reads the source or binaries and generates (typically) UML package, class, and sequence diagrams. These "blueprints" can help the reader understand the big picture elements, structure, and collaborations.

Before programming, some detailed diagrams can provide guidance for code generation (e.g., in Java), either manually or automatically with a tool. It's common that the diagrams are used for some code, and other code is filled in by a developer while coding (perhaps also applying UML sketching).

- 3) UML as programming language – Complete executable specification of a software system in UML. Executable code will be automatically generated, but is not normally seen or modified by developers; one works only in the UML "programming language." This use of UML requires a practical way to diagram all behaviour or logic (probably using interaction or state diagrams), and is still under development in terms of theory, tool robustness and usability.

8. What are the three perspectives to apply UML?

The three perspectives to apply UML are,

- 1) Conceptual perspective the diagrams are interpreted as describing things in a situation of the real world or domain of interest.
- 2) Specification (software) perspective the diagrams (using the same notation as in the conceptual Perspective) describe software abstractions or components with specifications and interfaces, but no commitment to a particular implementation (for example, not specifically a class in C# or java).

- 3) Implementation (software) perspective the diagrams describe software implementations in a particular technology (such as Java).

9. Define – Conceptual Class, Software Class and Implementation Class

- 1) **Conceptual class** – real world concept or thing. A conceptual or essential perspective. The UP Domain Model contains conceptual classes.
- 2) **Software class** – a class representing a specification or implementation perspective of software component, regardless of the process or method.
- 3) **Implementation class** – a class implemented in a specific OO language such as Java.

What is the Unified process (UP) phases?

10. Define – Unified Process (UP)

The Unified Process is defined as a popular iterative software development process for building object-oriented systems. In particular, the Rational Unified Process or RUP, a detailed refinement of the Unified Process, has been widely adopted.

The UP is very flexible and open, and encourages including skilful practices from other iterative methods, such as from Extreme Programming (XP), Scrum, and so forth. For example, XP's test-driven development, refactoring and continuous integration practices can fit within a UP project. So can Scrum's common project room ("war room") and daily Scrum meeting practice.

The UP combines commonly accepted best practices, such as an iterative lifecycle and risk-driven development, into a cohesive and well-documented process description.

11. What is the importance of the unified process?

The importance of the Unified Process (UP) is,

- 1) The UP is an iterative process. Iterative development influences how to introduce OOAD, and to understand how it is best practiced.
- 2) UP practices provide an example structure for how to do and thus how to explain OOAD.
- 3) The UP is flexible, and can be applied in a lightweight and agile approach that includes practices from other agile methods (such as XP or Scrum).

12. What is iterative and evolutionary development?

A key practice in both the UP and most other modern methods is iterative development. In this lifecycle approach, development is organized into a series of short, fixed-length (for example, three-week) mini-projects called iterations; the outcome of each is a tested, integrated, and executable partial system. Each iteration includes its own requirements analysis, design, implementation, and testing activities.

The iterative lifecycle is based on the successive enlargement and refinement of a system through multiple iterations, with cyclic feedback and adaptation as core drivers to converge upon a suitable system. The system grows incrementally over time, iteration by iteration, and thus this approach is also known as iterative and incremental development. Because feedback and adaptation evolve the

specifications and design, it is also known as iterative and evolutionary development.

13. What are the benefits of iterative development?

The benefits of iterative development are,

- 1) Less project failure, better productivity, and lower defect rates; shown by research into iterative and evolutionary methods
- 2) Early rather than late mitigation of high risks (technical, requirements, objectives, usability, and so forth) early visible progress
- 3) Early feedback, user engagement, and adaptation, leading to a refined system that more closely meets the real needs of the stakeholders
- 4) Managed complexity; the team is not overwhelmed by "analysis paralysis" or very long and complex steps
- 5) The learning within an iteration can be methodically used to improve the development process itself, iteration by iteration

14. What are agile methods?

Agile development methods usually apply time boxed iterative and evolutionary development, employ adaptive planning, promote incremental delivery, and include other values and practices that encourage agility, rapid and flexible response to change.

Agile methods share best practices like evolutionary refinement of plans, requirements, and design. In addition, they promote practices and principles that reflect an agile sensibility of simplicity, lightness, communication, self-organizing teams, and more.

15. List any five agile principles.

The five agile principles are,

- 1) Satisfy the customer through early and continuous delivery of valuable software
- 2) Agile processes harness change for customer's competitive advantage
- 3) Deliver working software frequently
- 4) Agile software promote sustainable development
- 5) The best architecture, requirements, and design emerge from self-organizing teams

16. Define – Agile Modelling

The very act of modeling can and should provide a way to better understand the problem or solution space. The purpose of doing UML is to quickly explore (more quickly than with code) alternatives and the path to a good OO design. Many agile methods, such as feature-driven Development, DSDM, and Scrum include significant modelling sessions. The purpose of modelling is primarily support understanding and communication, not documentation.

17. List the other Critical UP practices.

(2Marks) The idea for UP practice is short time boxed iterative, evolutionary, and adaptive development.

Some additional best practices and key ideas in UP are,

- 1) Tackle high-risk and high-value issue in early iterations
- 2) Continual evaluation, feedback and requirements from users
- 3) Build cohesive, core architecture in early iterations
- 4) Continuously verify quality; test early, often and realistically
- 5) Practice Change Request and Configuration Management

18. List the different types of UP phases.

(2Marks) An UP Project organizes work and iterations across four major phases are,

- 1) Inception – approximate Vision, Business case, scope, vague estimates
- 2) Elaboration – Refined Vision, iterative implementation of the core architecture, resolution of high risks, identification of most requirements and scope, more realistic estimates
- 3) Construction – Iterative implementation of the remaining lower risk and easier elements, and preparation for deployment
- 4) Transition – beta tests, deployment

19. List the types of UP disciplines.

(2Marks)

Some of the artifacts in the following Disciplines are:

- 1) Business Modelling – The Domain Model artifact, to visualize noteworthy concepts in the application domain
- 2) Requirements – The Use Case Model and Supplementary specification artifacts to capture functional and non-functional requirements
- 3) Design – The Design Model artifact, to design the software artifacts
- 4) Implementation – Programming and building the system, not deploying it

20. What does Use case Diagram represent? Give an example.

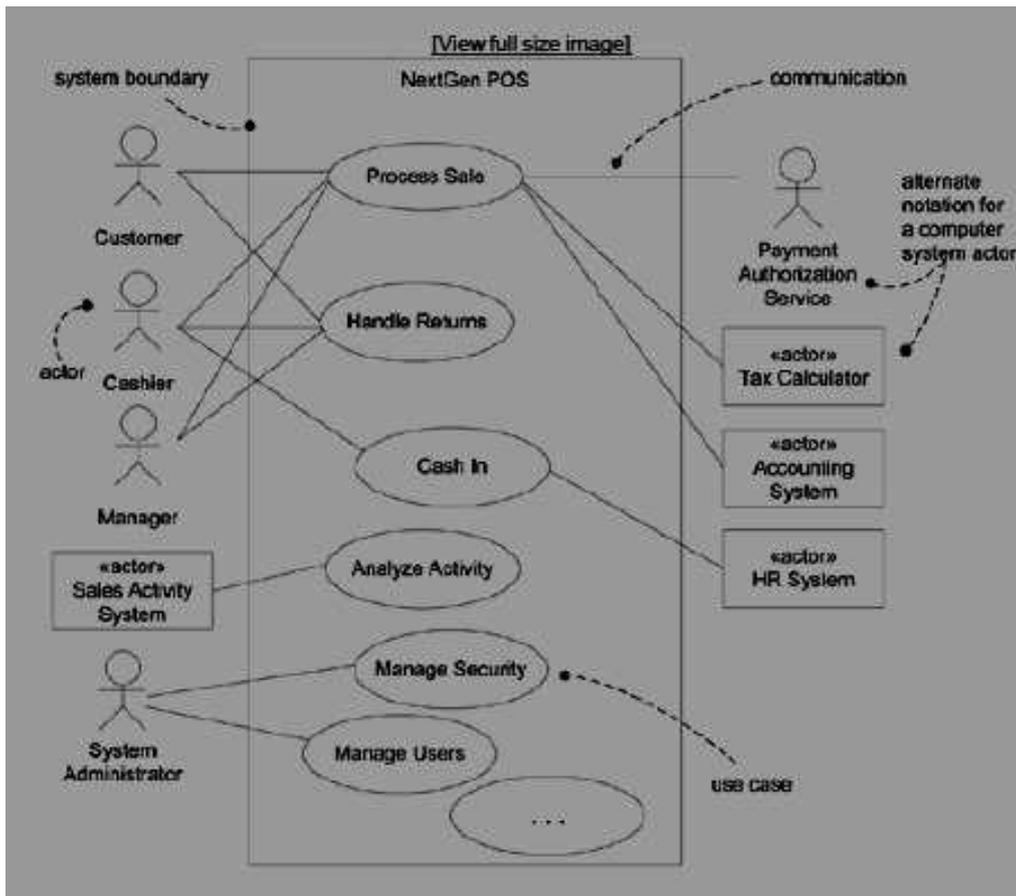
(2Marks)

The UML provides use case diagram notation to illustrate the names of use cases and actors, and the relationships between them.

Guideline:

A simple use case diagram is drawn in conjunction with an actor-goal list.

A use case diagram is an excellent picture of the system context; it makes a good context diagram that is, showing the boundary of a system, what lies outside of it, and how it gets used. It serves as a communication tool that summarizes the behaviour of a system and its actors. A sample partial use case context diagram for the NextGen system is shown below.



UML activity diagrams and modelling

21. What is an UML activity diagram?

(2Marks)

A UML activity diagram shows sequential and parallel activities in a process. They are useful for modeling business processes, workflows, data flows, and complex algorithms.

22. State the role and application of activity diagrams.

(2Marks)

A UML activity diagram offers rich notation to show a sequence of activities, including parallel activities. It may be applied to any perspective or purpose, but is popular for visualizing business workflows and processes, and use cases.

23. Define –Swim Lane

(N/D – 11, N/D – 12) (2Marks)

A swim lane (or swimlane) is defined as a visual element used in process flow diagrams, or flowcharts that visually distinguishes responsibilities for sub-processes of a business process. Swim lanes may be arranged either horizontally or vertically.

24. What is business object model?

(2Marks)

The UP Domain Model is a specialization of the UP Business Object Model (BOM) "focusing on

explaining 'things' and products important to a business domain" [RUP]. That is, a Domain Model focuses on one domain, such as POS related things. The more broad BOM, is an expanded, often very large and difficult to create, multi-domain model that covers the *entire* business and all its sub domains.

25. List the elements in activity modelling.

(2Marks) An activity diagram may have the following elements:

1. Activity states represent the performance of an activity or step within the workflow.
2. Transitions show what activity state follows another. This type of transition can be referred to as a completion transition. It differs from a transition in that it does not require an explicit trigger event; instead it's triggered by the completion of the activity that the activity state represents.
3. Decisions for which a set of guard conditions are defined. Guard conditions control which transition, of a set of alternative transitions, follows once the activity is complete. You may also use the decision icon to show where the threads merge again. Decisions and guard conditions allow you to show alternative threads in the workflow of a business use case.
4. Synchronization bars are used to show parallel sub flows. Synchronization bars allow you to show concurrent threads in the workflow of a business use case.

UML State Diagrams and Modeling

26. What are UML state machine diagrams?

(N/D – 12)(2Marks)

A UML state chart diagram illustrates the interesting events and states of an object, and the behavior of an object in reaction to an event. Transitions are shown as arrows, labeled with their event. States are shown in rounded rectangles.

27. Define – Event, State and Transition

(A/M – 11, M/J – 12) (2Marks)

An event is defined as a significant or noteworthy occurrence. It is a label associated with a transition that identifies the message which causes a state change. Example: A telephone receiver is taken off the hook.

A state is defined as the condition of an object at a moment in time, the time between events. Example: A telephone is in the state of being "idle" after the receiver is placed on the hook and until it is taken off the hook.

A transition is defined as a relationship between two states that indicates that when an event occurs, the object moves from the prior state to the subsequent state. Example: When the event "off hook" occurs, transition the telephone from the "idle" to "active" state.

UML Deployment and Component Diagrams

28. What are deployment diagrams?

(2Marks)

A deployment diagram shows the assignment of concrete software artifacts (such as executable files) to computational nodes (something with processing services). It shows the deployment of software elements to the physical architecture and the communication (usually on a network) between physical elements.

29. What is component diagram?

(M/J – 12)(2Marks)

The component diagram helps to model the physical aspect of an Object-Oriented software system. It illustrates the architectures of the software components and the dependencies between them.

30. Define–Component

(N/D – 11, N/D – 12) (2Marks)

A component is defined as a modular part of a system that encapsulates its contents and whose manifestation is replaceable within its environment. A component defines its behavior in terms of provided and required interfaces.

31. What is meant by an axiom? List the two design axioms of object oriented design.

(2Marks)

An axiom is a fundamental truth that always is observed to be valid and for which there is no counter example or exception. Two design axioms of object oriented design are:

Axiom 1: The independence axiom

Axiom 2: The information axiom

UNIT-II

Designing objects with responsibilities

1. What is an initial domain object?

(2Marks)

An initial domain object is a class and it is chosen to be at or near the root of the containment or aggregation hierarchy of domain objects. This may be a facade controller, such as Register, or some other object considered to contain all or most other objects, such as a Store.

2. List the ways to connect the UI Layer to the Domain Layer.

(2Marks) The ways are:

- An initializing routine (for example, a Java *main* method) creates both a UI and a domain object, and passes the domain object to the UI.
- A UI object retrieves the domain object from a well-known source, such as a factory object that is responsible for creating domain objects.

3. Write a note on interface and domain layer responsibilities.

(2Marks)

The UI layer should not have any domain logic responsibilities. It should only be responsible for user interface tasks, such as updating widgets. The UI layer should forward requests for all domain-oriented tasks on to the domain layer, which is responsible for handling them.

4. Define – Responsibilities and Methods

(2Marks) The UML defines a responsibility as "a contract or obligation of a classifier".

Responsibilities are related to the obligations of an object in terms of its behaviour. Basically, these responsibilities are of the following two types:

- knowing
- doing

Doing responsibilities of an object include:

- doing something itself, such as creating an object or doing a calculation
- initiating action in other objects
- controlling and coordinating activities in other objects

Knowing responsibilities of an object include:

- knowing about private encapsulated data
- knowing about related objects
- knowing about things it can derive or calculate

5. What is responsibility-driven design?

(2Marks)

A popular way of thinking about the design of software objects and also larger scale components are in terms of responsibilities, roles, and collaborations. This is part of a larger approach called responsibility-driven design or RDD.

6. What is meant by responsibilities?

(2Marks)

The UML defines a responsibility as "a contract or obligation of a classifier". Responsibilities are related to the obligations or behavior of an object in terms of its role.

7. List the types of responsibilities.

(2Marks)

The responsibilities are of the following two types: doing and knowing.

Doing responsibilities of an object include:

- doing something itself, such as creating an object or doing a calculation

- initiating action in other objects
- controlling and coordinating activities in other objects

Knowing responsibilities of an object include:

- knowing about private encapsulated data
- knowing about related objects
- knowing about things it can derive or calculate

Creator

8. Who is creator? (2Marks)

Assign class B the responsibility to create an instance of class A if one or more of the following is true:

- B aggregates an object.
- B contains an object.
- B records instances of objects.
- B closely uses objects.
- B has the initializing data that will be passed to A when it is created (thus B is an Expert with respect to creating A).
- B is a creator of an object.
- If more than one option applies, prefer a class B which aggregates or contains class A.

9. Define–Creator (2Marks)

Creation of objects is one of the most common activities in an object-oriented system. Which class is responsible for creating objects is a fundamental property of the relationship between objects of particular classes.

10. Define –Modular Design (2Marks)

Coupling and cohesion are old principles in software design; designing with objects does not imply ignoring well-established fundamentals. Another of these. Which is strongly related to coupling and cohesion? is to promote modular design.

11. What are the advantages of factory objects? (2Marks)

The advantages are:

- Separate the responsibility of complex creation into cohesive helper objects.
- Hide potentially complex creation logic.
- Allow introduction of performance-enhancing memory management strategies, such as object caching or recycling.

12. List some Abstract for Factory (GoF) for Families of Related Objects.

(2Marks) The Java POS implementations will be purchased from manufacturers.

For example:

```
// IBM's drivers
com.ibm.pos.jpos.CashDrawer (implements jpos.CashDrawer)
com.ibm.pos.jpos.CoinDispenser (implements jpos.CoinDispenser)
// NCR's drivers
com.ncr.posdrivers.CashDrawer (implements jpos.CashDrawer)
com.ncr.posdrivers.CoinDispenser (implements
jpos.CoinDispenser)
```

13. What is meant by abstract class abstract factory?

(2Marks)

A common variation on Abstract Factory is to create an abstract class factory that is accessed using the Singleton pattern, reads from a system property to decide which of its subclass factories to create, and then returns the appropriate subclass instance. This is used, for example, in the Java libraries with the *java.awt.Toolkit* class, which is an abstract class abstract factory for creating families of GUI widgets for different operating systems and GUI subsystems

14. What is meant by fine-grained classes?

(2Marks)

Consider the creation of the Credit Card, Drivers License, and Check software objects. Our first impulse might be to record the data they hold simply in their related payment classes, and eliminate such fine-grained classes. However, it is usually a more profitable strategy to use them; they often end up providing useful behavior and being reusable. For example, the Credit Card is a natural Expert on telling you its credit company type (Visa, MasterCard, and so on). This behavior will turn out to be necessary for our application.

Low Coupling

15.

Coupling
Define –Low

(2Marks)

Low Coupling is an evaluative pattern, which dictates how to assign responsibilities to support:

- Low dependency between classes;
- Low impact in a class of changes in other classes;
- High reuse potential;

16. Define– Coupling

(A/M-11, M/J-

12)(2Marks) The degree to which components depend on one another. There are two types of coupling, "tight" and "loose". Loose coupling is desirable for good software engineering but tight coupling may be necessary for maximum performance. Coupling is increased when the data exchanged between components becomes larger or more complex. Coupling is the degree to which one class knows about another class. Let us consider two classes class **A** and class **B**. If class **A** knows class **B** through its interface only i.e., it interacts with class **B** through its API then class **A** and class **B** are said to be loosely coupled. If on the other hand class **A** apart from interacting class **B** by means of its interface also interacts through the non-interface stuff of class **B** then they are said to be tightly coupled.

Applying GoF design patterns

17. Define–Patterns

(2Marks)

A pattern is a named problem/solution pair that can be applied in new context, with advice on how to apply it in novel situations and discussion of its trade-offs.

18. List the GRASP Patterns.

(2Marks)

The first five GRASP patterns are:

- . Information Expert
- . Creator
- . High Cohesion
- . Low Coupling
- . Controller

19. What is meant by patterns?

(M/J-13, M/J-12)

(2Marks) Design patterns are supposed to provide a structure in which problems can be solved. When solving a real problem, you have to consider many tiny variations of a solution to that problem to see whether any fits a design pattern. In particular, you will probably need to generalise your problem, or its solution, in order to make a design pattern fit.

20. What is meant by GRASP? (M/J-13)

(2Marks)

GRASP (General Responsibility Assignment Software Patterns) provides guidance for assigning responsibilities to classes and, to a limited extent, determining the classes that will be in a design in an object-oriented system. The different patterns and principles used in GRASP are: Controller, Creator, Indirection, Information Expert, High Cohesion, Low Coupling, Polymorphism, Protected Variations, and Pure Fabrication. All these patterns answer some software problem, and in almost every case these problems are common to almost every software development project.

21. Write a note on patterns.

(N/D-11)

(2Marks) Design pattern identifies the key aspects of a common design structure that makes it useful for creating a reusable object oriented design. Furthermore, it identifies the participating classes and instances, their roles and collaborations, and the distribution of responsibilities. It describes when it applies, whether it can be applied in view of other design constraints and the consequences and tradeoffs of its use.

A pattern is an instructive information that captures the essential structure and insight of a successful family of proven solutions to a recurring problem that arises within a certain context and system of forces.

22. Define – Object

(N/D-12) (2Marks)

Objects are key to understanding object-oriented technology. Real-world objects share two characteristics: They all have state and behavior. Dogs have state (name, color, breed, hungry) and behaviour (barking, fetching, wagging tail). Bicycles also have state (current gear, current pedal cadence, current speed) and behavior (changing gear, changing pedal cadence, applying brakes). Identifying the state and behavior for real-world objects is a great way to begin thinking in terms of object-oriented programming.

High Cohesion

23. List out some scenarios that illustrate varying degrees of functional cohesion.

(2Marks) Some scenarios that illustrate varying degrees of functional cohesion are,

- Very low cohesion
- Low cohesion
- High cohesion
- Moderate cohesion

24. Define – High Cohesion

(2Marks)

High Cohesion is an evaluative pattern that attempts to keep objects appropriately focused, manageable and understandable. High cohesion is generally used in support of Low Coupling. High cohesion means that the responsibilities of a given element are strongly related and highly focused. Breaking programs into classes and subsystems is an example of activities that increase the cohesive properties of a system.

25. What is meant by high cohesion?

(N/D-12) (2Marks)

Cohesion refers to the measure of how strongly-related the functions of a module. High cohesion refers to modules that have functions that are similar in many aspects.

The benefits of high cohesion are

1. Readability – (closely) related functions are contained in a single module
2. Maintainability – debugging tends to be contained in a single module
3. Reusability – classes that have concentrated functionalities are not polluted with useless functions

26. Distinguish between coupling and cohesion.

(N/D-11) (2Marks)

Cohesion	Coupling
Cohesion refers to what the class (or module) will do. Low cohesion would mean that the class does a great variety of actions and is not focused on what it should do. High cohesion would then mean that the class is focused on what it should be doing, i.e. only methods relating to the intention of the class.	It refers to how related are two classes / modules and how dependent they are on each other. Being low coupling would mean that changing something major in one class should not affect the other. High coupling would make your code difficult to make changes as well as to maintain it, as classes are coupled closely together, making a change could mean an entire system revamp.
High cohesion <i>within</i> modules	Low coupling <i>between</i> modules.
Increased cohesion and decreased coupling do lead to good software design.	The most effective method of decreasing coupling and increasing cohesion is design by interface .

Controller

27. What is controller?

(2Marks)

The Controller pattern assigns the responsibility of dealing with system events to a non-UI class that represent the overall system or a use case scenario. A Controller object is a non-user interface object responsible for receiving or handling a system event.

Information expert

28. What is information expert?

(2Marks)

Information Expert is a principle used to determine where to delegate responsibilities. These responsibilities include methods, computed fields and so on. Using the principle of Information Expert a general approach to assigning responsibilities is to look at a given responsibility, determine the information needed to fulfil it, and then determine where that information is stored. Information Expert will lead to placing the responsibility on the class with the most information required to fulfil it.

Adapter, singleton, factory and observer patterns.**29. What is singleton pattern? (2Marks)**

The singleton pattern is a design pattern used to implement the mathematical concept of a singleton, by restricting the instantiation of a class to one object. This is useful when exactly one object is needed to coordinate actions across the system.

30. What is adapter pattern? (2Marks)

The adapter pattern is a design pattern that translates one interface for a class into a compatible interface. An adapter allows classes to work together that normally could not because of incompatible interfaces, by providing its interface to clients while using the original interface. The adapter is also responsible for transforming data into appropriate forms.

31. What is facade pattern? (2Marks)

A facade is an object that provides a simplified interface to a larger body of code, such as a class library. A facade can:

- make a software library easier to use, understand and test, since the facade has convenient methods for common tasks;
- make code that uses the library more readable, for the same reason;
- reduce dependencies of outside code on the inner workings of a library, since most code uses the facade, thus allowing more flexibility in developing the system;
- wrap a poorly-designed collection of APIs with a single well-designed API (as per task needs).

32. What is observer pattern? (2Marks)

The observer pattern (a subset of the publish/subscribe pattern) is a software design pattern in which an object, called the subject, maintains a list of its dependents, called observers, and notifies them automatically of any state changes, usually by calling one of their methods. It is mainly used to implement distributed event handling systems.

UNIT III

Elaboration

1. What is an elaboration?

(M/J – 12) (2Marks)

Elaboration is used to build the core architecture, resolve the high-risk elements, define most requirements, and estimate the overall schedule and resources.

2. List the tasks performed in elaboration.

(2Marks)

The tasks performed in elaboration are:

1. The core, risky software architecture is programmed and tested
2. The majority of requirements are discovered and stabilized
3. The major risks are mitigated or retired

3. What are the key ideas and best practices that will manifest in elaboration?

(2Marks)

The key ideas and best practices are:

1. Do a short time boxed risk-driven iterations
2. Start programming early
3. Adaptively design, implement, and test the core and risky parts of the architecture
4. Test early, often, realistically
5. Adapt based on feedback from tests, users, developers
6. Write most of the use cases and other requirements in detail, through a series of workshops, once per elaboration iteration

Domain Models

4. What is a domain model?

(A/M – 11) (2Marks)

A domain model is a visual representation of conceptual classes or real-world objects in a domain of interest. They have also been called conceptual models, domain object models, and analysis object models.

5. List the steps involved in creating a domain model.

(2Marks) The steps involved in creating a domain model are,

1. Find the conceptual classes
2. Draw them as classes in a UML class diagram
3. Add associations and attributes

6. List the ways to illustrate a domain model.

(2Marks)

Applying UML notation, a domain model is illustrated with a set of class diagrams in which no operations (method signatures) are defined. It provides a conceptual perspective. It shows the following,

1. Domain objects or conceptual classes

2. Associations between conceptual classes
3. Attributes of conceptual classes

7. Why domain model is called as a "Visual Dictionary"? (2Marks)

The information domain model illustrates could alternatively have been expressed in plain text. But it's easy to understand the terms and especially their relationships in a visual language, since our brains are good at understanding visual elements and line connections. Therefore, the domain model is a visual dictionary of the noteworthy abstractions, domain vocabulary, and information content of the domain.

8. List the elements not suitable in a domain model.

(2Marks) The elements not suitable in a domain model are:

1. Software artifacts, such as a window or a database, unless the domain being modeled are of software concepts, such as a model of graphical user interfaces.
2. Responsibilities or methods.

Finding conceptual classes and description classes

9. Define –Conceptual Classes (2Marks)

The conceptual class is defined as an idea, thing, or object. More formally, a conceptual class may be considered in terms of its symbol, intension, and extension.

10. Define –Description Class (2Marks)

A description class is defined as information that describes something else. For example, a Product Description that records the price, picture, and text description of an Item.

11. List the three strategies to find conceptual classes.

(2Marks) The three strategies used to find conceptual classes are:

1. Reuse or modify existing models.
2. Use a category list.
3. Identify noun phrases

Associations

12. What is an association? (2Marks)

An association is a relationship between classes (more precisely, instances of those classes) that indicates some meaningful and interesting connection.

13. Draw the UML notation for an association. (2Marks)

Associations are defined as semantic relationship between two or more classifiers that involve connections among their instances.



14. State the reason to avoid adding many associations. (2Marks)

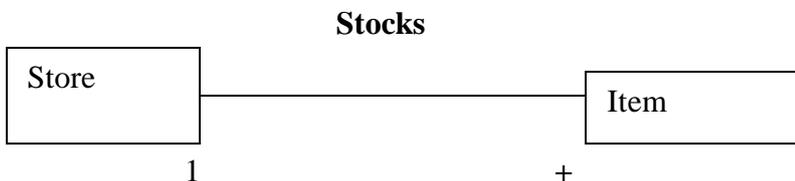
We need to avoid adding too many associations to a domain model. In discrete mathematics, in a graph with n nodes, there can be associations to other nodes a potentially very large number. A domain model with 20 classes could have 190 associations' lines! Many lines on the diagram will obscure it with "visual noise." Therefore, be parsimonious about adding association lines.

15. Write the format to name an association in UML. (2Marks)

Name in an association is based on a Class Name-Verb Phrase- Class Name format where the verb phrase creates a sequence that is readable and meaningful.

16. What is multiplicity? (2Marks)

Multiplicity defines how many instances of class A can be associated with one instance of a class B.


17. What are association role names? (2Marks)

Each end of an association is a role, which has various properties, such as:

1. Name
2. Multiplicity

A role name identifies an end of an association and ideally describes the role played by objects in the association.

18. What is qualified association? (2Marks)

A qualifier may be used in an association; it distinguishes the set of objects at the far end of the association based on the qualifier value. An association with a qualifier is a qualified association. For example, ProductDescriptions may be distinguished in a ProductCatalog by their itemID.

Attributes

19. What is an attribute? (2Marks)

An attribute is a logical data value of an object. It is useful to identify those attributes of conceptual classes that are needed to satisfy the information requirements of the current scenarios under development.

20. What is a derived attribute? (2Marks)

The derived attribute is an attribute whose value is calculated from some other attribute. The total attribute in the Sale can be calculated or derived from the information in the SalesLineItems.

Domain Model Refinement

21. How does domain model is further refined after the first iteration? (2Marks)

Generalization and specializations are fundamental concepts in domain modeling. Conceptual class hierarchies are often inspiration for software class hierarchies that exploits inheritance and reduce duplication of code.

Packages are a way to organize large domain models into smaller units.

Domain model is further refined with Generalization, Specialization, Association classes, Time intervals, Composition and packages, usage of subclasses.

22. State the way to develop a domain model incrementally. (2Marks)

The domain model is incrementally developed by considering concepts in the requirements for this iteration.

23. What is generalization? (2Marks)

Generalization is the activity of identifying commonality among concepts and defining superclass (general concept) and subclass (specialized concept) relationships.

24. Differentiate single inheritance from multiple inheritances. (N/D –12) (2Marks)

In single inheritance one class inherits its state (attributes) and behavior from exactly one super class. When one class inherits its state (attributes) and behavior from more than one super class, it is referred to as multiple inheritances.

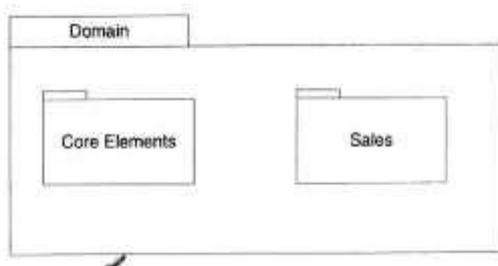
25. What is the need for packages? (2Marks)

A domain model can easily grow large enough that it is desirable to factor it into packages of strongly related concepts, as an aid to comprehension and parallel analysis work in which different people do domain analysis within different sub-domains.

Following Sections illustrate a package structure for the UP Domain Model:

1. UML Package Notation
2. Ownership and References
3. Package Dependencies

26. Draw the UML Package with an example. (2Marks)



A UML package is shown as tabbed folder as shown above. Subordinate packages may be shown within it. The package name is within the tab if the package depicts its elements; otherwise, it is centered within the folder itself.

27. What is packagedependency?

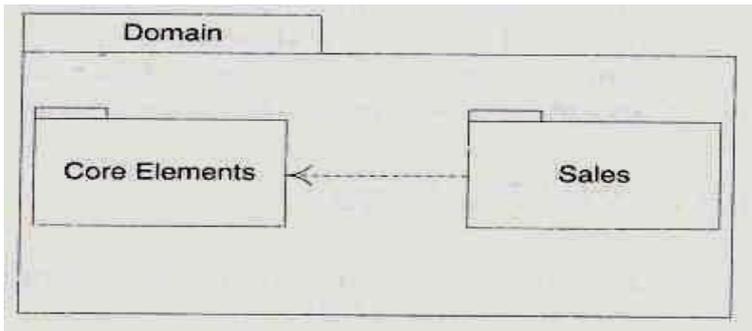
(2Marks)

If a model element is in some way dependent on another,

1. The dependency may be shown with a dependencyrelationship
2. Depicted with ArrowedLine
3. Apackagedependencyindicateselementsofthedependentpackageknowaboutorare coupled to elements in the targetpackage

Example:

The sales package has a dependency on the Core Elements package.



Finding conceptual class hierarchies

28. What are the uses of defining conceptual super classesandsubclasses?

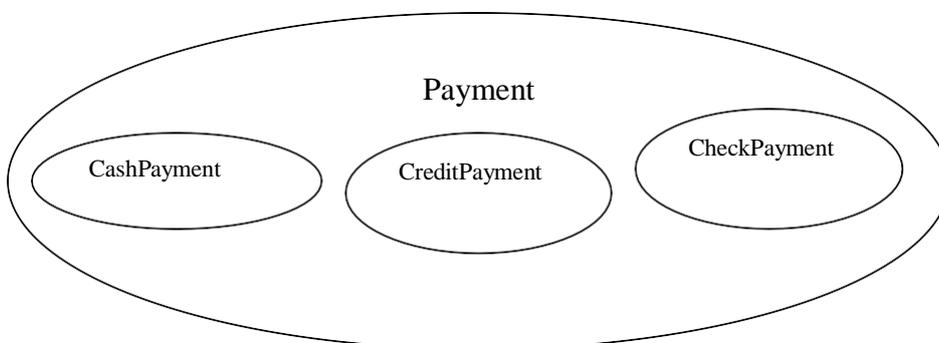
(2Marks)

Defining is valuable to identify conceptual super and subclasses, it is useful to clearly and precisely understand generalization, super classes, and subclasses in terms of class definition and class sets.

29. State the role of conceptual subclass and super classes insetmembership.

(2Marks)

Conceptual subclasses and super classes are related in terms of set membership. By definition, all members of a conceptual subclass set are members of their super class set. For example, in terms of set membership, all instances of the set CreditPayment are also members of the set Payment. This is shown in the Venn diagram shownbelow.



30. What is 100% rule?

(2Marks)

100% of the conceptual Super class's definition should be applicable to the subclass. The subclass must conform to 100% of the Super class's attributes and associations.

31. Write the guidelines followed in defining a superclass.

(2Marks)

The guidelines are:

Create a super class in a generalization relationship to subclasses when :

1. The potential conceptual subclasses represent variations of a similar concept
2. The subclasses will conform to the 100% and Is-A rules
3. All subclasses have the same attribute that can be factored out and expressed in the super class
4. All subclasses have the same association that can be factored out and related to the super class

32. What are the strong motivations to partition a conceptual class with subclasses? (2 Marks)

The following are the strong motivations to partition a class into subclasses:

Create a conceptual subclass of a super class when :

1. The subclass has additional attributes of interest
2. The subclass has additional associations of interest
3. The subclass concept is operated on, handled, reacted-to, or manipulated differently than the super class or other subclasses

Aggregation and Composition

33. What is aggregation?

(A/M – 11, M/J – 12) (2Marks)

Aggregation is a vague kind of association in the UML that loosely suggests whole-part relationships (as do many ordinary associations). It has no meaningful distinct semantics in the UML versus a plain association, but the term is defined in the UML.

34. What is composition?

(A/M – 11, M/J – 12) (2Marks)

Composition, also known as composite aggregation, is a strong kind of whole-part aggregation and is useful to show in some models. A composition relationship implies that 1) an instance of the part belongs to only one composite instance at a time, 2) the part must always belong to a composite and 3) the composite is responsible for the creation and deletion of its parts either by itself creating/deleting the parts, or by collaborating with other objects.

35. What are the benefits of showing composition?

(2Marks)

Showing composition clarifies the domain constraints regarding the eligible existence of the part independent of the whole. In composite aggregation, the part may not exist outside of the lifetime of the whole.

1. During design work, this has an impact on the create-delete dependencies between the whole and part software classes and database elements (in terms of referential integrity and cascading delete paths)

2. It consists in the identification of a creator (the composite) using the GRASP Creator Pattern.
3. Operations – such as copy and delete – applied to the whole often propagate to the parts.

Inception -Use case Modeling

36. Define–Inception (2Marks)

The inception is defined as an envision of product scope, vision, and business case.

The purpose of inception stage is not to define all the requirements. The Up is not the waterfall and the first phase inception is not the time to do all requirements or creates believable estimates or plans. That happens during elaboration.

37. State the intent of inception phase. (2Marks)

The intent of inception is to establish some initial common vision for the objectives of the project, determine if it is feasible, and decide if it is worth some serious investigation in elaboration. It can be brief.

38. Define–Requirements (2Marks)

Requirements are capabilities and conditions, to which the system and more broadly, the project must conform,

- 1) The UP promotes a set of best practices, one of which is to manage requirements.
- 2) In the context of changing and unclear stakeholder's wishes – Managing requirements means a systematic approach to finding, documenting, organizing, and tracking the changing requirements of a system.

A prime challenge of requirements analysis is to find, communicate, and remember (To write down) what is really needed, in a form that clearly speaks to the client and development team members.

39. List the types and categories of requirements.

(2Marks)

In the UP, requirements are categorized as the following,

- 2) Functional - features, capabilities, security.
- 3) Usability - human factors, help, documentation.
- 4) Reliability - frequency of failure, recoverability, predictability.
- 5) Performance - response times, throughput, accuracy, availability, resource usage.
- 6) Supportability - adaptability, maintainability, internationalization, configurability.

40. List the key requirements of artifacts.

(2Marks)

The Key requirements artifacts are:

- 1) Use Case Model - A set of typical scenarios of using a system. There are primarily for functional (behavioural) requirements.
- 2) Supplementary Specification - Basically, everything not in the use cases. This artifact is primarily for all non-functional requirements, such as performance or licensing. It is also the

place to record functional features not expressed (or expressible) as use cases; for example, a report generation.

- 3) Glossary - In its simplest form, the Glossary defines noteworthy terms. It also encompasses the concept of the data dictionary, which records requirements related to data, such as validation rules, acceptable values, and so forth. The Glossary can detail any element: an attribute of an object, a parameter of an operation call, a report layout, and so forth.
- 4) Vision - Summarizes high-level requirements that are elaborated in the Use-Case Model and Supplementary Specification, and summarizes the business case for the project.
- 5) Business Rules - Business rules (also called Domain Rules) typically describe requirements or policies that transcend one software project they are required in the domain or business, and many applications may need to conform to them. An excellent example is government tax laws. Domain rule details may be recorded in the Supplementary Specification, but because they are usually more enduring and applicable than for one software project, placing them in a central Business Rules artifact (shared by all analysts of the company) makes for better reuse of the analysis effort.

41. Define -- Actors, Scenarios and Use Cases

(A/M-08) (2Marks)

An actor is something with behaviour, such as a person (identified by role), computer system, or organization; for example, a cashier.

A scenario is a specific sequence of actions and interactions between actors and the system; it is also called a use case instance. It is one particular story of using a system, or one path through the use case; for example, the scenario of successfully purchasing items with cash, or the scenario of failing to purchase items because of a credit payment denial.

A use case is a collection of related success and failure scenarios that describe an actor using a system to support a goal.

42. Define – Use Case Modelling

(2Marks)

Use-Case Model is the set of all written use cases; it is a model of the system's functionality and environment. Use cases are text documents, not diagrams, and use-case modeling is primarily an act of writing text, not drawing diagrams.

The Use-Case Model is not the only requirement artifact in the UP. There are also the Supplementary Specification, Glossary, Vision, and Business Rules. These are all useful for requirements analysis, but secondary at this point.

The Use-Case Model may optionally include a UML use case diagram to show the names of use cases and actors, and their relationships. This gives a nice context diagram of a system and its environment. It also provides a quick way to list the use cases by name.

43. List the three kinds of actors.

(2Marks)

Actors are roles played not only by people, but by organizations, software, and machines. The three kinds of external actors in relation to the SuD are,

- 1) Primary actor has user goals fulfilled through using services of the SuD. For example, the cashier.
- 2) Supporting actor provides a service (for example, information) to the SuD. The automated

payment authorization service is an example. Often a computer system, but could be an organization or person.

- 3) Offstage actor has an interest in the behaviour of the use case, but is not primary or supporting; for example, a government tax agency.

44. What are the preconditions and post conditions of scenarios?

(2 Marks) Pre

conditions state what must always be true before a scenario is begun in the use case. Preconditions are not tested within the use case; rather, they are conditions that are assumed to be true. Typically, a precondition implies a scenario of another use case, such as logging in, that has successfully completed.

Success guarantees (or post conditions) state what must be true on successful completion of the use case either the main success scenario or some alternate path. The guarantee should meet the needs of all stakeholders.

EXAMPLE:

Preconditions: Cashier is identified and authenticated.

Success Guarantee (Post conditions): Sale is saved. Tax is correctly calculated. Accounting and Inventory are updated. Commissions recorded. Receipt is generated.

45. List the steps to find use cases.

(N/D-

12) (2 Marks) Use cases are defined to satisfy the goals of the primary actors. Hence, the basic procedures are,

- 1) Choose the system boundary
- 2) Identify the primary actors those that have goals fulfilled through using services of the system.
- 3) Identify the goals for each primary actor.
- 4) Define use cases that satisfy user goals; name them according to their goal.

UNIT IV

System Sequence Diagram

1. What is the system sequence diagram? Mention its use.

(A/M-11) (M/J-12)(N/D-11) (2Marks)

A system sequence diagram (SSD) is a picture that shows, for a particular scenario of a use case, the events that external actors generate their order, and inter-system events. All systems are treated as a black box; the emphasis of the diagram is events that cross the system boundary from actors to systems.

2. What is meant by sequence number in UML? Where and for what it is used?

(N/D-11) (2 Marks)

Sequence number can be of:

- Automatic - Sequence numbers of messages will be updated automatically after you moved the messages, and you cannot specify custom sequence number to messages.
- Manual - Sequence numbers of messages will not be updated even after you moved the messages, and you can specify custom sequence number to messages.

Uses:

- Trace external interactions with the software
- Plan the internal behavior of the application
- Study the software structure
- View the system architecture
- Trace behavior down to physical components

3. List the relationships used in class diagram.

(A/M-11) (2Marks)

Category	Function
Activity edges	Represent the flow between activities
Associations	Indicate that instances of one model element are connected to instances of another model element
Dependencies	Indicate that a change to one model element can affect another model element
Generalizations	Indicate that one model element is a specialization of another model element
Realizations	Indicate that one model element provides a specification that another model element implements
Transitions	Represent changes in state

4. What is meant by system behavior?

(2Marks) System behavior is a description of *what* a system does, without explaining *how* it does it. One

Part of that description is a system sequence diagram. Other parts include the Use cases, and system

contracts.

5. What is meant by inter-system SSDs?

(2Marks)SS

Ds can also be used to illustrate collaborations between systems, such as between the NextGen POS and the external credit payment authorizer. However, this is deferred until a later iteration in the case study, since this iteration does not include remote systems collaboration.

6. Define – System Events, System Boundary

(2Marks)

To identify system events, it is necessary to be clear on the choice of system boundary, as discussed in the prior chapter on use cases. For the purposes of software development, the system boundary is usually chosen to be the software system itself; in this context, a system event is an external event that directly stimulates the software.

7. Write short note on naming system events and operations.

(2Marks)

System events (and their associated system operations) should be expressed at the level of intent rather than in terms of the physical input medium or interface widget level.

It also improves clarity to start the name of a system event with a verb Thus "enter item" is better than "scan" (that is, laser scan) because it captures the intent of the operation while remaining abstract and noncommittal with respect to design choices about what interface is used to capture the system event.

Logical architecture and UML Package Diagram

8. Define–Package

(M/J- 12) (2Marks)

A package groups and manages the modelling elements, such as classes, their associations, and their structures. Packages themselves may be nested within other packages. A package may contain both other packages and ordinary model elements. The entire system description can be thought of as a single high-level sub-system package with everything else init. All kinds of UML model elements and diagrams can be organized into packages.

9. What are interactive diagrams? List the components involved in interactive diagrams.

(N/D-12) (2 Marks)

Interaction diagrams are diagrams that describe how groups of objects collaborate to get the job done interaction diagrams capture the behavior of the single use case, showing the pattern of interaction among objects.

There are two kinds of interaction models,

- SequenceDiagram
- CollaborationDiagram.

10. What are the uses of UML component diagram?

(A/M-

10)(2Marks)The uses of UML component diagram are:

- Model the components of a system.
- Model database schema.
- Model executables of an application.
- Model system's source code.

11. What is Software Architecture?

(2Marks)

An architecture is the set of significant decisions about the organization of a software system, the selection of the structural elements and their interfaces by which the system is composed, together

with their behavior as specified in the collaborations among those elements, the composition of these structural and behavioral elements into progressively larger subsystems, and the architectural style that guides this organization these elements and their interfaces, their collaborations, and their composition.

Logical Architecture Refinement

12. What is a layer? (2Marks)

A layer is a very coarse-grained grouping of classes, packages, or subsystems that has cohesive responsibility for a major aspect of the system. Also, layers are organized such that "higher" layers (such as the UI layer) call upon services of "lower" layers, but not normally vice versa.

13. What are the UML Operations and method? (2Marks)

A UML operation is a declaration, with a name, parameters, return type, exceptions list, and possibly a set of constraints of pre- and post-conditions. But, it isn't an implementation rather, methods are implementations.

A UML method is the implementation of an operation; if constraints are defined, the method must satisfy them. A method may be illustrated several ways, including:

- In interaction diagrams, by the details and sequence of messages
- In class diagrams, with a UML note symbol stereotyped with «method»

14. What is the logical architecture? (2Marks)

The logical architecture is the large-scale organization of the software classes into packages (or namespaces), subsystems, and layers. It's called the logical architecture because there's no decision about how these elements are deployed across different operating system processes or across physical computers in a network.

15. Define – Classifier (2Marks)

A UML classifier is a model element that describes behavioral and structure features. Classifiers can also be specialized. They are a generalization of many of the elements of the UML, including classes, interfaces, use cases, and actors. In class diagrams, the two most common classifiers are regular classes and interfaces.

16. What are the connection between SSDs, system operations, and layers? (2Marks)

The SSDs illustrate these system operations, but hide the specific UI objects. Nevertheless, normally it will be objects in the UI layer of the system that capture these system operation requests, usually with a rich client GUI or Web page.

UML Class Diagrams

17. What are UML properties and property strings? (2Marks)

In the UML, a property is a named value denoting a characteristic of an element. A property has semantic impact. Some properties are predefined in the UML, such as visibility a property of an operation. Others can be user-defined.

Properties of elements may be presented in many ways, but a textual approach is to use the UML property string {name1=value1, name2=value2} format, such as {abstract, visibility=public}. Some properties are shown without a value, such as {abstract}; this usually implies a boolean property, shorthand for {abstract=true}. Note that {abstract} is both an example of a constraint and a property string.

18. What is an association class? (2Marks)

An association class allows you to treat an association itself as a class, and model it with attributes, operations, and other features. For example, if a Company employs many Persons, modelled with an Employs association, you can model the association itself as the Employment class, with attributes such as startDate.

19. List the relationships used in class diagram? (M/J - 12)(2Marks)

The following are the relationships used in class diagram:

1. Generalization (class to class)
2. Association (object to object)
3. Aggregation (object to object)
4. Composition (object to object)

20. What is design class diagram (DCD)? (2Marks)

It can be explored, the same UML diagram can be used in multiple perspectives (Figure below). In a conceptual perspective the class diagram can be used to visualize a domain model. A unique term to clarify when the class diagram is used in software or design perspective is called design class diagram (DCD), and all DCDs form part of the Design Model.

Package diagrams

21. How to show methods in class diagrams? (2Marks)

A UML method is the implementation of an operation; if constraints are defined, the method must satisfy them. A method may be illustrated several ways, including: in interaction diagrams, by the details and sequence of messages in class diagrams, with a UML note symbol stereotyped with «method».

22. Define – Keywords, Stereotype, Profiles and Tags (2Marks)

Keywords, stereotypes are shown with guillemot's symbols, such as «authorship». But, they are not keywords, which can be confusing. A stereotype represents a refinement of an existing modelling concept and is defined within a UML profile—informally, a collection of related stereotypes, tags, and constraints to specialize the use of the UML for a specific domain or platform, such as a UML profile for project management or for data modelling. The UML predefines many stereotypes, such as «destroy» (used on sequence diagrams), and also allows user-defined ones. Thus, stereotypes provide an extension mechanism in the UML.

UML Interaction Diagrams

23. What is the use of interaction diagram? (M/J – 13)(2Marks)

The term interaction diagram is a generalization of two more specialized UML diagram types; both can be used to express similar message interactions:

- Collaboration diagrams
- Sequence diagrams

24. What is meant by link? (2Marks)

A link is a connection path between two objects; it indicates some form of navigation and visibility between the objects is possible. More formally, a link is an instance of an association. For example, there is a link or path of navigation from a Register to a Sale, along which messages may flow, such as the make 2 Payment message.

Each message between objects is represented with a message expression and small arrow indicating the direction of the message. Many messages may flow along this link. A sequence number

is added to show the sequential order of messages in the current thread of control.

25. What is an instance?

(2Marks)

Any message can be used to create an instance, but there is a convention in the UML to use a message named *create* for this purpose. If another (perhaps less obvious) message name is used, the message may be annotated with a special feature called a UML stereotype, like so: «create».

The create message may include parameters, indicating the passing of initial values. This indicates, for example, a constructor call with parameters in Java.

UNIT V

Mapping Design to Code

1. List the steps involved in mapping design to code.

(2Marks)

The steps involved in mapping design to code are,

1. Class and interface definitions

The required visibility and associations between classes are indicated by the interaction diagrams.

2. Method definitions

A method body implementation may be shown in a UML note box. It should be placed within braces, it is semantic influence. The syntax may be pseudo-code, or any language. It is common to exclude the method signature (public void ...), but it is legal to include it.

2. What is State-Independent and State-Dependent Objects?

(2Marks)

If an object always responds the same way to an event, then it is considered state independent (or modeless) with respect to that event.

If for all events of interest, an object always reacts the same way, it is a state-independent object. By contrast, state-dependent objects react differently to events depending on their state or mode.

3. Write the attribute presentation suggested by UML?

(2Marks)

OCL can be used during the design phase to define the class attributes. The following is the attribute presentation suggested by UML.

Visibility name: type – expression – initial-value where visibility is

+ public visibility

protected visibility

- private visibility

Type – expression is language dependent specification. Initial – value is language dependent expression for the initial value of a newly created object.

4. What are the three relationships that can be shown in UML diagram? Define them.

(2Marks)

The three relationships are:

1. Association how are objects associated? This information will guide us in designing classes.
2. Super-Sub Structure How are objects organized into super classes and sub classes? This information provides us the direction of inheritance.
3. Aggregation and a part of Structure what is the composition of complex classes? This information guides as in defining mechanisms that properly manage object within object.

5. Define – Database Models

(2Marks)

A database model is a collection of logical constructs representing the data structure and data relationship within the database.

Database models is of two categories

1. Conceptual model

2. Implementation model

Conceptual Model: Focuses on logical nature of data. It deals with what is represented in the database.

Implementation Model: is concerned with how it is represented.



6. Define–Testing (2Marks)

- Testing is generally described as a group of procedures carried out to evaluate some aspect of a piece of software.
- Testing can be described as a process used for revealing defects in software and for establishing that the software has attained a specified degree of quality with respect to selected attributes.

7. State some of the important test related issues. (2Marks)

- There is a demand for software of high quality with low defects;
- Process is important in the software engineering discipline;
- Software testing is an important software development subprocess;
- Existing software evaluation and improvement models have not adequately addressed testing issues.

8. What are the two major goals considered for integrated testing? (2Marks)

- to detect defects that occur on the interfaces of units;
- to assemble the individual units into working subsystems and finally a complete system that is ready for system test.

9. List the issues of class testing. (2Marks)

- Issue1: Adequately testing classes
- Issue2: Observation of object states and state changes.
- Issue3: The retesting of classes-I
- Issue4: The retesting of classes-II

10. What are the challenges of class testing?

(2Marks) OO class is the target for test case design.

 Encapsulation:

Difficult to obtain a snapshot of a class without building extra methods which display the classes' state

 Inheritance and polymorphism:

Each new context of use (subclass) requires re-testing because a method may be implemented differently (polymorphism).

Other unaltered methods within the subclass may use the redefined method and need to be tested.

11. How is class testing different from conventional testing? (2Marks)

Conventional testing focuses on input-process-output, whereas class testing focuses on each method, then designing sequences of methods to exercise states of a class.

12. Write the impact of OO programming on testing. (2Marks)

When an operation is invoked, it may be hard to tell exactly what code gets exercised. It can be hard to determine the exact type or class of a parameter. OO operations are smaller, more time needed for integration. So integration faults become more plausible.

13. List the class testing techniques. (2Marks)

- State transition testing



- Transaction flowtesting
- Exceptiontesting

14. What are the three different incremental strategies of integrationtesting? (2Marks)

- Thread-basedtesting
- Use-basedtesting
- Clustertesting

15. List the types of errors found during integrationtesting. (2Marks)

- Messagingerrors
- User interfaceerrors

16. What are the challenges of GUItesting? (2Marks)

- GUI test automation is harder than API testautomation
 - Documentation; GUIs are slower than APIs
- Observing visible GUI state is difficult
- Observing invisible GUI state is tricky almost impossible
- Controlling GUI actions is difficult

17. List the automated GUItestingtools. (2Marks)

- Capture/Replay testingtools
- Randomtesting
- Unittesting
- Model-Basedtesting



16 Marks Questions

Unit I

PART- B

1. Explain about POS generation systems.

- The Next Gen POS System
- Architectural Layers and Case Study Emphasis
- Iterative Development and Iterative Learning

2. Define Inception. Explain about artifacts of Inception

- Inception: An Analogy
- What Artifacts May Start in Inception
- You Didn't Understand Inception When...

3. Explain about Unified process phases. APRIL/MAY-2011

- Iterative improvement
- UP Practices and Concepts
- The UP Phases and Schedule
- The UP Disciplines (was Workflows)
- The Agile UP
- The Sequential "Waterfall"

4. Explain about Use-Case Model and its Writing Requirements in Context. APRIL/MAY-2011

- Background
- Use Cases and Adding Value
- Use Cases and Functional Requirements
- Use Case Types and Formats
- Fully Dressed Example: Process Sale
- Relating use cases- Include, Exclude, Generalize
- Example with diagram-ATM, Library Management System etc

5. List out the components of Object-Oriented Analysis and Design.

- Applying UML and Patterns in OOA/D
- Assigning Responsibilities
- What Is Analysis and Design?
- What Is Object-Oriented Analysis and Design?
- An Example
- The UML

UNIT-II

PART -B

1. Write briefly about elaboration and discuss the differences between Elaboration and Inception with examples.

- Iteration 1 Requirements and Emphasis: Core OOA/D Skills
- Inception and Elaboration
- Planning the Next Iteration

2. Illustrate the concept of Domain model with examples. APRIL/MAY-2011

- Definitions
- Guidelines for creating domain model
- Examples

3. What is activity diagram? Explain about its applications briefly? APRIL/MAY-2011

- UML Activity Diagram Notation
- Guidelines for activity modeling
- Example –Next Gen Activity Diagram

4. Explain about Aggregations and compositions

- Definitions
- Identify Composition & Aggregations
- Example: the Next Gen Domain Model

Unit III

PART- B

1. How to Adding New SSDs and Contracts?

- New System Sequence Diagrams
- New System Operations
- New System Operation Contracts

2. Explain about Interaction Diagram Notation? APRIL/MAY-2011

- Sequence and Collaboration Diagrams
- Collaboration Diagram
- Sequence Diagram
- Common Interaction Diagram Notation
- Basic Collaboration Diagram
- Notation
- Basic Sequence Diagram Notation

3. Design the Model and Creating Design Class Diagrams.

- When to Create DCDs
- Example DCD
- DCD and UP
- Domain Model vs. Design Model Classes
- DCDs, Drawing, and CASE Tools
- DCDs within the UP

4. What are concepts involved in domain refinement?

- Generalization
- Defining Conceptual Super classes and Subclasses
- Class Hierarchies and Inheritance
- Aggregation and Composition
- Examples

5. Illustrate with an example, the relationship between sequence diagram and use cases.

APRIL/MAY-2011

Unit IV

PART- B

1. Explain Grasp: designing objects with responsibilities.

- Responsibilities and Methods
- Responsibilities and Interaction Diagrams
- Patterns

2. Explain GRASP: Patterns of General Principles in Assigning Responsibilities.

APRIL/MAY-2011

- The UML Class Diagram Notation
- Information Expert (or Expert)
- Creator
- low coupling

-high cohesion

-controller

-object design and CRC CARDS

3. How to Determining the Visibility of the Design Model?

-Visibility between Objects

-Visibility

4. Explain about Patterns for Assigning Responsibilities.

-Polymorphism

-Pure Fabrication

-Indirection

-Protected Variations

5. Designing the Use-Case Realizations with GoF Design Patterns. APRIL/MAY-2011

-Analysis" Discoveries during Design: Domain Model

-Factory

-Singleton

-Conclusion of the External Services with Varying Interfaces Problem 3

-Strategy

-Composite

-Façade

Unit-V

PART- B

1. Explain UML State Machine Diagrams and Modeling.

-Definition

-How to apply

-Example

-Process

2. What is the operation of contracts works.

-Contracts

-Contract Sections

-Post Conditions

-Guidelines: Contracts

-Next Gen POS Example

-Changes to the Domain Model

-Contracts, Operations, and the UML

-Operation Contracts With in the UP

3. Explain the operation of Mapping Designs to Code. APRIL/MAY-2011

- Programming and the Development Process

-Mapping Designs to Code

-Creating Class Definitions from DCDs

-Creating Methods from Interaction Diagrams

-Container/Collection Classes in Code

-Exceptions and Error Handling

-Defining the Sale--makeLineItem Method

-Order of Implementation

-Test-First Programming

4. What is operation of UML Deployment and Component Diagram? Draw the diagram for a banking application. APRIL/MAY-2011

-Deployment Diagram

-Component Diagram

V.S.B Engineering College,karur

Assignment Topics

Subject code / Subject Name : CS6502 / Object Oriented Analysis and Design

YEAR / SEM / SEC : III / V

S.No	Assignment Topic
1	Explain in detail about the Unified process in object oriented Analysis and Design? Explain the phases with neat diagrams.
2	(i)Discuss about the Concepts of Component and Deployment Diagram (ii).Draw component and deployment diagrams for Book bank system
3	A University conducts examinations and the results are announced. Prepare a report for the following: <ul style="list-style-type: none">• Print the marks in the register number order semester wise for each department• Print the Arrear list semester wise.• Prepare a Rank list for each department.• Prepare the final aggregate mark list for final year students. Identify the problem statement and Design and Explain the classes for each sequence. Draw a detailed flow chart using state chart diagrams. Design this system using Rational Rose. Draw all the UML diagrams for designing this system.
4	(i).Describe in detail about the Class Diagram. (ii).What is use case Diagram? Model a use case diagram for a Banking System. Explain the business rules you are considering. b) Consider the following use Cases that play a role in the Banking System you have modeled: 1.Deposit 2.Withdraw Model sequence diagrams for the above two use cases
5	Write a problem statement for Library Management System. Design the UML Use Case diagram, Activity diagram ,Class diagram, Sequence diagram, State chart diagram, Package diagram, Component and Deployment diagram
6	List the various UML diagram and examine the purpose of each diagram
7	(i)Describe the UML notation for Class diagram with an example. Explain the concept of link, association and inheritance.(ii).Identify the major difference between sequence and collaboration diagram
8	(i).Summarize with an example,how Interaction Diagram are used to model the dynamic aspects of asystem. (ii).Discuss the topic on <ul style="list-style-type: none">(i). Aggregation and Composition(ii).Generalization and Specialization.(iii).Attributes and Association

9	<p>Illustrate about UML state machine diagram and Modeling (ii).A Library lends books and magazines to member, who is registered in the system. It also maintains the purchase of new books and magazines for the Library. A member can reserve a book or magazine that is not currently available in the library, so that when it is returned or purchased by the library, that person is notified. The library can easily create, replace and delete information about the books, members, and reservation in the system. The books transactions are stored in the database. The fine list while the member returns the book after the due date must be generated. Analyze and discover the users and actors of this system, and the interactions between them must be depicted.</p>
10	<p>(i)Describe in Detail about the Sequence Diagrams. (ii).Discuss in detail about the three types of different perspectives. (iii).Givethe three ways to apply Unified Modeling Language (UML).</p>
11	<p>(i).What is UML activity diagram?Using an example point out the features of basic UML activity diagram notation. (ii).Draw and explain the Sequence and collaboration diagram for an Online Purchase System.</p>
12	<p>(i).Explain with an example, how use case modeling is used to describe functional requirements, Identify the actors, scenario and use cases for the example. (ii).Comparison between Activity and State chart Diagram.</p>
13	<p>(i).Describe UML Package diagram with example. (ii).When to use Activity and State chart diagram?</p>
14	<p>Illustrate about UML Deployment and Component diagram with an example</p>
15	<p>Explain GRASP:Designing objects with responsibilities.</p>
16	<p>What is GRASP? Describe the design patterns and principles used in it.</p>
17	<p>Examine the following GRASP patterns: (i)Creator, (ii).Information Expert, (iii)Low coupling, (iv).High cohesion</p>
18	<p>(i).Explain about Creator and information Expert with example. (ii).Explain the Benefits of Low coupling and difficulties of Low cohesion</p>
19	<p>Compare cohesion and coupling with suitable example. Summarize and state the role and patterns while developing system design.</p>
20	<p>(i).Generalize your idea on Controller pattern with example (ii).Generalize the concepts of Façade, session and bloated controller.</p>
21	<p>(i).Discuss about Low Coupling and High Cohesion with example. (ii).Describe the different scenarios of functional cohesion.</p>

22	(i).Givean account on Factory method. (ii).Discuss the topic on coupling and mention its types.
23	Differentiate Adapter and Bridge pattern . Analyze how will you design the behavioral pattern.
24	Discuss the topic on (i).Adapter Pattern (ii). Observer Pattern (iii).Factory Pattern
25	(i).Describe about Strategy pattern. (ii).List out the applications of Factory pattern and mention its Limitations
26	(i).Examine in detail about Behavioral pattern.(ii).Describe the concepts of Singleton Pattern.
27	Demonstrate in detail about the various categories of design pattern.
28	(i).Illustrate your views about Structural patterns. (ii).What is Visibility? Classify the ways of visibility and explain it.
29	(i).What is a POS system? Briefly explain about Inception Phase. (ii).Comparison between Association and attributes.
30	Prepare a suitable example showing the various relationships used in Use Case and also give a short note on each relationship
31	(i).Summarize the Elaboration phase. (ii).Discuss the difference between elaboration and inception with example
32	(i).Describe the strategies used to identify the conceptual classes. (ii).Describe the steps to create a domain model used for representing the conceptual classes
33	(i).Illustrate the concepts of Domain model with example. (ii).Show when to model with Description classes with example.
34	(i).Summarize the steps and explain how to find Use cases with an example. (ii).Rank the 3 kinds of actors and explain the 3 common Use Case formats
35	(i).Describe briefly about association and formulate the guidelines to be followed with UML with suitable example. (ii).Describe the concepts of Derived attribute.
36	(i).Discuss about attributes with example. (ii).Discuss the topic on a).Coconceptual subclass b)Conceptual super class c) Multiplicity

37	Explain in detail about domain Model refinement.
38	Analyze the guidelines to define a conceptual subclass with suitable example.
39	(i).What are the guidelines used to partition the classes in the domain model to be organized into packages? Explain with suitable examples. (ii).Describe the benefits of composition
40	(i).Examine the various sections in the Use Case template with example. (ii).List the guidelines to be followed when writing Use case.
41	(i).Describein detail about the Finding Conceptual class Hierarchies. (ii).Describe briefly about association classes and association role.
42	i).Illustrate about aggregation and composition with Example. Mention the guidelines to be followed.
43	i).Illustrate the relationship between sequence diagram and Use Case with example. (ii).Demonstrate the Interaction Diagram notations and explain it?
44	(i).Describe briefly about the logical architecture and UML package diagram. (ii).Identify the relationship between Domain layer and Domain model.
45	(i)What is Model View separation principle? Examine the motivation for Model View separation. (ii)Describe the concepts of Dependency relationship
46	Briefly discuss about the various collaborations with the layers.
47	With an example explain notations used in sequence diagram for the following: (i) Object destruction (ii) Frames (iii) Conditional message (iv) Mutually exclusive conditional message (v) Iterations over a collection
48	(i).Give short notes on inter layer and inter package coupling. (ii).Discuss on the classic 3 tier architecture.
49	Describe how to adding a new System sequence diagram and contracts?
50	Examine with an example about Interaction diagram.

51	Illustrate the topic on (i). Generalization (ii). Specialization (iii). Conceptual class hierarchies.
52	Analyze the guidelines to define a conceptual super class with suitable example.
53	What is use cases and Explain in detail about the sample Unified process Artifacts Relationships
54	Explain with the example, Illustrate how interaction diagram are used to model the dynamic aspects of the system
55	Describe the UML notation for class diagram with an example. Explain the concept of Link, Association and Inheritance.
56	Discuss in detail about Logical Architecture refinement

2 MARKS & 16 MARKS

2 Mark Question

UNIT – I

1. Define Computer graphics.

Computer graphics remains one of the most existing and rapidly growing computer fields. Computer graphics may be defined as a pictorial representation or graphical representation of objects in a computer.

2. What is meant by scan code?

When a key is pressed on the keyboard, the keyboard controller places a code carry to the key pressed into a part of the memory called as the keyboard buffer. This code is called as the scan code.

3. What is meant by refreshing of the screen?

Some method is needed for maintaining the picture on the screen. Refreshing of screen is done by keeping the phosphorus glowing to redraw the picture repeatedly. (i.e.)By quickly directing the electronic beam back to the same points.

4. Define Random scan/Raster scan displays?

Random scan is a method in which the display is made by the electronic beam which is directed only to the points or part of the screen where the picture is to be drawn. The Raster scan system is a scanning technique in which the electrons sweep from top to bottom and from left to right. The intensity is turned on or off to light and unlight the pixel.

5. List out the merits and demerits of Penetration techniques?

The merits and demerits of the Penetration techniques are as follows

- It is an inexpensive technique
- It has only four colors
- The quality of the picture is not good when it is compared to other techniques
- It can display color scans in monitors
- Poor limitation etc.

6. List out the merits and demerits of DVST?

The merits and demerits of direct view storage tubes [DVST] are as follows

- It has a flat screen
- Refreshing of screen is not required
- Selective or part erasing of screen is not possible
- It has poor contrast
- Performance is inferior to the refresh CRT.

7. What do you mean by emissive and non-emissive displays?

- a. The emissive display converts electrical energy into light energy. The plasma panels, thin film electro-luminescent displays are the examples.
- b. The Non emissive are optical effects to convert the sunlight or light from any other source to graphic form. Liquid crystal display is an example.

8. List out the merits and demerits of Plasma panel display?

Merits

- Refreshing is not required
- Produce a very steady image free of Flicker
- Less bulky than a CRT.

Demerits

- Poor resolution of up to 60 d.p.i
- It requires complex addressing and wiring
- It is costlier than CRT.

9. What is persistence?

The time it takes the emitted light from the screen to decay one tenth of its original intensity is called as persistence.

10. What is resolution?

The maximum number of points that can be displayed without an overlap on a CRT is called as resolution.

11. What is Aspect ratio?

The ratio of vertical points to the horizontal points necessary to produce length of lines in both directions of the screen is called the Aspect ratio. Usually the aspect ratio is $\frac{3}{4}$.

12. What is meant by Addressability?

The Addressability is the number of individual dots per inch (d.p.i) that can be created. If the address of the current dot is (x, y) then the next dot will be (x+y), (x+y+1) etc.

13. What is a dot size?

Dot size may be defined as the diameter of a single dot on the devices output. Dot size is also called as the Spot size.

14. What is interdot distance?

Interdot distance is the reciprocal of addressability. If the addressability is large, the interdot distance will be less. The interdot distance should be less to get smooth shapes.

15. What is the difference between impact and non-impact printers?

- a. Impact printer press formed character faces against an inked ribbon on to the paper.
Examples are line printer and dot-matrix printer.
- b. Non-impact printer and plotters use Laser techniques, inkjet sprays, Xerographic process, electrostatic methods and electro thermal methods to get images onto the papers.
Examples are Inkjet/Laser printers.

16. What are the features of Inkjet printers?

- They can print 2 to 4 pages/minutes.
- Resolution is about 360d.p.i. Therefore better print quality is achieved.

- The operating cost is very low. The only part that requires replacement is ink cartridge.
- 4 colors cyan, yellow, magenta, black are available.

17. What are the advantages of laser printer?

- High speed, precision and economy.
- Cheap to maintain.
- Quality printers.
- Lasts for longer time.
- Toner power is very cheap.

18. What are the advantages of electrostatic plotters?

- They are faster than pen plotters and very high quality printers.
- Recent electrostatic plotters include a scan-conversion capability.
- Color electrostatic plotters are available. They make multiple passes over the paper to plot color pictures.

19. Define pixel?

Pixel is shortened form of picture element. Each screen point is referred to as pixel or pel.

20. What is frame buffer?

Picture definition is stored in a memory area called frame buffer or refresh buffer.

21. What is bitmap and what is pixmap?

The frame buffer used in the black and white system is known as bitmap which take one bit per pixel. For systems with multiple bits per pixel, the frame buffer is often referred to as a pixmap.

22. What is a Vector display or stroke writing or calligraphic display?

Random scan monitors draw a picture one line at a time and for this reason are also referred as vector displays.

23. Where the video controller is used?

A special purpose processor, which is used to control the operation of the display device, is known as video controller or display controller.

24. What do you mean by scan conversion?

A major task of the display processor is digitizing a picture definition given in an application program into a set of pixel-intensity values for storage in the frame buffer. This digitization process is called scan conversion.

25. What is an output primitive?

Graphics programming packages provide function to describe a scene in terms of these basic geometric structures, referred to as output primitives.

26. What do you mean by jaggies?

Line with stair step appearance is known as jaggies.

27. What is point in the computer graphics system?

The point is a most basic graphical element & is completely defined by a pair of user coordinates (x, y).

28. Write short notes on lines?

A line of infinite extent can be defined by an angle of slope θ and one point on the line $P=P(x,y)$. This can also be defined as $Y=mx+C$ where C is the Y-intercept.

29. Define Circle?

Circle is defined by its center x_c , y_c and its radius in user coordinate units. The equation of the circle is $(x-x_c)^2 + (y-y_c)^2 = r^2$.

30. Define Ellipse?

An ellipse can use the same parameters x_c , y_c , r as a circle, in addition to the eccentricity e . The eqn of an ellipse is:

$$(x-x_c)^2/a^2 + (y-y_c)^2/b^2 = 1$$

31. Define polygon?

A polygon is any closed continuous sequence of line segments i.e., a polyline whose last node point is same as that of its first node point. The line segments form the sides of the polygon and their intersecting points form the vertices of the polygon.

32. Distinguish between convex and concave polygons?

If the line joining any two points in the polygon lies completely inside the polygon then, they are known as convex polygons. If the line joining any two points in the polygon lies outside the polygon then, they are known as concave polygons.

33. What is seed fill?

One way to fill a polygon is to start from a given point (seed) known to be inside the polygon and highlight outward from this point i.e. neighboring pixels until encounter the boundary pixels, this approach is called seed fill.

34. What is scan line algorithm?

One way to fill the polygon is to apply the inside test. i.e. to check whether the pixel is inside the polygon or outside the polygon and then highlight the pixel which lie inside the polygon. This approach is known as scan-line algorithm.

35. What is an active edge list in the scan line algorithm?

The active edge list for a scan line contains all edges crossed by that scan line.

36. What is a winding number?

Winding number method is used to check whether a given point is inside or outside the polygon. In this method give a direction number to all the edges which cross the scan line. If the edge starts below the line and ends above scan line give direction as -1. otherwise 1. For polygons or two dimensional objects, the point is said to be inside when the value of winding number is nonzero.

37. What is pixel mask?

Pixel mask is a string containing the digits 1 and 0 to indicate which positions to plot along the line path. The mask 1111000, could be used to display a dashed line with a dash length of 4 and inter dot spacing of three.

38. What is a Line cap?

Line caps can be used to adjust the shape of the line ends to give a better appearance. There are three types of line caps. Butt cap which has a square end, round cap which has a semi circle end, projecting square cap which has one half of the line width beyond the specified end points.

39. List out the methods used for smoothly joining two line segments?

- . Mitter join-by extending the outer boundaries of each of the two lines until they meet.
- . Round join – by capping the connection between the two segments with a circular boundary whose diameter is equal to the line width.
- . Bevel join – by displaying the line segments with butt caps and filling in the triangular gap where the segment meet.

40. What is Color Look up table?

In color displays, 24 bits per pixel are commonly used, where 8 bits represent 256 level for each color. It is necessary to read 24-bit for each pixel from frame buffer. This is very time consuming. To avoid this video controller uses look up table to store many entries to pixel values in RGB format. This look up table is commonly known as colour table.

41. What is aliasing?

In the line drawing algorithms, all rasterized locations do not match with the true line and have to represent a straight line. This problem is severe in low resolution screens. In such screens line appears like a stair-step. This effect is known as aliasing.

42. What is antialiasing?

The process of adjusting intensities of the pixels along the line to minimize the effect of aliasing is called antialiasing.

43. What is pixel phasing?

Pixel phasing is an antialiasing technique, stair steps are smoothed out by moving the electron beam to more nearly approximate positions specified by the object geometry.

UNIT-II

1. What is Transformation?

Transformation is the process of introducing changes in the shape size and orientation of the object using scaling rotation reflection shearing & translation etc.

2. Write short notes on active and passive transformations?

In the active transformation

the points x and x^1 represent different coordinates of the same coordinate system. Here all the points are acted upon by the same transformation and hence the shape of the object is not distorted.

In a passive transformation

the points x and x^1 represent same points in the space but in a different coordinate system. Here the change in the coordinates is merely due to the change in the type of the user coordinate system.

3. What is translation?

Translation is the process of changing the position of an object in a straight-line path from one coordinate location to another. Every point (x,y) in the object must under go a displacement to (x^1,y^1) . the transformation is: $x^1 = x + tx$; $y^1 = y + ty$

4. What is rotation?

A 2-D rotation is done by repositioning the coordinates along a circular path, in the x-y plane by making an angle with the axes. The transformation is given by:
 $X^1 = r \cos(\theta + \phi)$ and $Y^1 = r \sin(\theta + \phi)$.

5. What is scaling?

The scaling transformations changes the shape of an object and can be carried out by multiplying each vertex (x,y) by scaling factor S_x, S_y where S_x is the scaling factor of x and S_y is the scaling factor of y .

6. What is shearing?

The shearing transformation actually slants the object along the X direction or the Y direction as required. ie; this transformation slants the shape of an object along a required plane.

7. What is reflection?

The reflection is actually the transformation that produces a mirror image of an object. For this use some angles and lines of reflection.

8. Distinguish between window port & view port?

A portion of a picture that is to be displayed by a window is known as window port. The display area of the part selected or the form in which the selected part is viewed is known as view port.

9. Define clipping?

Clipping is the method of cutting a graphics display to neatly fit a predefined graphics region or the view port.

10. What is covering (exterior clipping)?

This is just opposite to clipping. This removes the lines coming inside the windows and displays the remaining. Covering is mainly used to make labels on the complex pictures.

11. What is the need of homogeneous coordinates?

To perform more than one transformation at a time, use homogeneous coordinates or matrixes. They reduce unwanted calculations intermediate steps saves time and memory and produce a sequence of transformations.

12. Distinguish between uniform scaling and differential scaling?

When the scaling factors s_x and s_y are assigned to the same value, a uniform scaling is produced that maintains relative object proportions. Unequal values for s_x and s_y result in a differential scaling that is often used in design application.

13. What is fixed point scaling?

The location of a scaled object can be controlled by a position called the fixed point that is to remain unchanged after the scaling transformation.

14. Distinguish between bitBlt and pixBlt?

Raster functions that manipulate rectangular pixel arrays are generally referred to as raster ops. Moving a block of pixels from one location to another is also called a block transfer of pixel values. On a bilevel system, this operation is called a bitBlt (bit-block transfer), on multilevel system t is called pixBlt.

15. List out the various Text clipping?

All-or-none string clipping -if all of the string is inside a clip window, keep it otherwise discards.All-or-none character clipping – discard only those characters that are not completely inside the window. Any character that either overlaps or is outside a window boundary is clipped Individual characters – if an individual character overlaps a clip window boundary, clip off the parts of the character that are outside the window.

UNIT-III

1. What are the various representation schemes used in three dimensional objects?

- Boundary representation (B-res) – describe the 3 dimensional object as a set of surfaces that separate the object interior from the environment.
- Space-portioning representation– describe interior properties, by partitioning the spatial region containing an object into a set of small, no overlapping, contiguous solids.

2. What is Polygon mesh?

Polygon mesh is a method to represent the polygon, when the object surfaces are tiled, it is more convenient to specify the surface facets with a mesh function. The various meshes are

Triangle strip – (n-2) connected triangles

Quadrilateral mesh – generates (n-1)(m-1) Quadrilateral

3. What is Bezier Basis Function?

Bezier Basis functions are a set of polynomials, which can be used instead of the primitive polynomial basis, and have some useful properties for interactive curve design.

4. What is surface patch?

A single surface element can be defined as the surface traced out as two parameters (u, v) take all possible values between 0 and 1 in a two-parameter representation. Such a single surface element is known as a surface patch.

5. Write short notes on rendering bi-cubic surface patches of constant u and v method?

The simple way is to draw the iso-parametric lines of the surface. Discrete approximations to curves on the surface are produced by holding one parameter constant and allowing the other to vary at discrete intervals over its whole range. This produce curves of constant u and constant v.

6. What are the advantages of rendering polygons by scan line method?

- i. The max and min values of the scan were easily found.
- ii. The intersection of scan lines with edges is easily calculated by a simple incremental method.
- iii. The depth of the polygon at each pixel is easily calculated by an incremental method.

7. What are the advantages of rendering by patch splitting?

- i. It is fast-especially on workstations with a hardware polygon-rendering pipeline.
- ii. It's speed can be varied by altering the depth of sub-division.

8. Define B-Spline curve?

A B-Spline curve is a set of piecewise(usually cubic) polynomial segments that pass close to a set of control points. However the curve does not pass through these control points, it only passes close to them.

9. What is a spline?

To produce a smooth curve through a designed set of points, a flexible strip called spline is used. Such a spline curve can be mathematically described with a piecewise cubic polynomial function whose first and second derivatives are continuous across various curve section.

10. What is the use of control points?

Spline curve can be specified by giving a set of coordinate positions called control points, which indicates the general shape of the curve, can specify spline curve.

11. What are the different ways of specifying spline curve?

- Using a set of boundary conditions that are imposed on the spline.
- Using the state matrix that characteristics the spline
- Using a set of blending functions that calculate the positions along the curve path by specifying combination of geometric constraints on the curve

12. What are the important properties of Bezier Curve?

- It needs only four control points
- It always passes through the first and last control points
- The curve lies entirely within the convex hull formed by four control points.

13. Differentiate between interpolation spline and approximation spline?

When the spline curve passes through all the control points then it is called interpolate. When the curve is not passing through all the control points then that curve is called approximation spline.

14. What do you mean by parabolic splines?

For parabolic splines a parabola is fitted through the first three points p_1, p_2, p_3 of the data array of knot points. Then a second parabolic arc is found to fit the sequence of points p_2, p_3, p_4 . This continues in this way until a parabolic arc is found to fit through points p_{n-2}, p_{n-1} and p_n . The final plotted curve is a meshing together of all these parabolic arcs.

15. What is cubic spline?

Cubic splines are a straight forward extension of the concepts underlying parabolic spline. The total curve in this case is a sequence of arcs of cubic rather than parabolic curves .

Each cubic satisfies : $ax^3 + bx^2 + cx + d$.

16. What is a Blobby object?

Some objects do not maintain a fixed shape, but change their surface characteristics in certain motions or when in proximity to other objects. That is known as blobby objects. Example – molecular structures, water droplets.

17. Define Octrees?

Hierarchical tree structures called octrees, are used to represent solid objects in some graphics systems. Medical imaging and other applications that require displays of object cross sections commonly use octree representation.

18. Define Projection?

The process of displaying 3D into a 2D display unit is known as projection. The projection transforms 3D objects into a 2D projection plane.

19. What are the steps involved in 3D transformation?

- Modeling Transformation
- Viewing Transformation
- Projection Transformation
- Workstation Transformation

20. What do you mean by view plane?

A view plane is nothing but the film plane in camera which is positioned and oriented for a particular shot of the scene.

21. What is view-plane normal vector?

This normal vector is the direction perpendicular to the view plane and it is called as [DXN DYN DZN]

22. What is view distance?

The view plane normal vector is a directed line segment from the view plane to the view reference point. The length of this directed line segment is referred to as view distance

23. Define projection?

The process of converting the description of objects from world coordinates to viewing coordinates is known as projection

24. What you mean by parallel projection?

Parallel projection is one in which z coordinates is discarded and parallel lines from each vertex on the object are extended until they intersect the view plane.

25. What do you mean by Perspective projection?

Perspective projection is one in which the lines of projection are not parallel. Instead, they all converge at a single point called the center of projection.

26. What is Projection reference point?

In Perspective projection, the lines of projection are not parallel. Instead, they all converge at a single point called Projection reference point.

27. What is the use of Projection reference point?

In Perspective projection, the object positions are transformed to the view plane along these converged projection line and the projected view of an object is determined by calculating the intersection of the converged projection lines with the view plane.

28. What are the different types of parallel projections?

The parallel projections are basically categorized into two types, depending on the relation between the direction of projection and the normal to the view plane. They are orthographic parallel projection and oblique projection.

29. What is orthographic parallel projection?

When the direction of the projection is normal (perpendicular) to the view plane then the projection is known as orthographic parallel projection

30. What is orthographic oblique projection?

When the direction of the projection is not normal (not perpendicular) to the view plane then the projection is known as oblique projection.

UNIT – IV

1. What is illumination?

- Illumination, an observable property and effect of light, may also refer to Lighting, the use of light sources:

Illumination (image), the use of light and shadow in art

Illuminated manuscript, the artistic decoration of hand-written texts

Global illumination, algorithms used in 3D computer graphics

- the transport of light from a source to a point via direct and indirect paths

2. What are subtractive colors?

In four-color printing process, the CMYK colors (Cyan, Magenta, Yellow, and black) which are overlapped in various combinations and proportions to produce all other colors. The visible color is the reflected (and not the retained) color, and is called 'subtractive' because its wavelength is less than sum of the wavelengths of its constituting colors.

* Color specified by what is subtracted from white light

*Cyan absorbs red, magenta absorbs green, and yellow absorbs blue

3. What is RGB color model?

The RGB color model is an additive color model in which red, green and blue light are added together in various ways to reproduce a broad array of colors. The name of the model comes from the initials of the three additive primary colors, red, green, and blue.

4. What is HSV color model?

Hue, Saturation, and Value (HSV) is a color model that is often used in place of the RGB color model in graphics and paint programs. In using this color model, a color is specified then white or black is added to easily make color adjustments. HSV may also be called HSB

5. What is HSL color model?

The HSL model describes colors in terms of hue, saturation, and lightness (also called luminance). (Note: the definition of saturation in HSL is substantially different from HSV, and lightness is not intensity.) The model has two prominent properties:

- The transition from black to a hue to white is symmetric and is controlled solely by increasing lightness
- Decreasing saturation transitions to a shade of gray dependent on the lightness, thus keeping the overall intensity relatively constant

6. What is color look up table?

A color lookup table is a file that changes all the colors of your image to different ones, usually to apply some sort of effect or stylized look such as, for example, changing all the all tones to be much duller with a brown tint.

7. Write the Lamberts cosine law

- Lambert's Cosine Law - reflected energy from a small surface area in a particular direction is proportional to the cosine of the angle between that direction and the surface normal.

8. What is Polygon shading?

Polygon Shading Method. The shading applied to each polygon, defined by its four surrounding elevations, can be either constant over the entire cell or interpolated. ... The Gouraud method of interpolation is used: the shade values are computed at each elevation point, coinciding with each polygon vertex.

9. What are the various types of polygon shading?

There are many different types of shading algorithm, the most well known being flat (or constant) shading, Gouraud shading and Phong shading. Of these, the first two are used in games consoles. Flat shading is the least realistic of all shading methods.

10. What is halftone ?

Halftone is a graphic design technique used to reproduce an image by using dots of varying length with one or more colors. It enables image display similar to a continuous tone-like image but on a pixelated or halftone background.

11. Define Dithering?

Dithering is used in computer graphics to create the illusion of "color depth" in images with a limited color palette – a technique also known as color quantization. In a dithered image, colors that are not available in the palette are approximated by a diffusion of colored pixels from within the available palette.

12. Define Lighting and shading?

- Lighting - computing the luminous intensity for a specified 3D point, given a viewpoint
- Shading - assigning colors to pixels

13. What are illumination models?

Illumination Models:

- Empirical - approximations to observed light properties
- Physically based - applying physics properties of light and its interactions with matter

UNIT – V

1. Define computer graphics animation?

Computer graphics animation is the use of computer graphics equipment where the graphics output presentation dynamically changes in real time. This is often also called real time animation.

2. What is tweening?

It is the process, which is applicable to animation objects defined by a sequence of points, and that change shape from frame to frame.

3. Define frame?

One of the shape photographs that a film or video is made of is known as frame.

4. What is key frame?

One of the shape photographs that a film or video is made of the shape of an object is known initially and for a small no of other frames called keyframe.

5. What is pseudo animation?

Pseudo animation is creating a sequence of stills, photographing or video graphing each still as one frame, and then later playing back the frames at a faster speed.

6. What is the normal speed of a visual animation?

Visual animation requires a playback of at least 25 frames per second.

7. What are the different tricks used in computer graphics animation?

a. Color look Up Table manipulation
b. Bit plane manipulation

- c. Use of UDCS
- d. Special drawing modes e. Sprites
- f. Bit blitting

8. What is color look up table?

In color display unit it is necessary to read 44-bit for each pixel from buffer. This very time consuming process. To avoid this video controller uses look up table to store many entries of pixel vales in RGB format. This look up table is commonly known as color look up table.

9. What is solid modeling?

The construction of 3 dimensional objects for graphics display is often referred to as solid modeling.

10. What is an intuitive interface?

The intuitive interface is one, which simulates the way a person would perform a corresponding operation on real object rather than have menu command.

11. What is Sprite?

A Sprite is graphics shape in animation and games programs. Each sprite provided in the system has its own memory area similar to but smaller than pixel RAM.

12. What is the UDC technique?

UDC stands for User Defined Character set. It is graphics animation trick, which is used in early microcomputer system.

13. What is the use of hidden line removing algorithm?

The hidden line removal algorithm determines the lines, edges, surfaces or volumes that are visible or invisible to an observer located at a specific point in space.

14. What is computer graphics realism?

The creation of realistic picture in computer graphics is known as realism. It is important in fields such as simulation, design, entertainments, advertising, research, education, command, and control.

15. How realistic pictures are created in computer graphics?

To create a realistic picture, it must be process the scene or picture through viewing- coordinate transformations and projection that transform three-dimensional viewing coordinates onto two-dimensional device coordinates.

16. What is Fractals?

A Fractal is an object whose shape is irregular at all scales.

17. What is a Fractal Dimension?

Fractal has infinite detail and fractal dimension. A fractal imbedded in n-dimensional space could have any fractional dimension between 0 and n.

The Fractal Dimension $D = \frac{\log N}{\log S}$

Where N is the No of Pieces and S is the Scaling Factor.

18. What is random fractal?

The patterns in the random fractals are no longer perfect and the random defects at all scale.

19. What is geometric fractal?

A geometric fractal is a fractal that repeats self-similar patterns over all scales.

20. What is Koch curve?

The Koch curve can be drawn by dividing line into 4 equal segments with scaling factor $1/3$. and middle 2 segments are so adjusted that they form adjustment sides of an equilateral triangle.

21. What is turtle graphics program?

The turtle program is a Robot that can move in 2 dimensions and it has a pencil for drawing. The turtle is defined by the following parameters.

- a) Position of the turtle (x, y)
- b) Heading of the turtle θ the angle from the x axis.

22. What is graftals?

Graftals are applicable to represent realistic rendering plants and trees. A tree is represented by a String of symbols 0, 1, [,]

23. What is a Particle system?

A particle system is a method for modeling natural objects, or other irregularly shaped objects, that exhibit "fluid-like" properties. Particle systems are suitable for realistic rendering of fuzzy objects, smoke, sea and grass.

24. Give some examples for computer graphics standards?

- CORE – The Core graphics standard
- GKS -- The Graphics Kernel system
- PHIGS – The Programmers Hierarchical Interactive Graphics System. GSX – The Graphics system extension
- NAPLPS – The North American presentation level protocol syntax.

16 Marks Question

UNIT I

1. Explain refresh cathode ray tube?

- A beam of electrons, emitted by an electron gun, passes through focusing and deflection systems that direct the beam toward specified positions on the phosphor coated area.
- The phosphor that emits a small spot of light at each position contacted by the electron beam
- One way to keep phosphor glowing is to redraw the picture repeatedly by quickly directing the electron beam back over the same points. This type of display is called refresh CRT
- Primary components of electron gun are:
 - 1.Heated metal cathode
Heat is supplied to the cathode by directing the beam through a coil of wire called the filament inside the cylindrical cathode structure.
 - 2.Control grid
Intensity of the electron beam is controlled by setting the voltage levels on the control grid, which is a metal cylinder that fits to the cathode.
- Different kinds of phosphorus are available
- Besides color the major difference between phosphors is their persistence
- The maximum number of points that can be displayed without overlap on a CRT is referred to as resolution.
- Another property is aspect ratio

2.Explain color CRT monitors?

- A CRT monitor displays color pictures by using a combination of phosphors that emit different colored light.
- By combining the emitted light from the different phosphors, range of colors can be generated
- Two techniques
 - . Beam penetration method
 - . Shadow mask method
 - . Display color pictures
 - . Beam penetration method
- Two layers of phosphors, usually red and green
 - .A beam of slow electrons excites the outer red layer
 - .Fast electron penetrates through the red layer and excites the inner green layer
 - . An intermediate beam speeds, combinations of red and green light.
- Shadow mask method
 - . They produce a wider range of colors
 - . Has three phosphor color dots at each pixel position
One emits red light, another emits green light, and the third emits a blue light
 - . The three beams are deflected and focused as a group on to the shadow mask, which contains a series of holes aligned with a phosphor dot patterns.
 - . When the three beams passes through a hole in the shadow mask, they activate a dot triangle, which appears as a small color spot on the screen.

3.Explain direct view storage tubes and liquid crystal displays?

Liquid crystal displays

- Refers to the compounds having crystalline arrangement of molecules flow of liquid

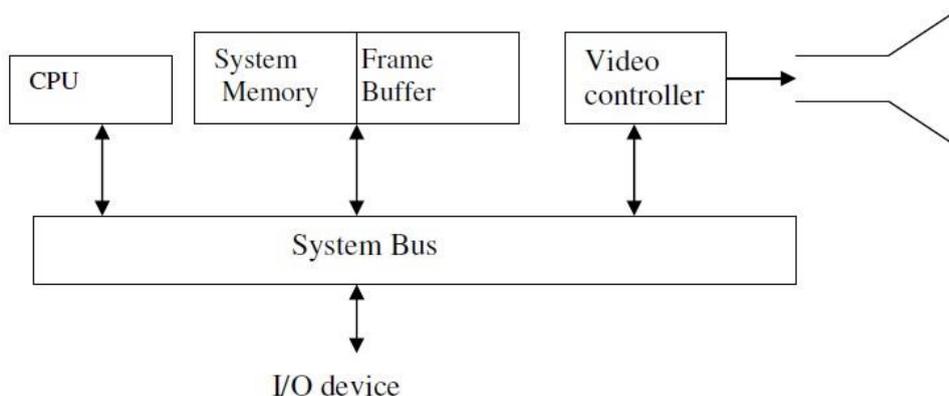
- Two plates each glass plate contains a light polarized that are right angles to each other
- Two types
 - Passive matrix LCD
 - Active matrix LCD

Direct view storage tubes

- Alternative method for maintaining a screen image
- Stores picture information as a charge distribution
- Very complex pictures can be displayed at very high resolutions
- To eliminate the picture section the entire screen must be erased
- The erasing and redrawing process can take several seconds for a picture

4.Explain Raster scan systems?

- Several processing units
- Contains a special purpose processor, called video controller or display controller
- Video controller
 - A fixed area of the system memory is reserved for the frame buffer, and the video controller is given access to the frame-buffer memory
 - Two registers are used to store the coordinates of the screen pixels
 - The value is stored in the frame buffer for this pixel position is then retrieved and used to set the intensity of the CRT beam
 - Initially the x register is set to 0 and the y register is set to ymax
 - Then the x register is incremented by 1, and the process repeated for the next pixel on the top scan line
 - This process is repeated for each pixel along the scan line
 - After the last pixels on the top scan line has been processed, the x register is reset to 0 and the yregister is decremented by 1.
 - The procedure is repeated fro each successive line
- Frame buffer locations and the corresponding screen positions are referenced as Cartesian coordinates



5. Explain the following?

1. Z-mouse
2. Joysticks
3. Touch panels
4. Image scanners
5. Data glove

Z-Mouse:

- . Include three buttons
- . A thumb wheel on the side, a track ball on the top, and a standard mouse ball underneath.
- . This design provides six degrees of freedom to select an object from the spatial position.
- . Wit this we can pickup an object, rotate it and we can move it in any direction
- . Used in virtual reality and CAD systems

Joysticks

- Consists of small vertical liver mounted on a base

- Used to move the cursor around the screen
- The screen cursor is moved according to the distance
- One or two buttons is usually intended for signaling certain actions

Touch panels

- Three types

- Optical touch panel
- Electrical touch panel
- Acoustical touch panel

• Allow selecting the screen position with the touch of finger. Touch input can be recorded

using optical, electrical methods

Image scanners

. Drawings, color and black and white photos or text can be given as an input to the computer with an optical scanning mechanism.

. According to reflected light intensity the gradations of gray scale or color can be stored in an array

Data glove

. Constructed with a series of sensors that can detect hand and finger motions

. The transmitting and receiving antennas can be structured as a set of three mutually perpendicular coils, forming a three dimensional Cartesian coordinates system.

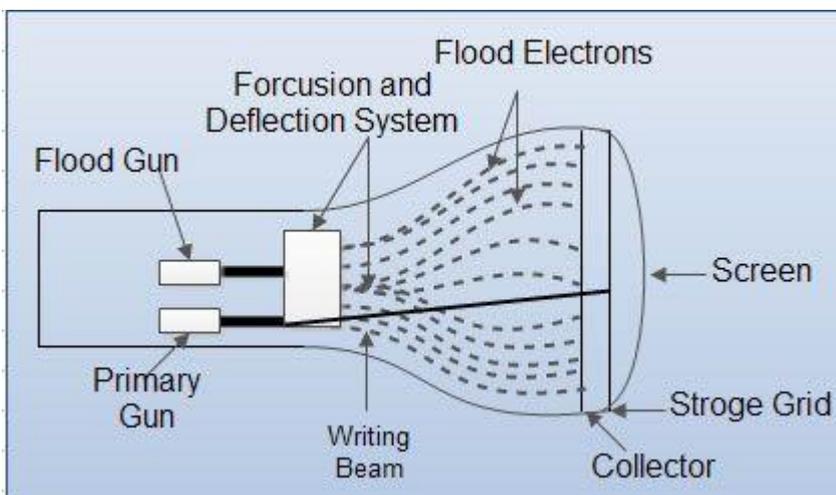
. Electromagnetic coupling between the three pairs of coil is used to provide information about the position and orientation of hand.

6.Explain about DVST?

- Direct View Storage Tubes / Devices. So these are also called DVST.
- DVST stores the picture information as charge distribution just behind the computer screen.
- In this **2 electron guns** are used :
- **Primary** It is responsible to store the picture information in form of charge.

Secondary It was responsible to display the picture on the Screen.

It is similar to CRT as far as the electronic gun and phosphor-coated mechanisms are concerned. But instead of the electron beam directly writing the pictures on the phosphor coated CRT screen, the writing is done with the help of a fine-mesh wire grid.



7. Explain in detail about the DDA scan conversion algorithm?

The digital differential analyzer is a scan conversion algorithm based on calculation either Δy or Δx using the following equations

$$\Delta y = m \Delta x$$

$$\Delta x = \Delta y / m$$

Sample the line at unit intervals in one coordinate and determine corresponding integer values nearest the line path for the coordinates

Sample at X intervals ($\Delta x = 1$) and compute each successive Y value as $Y_{k+1} = Y_k + m$

For lines with positive slope greater than 1, reverse the roles of X and Y. Sample at unit Y intervals ($\Delta y = 1$) and calculate each successive X value as $X_{k+1} = X_k + 1/m$

Algorithm

- Step 1: Input the line endpoints and store the left endpoint in (x_1, y_1) and right endpoint in (x_2, y_2)
- Step 2: Calculate the values of Δx and Δy using $\Delta x = x_b - x_a$, $\Delta y = y_b - y_a$
- Step 3: if the values of $\Delta x > \Delta y$ assign values of steps as Δx otherwise the values of steps as Δy
- Step 4: Calculate the values of X increment and Y increment and assign the value $x = x_a$ and $y = y_a$
- Step 5: for $k=1$ to steps do
 - $X = X + X \text{ increment}$
 - $Y = Y + Y \text{ increment}$
 - Putpixel(ceil(x), ceil(y), 15)
- Step 6: End

8. Explain Bresenham's line drawing algorithm?

In Bresenham's approach the pixel position along a line path are determined by sampling unit X intervals. Starting from the left end point (X_0, Y_0) of a given line we step to each successive columns and plot the pixel whose scan line Y-value is closest to the line path. Assuming the Kth step in process, determined that the pixel at (X_k, Y_k) decide which pixel to plot in column X_{k+1} . The choices are (X_{k+1}, Y_k) and (X_{k+1}, Y_{k+1})

Algorithm

Step 1: Input the line endpoints and store the left endpoint in (X_0, Y_0)

Step 2: Load (X_0, Y_0) in to the frame buffer

Step 3: Calculate constants Δx , Δy , $2 \Delta y$, $-2 \Delta x$, and obtain the decision parameters as

$$P_0 = 2 \Delta y - \Delta x$$

Step 4 : At each X_k along the line, starting at $k = 0$, perform the following test

If $P_k < 0$, the next point to plot is (X_{k+1}, Y_k) and

$$P_{k+1} = P_k + 2 \Delta y$$

Otherwise, the next point to plot is (X_{k+1}, Y_{k+1}) and

$$P_{k+1} = P_k + 2 \Delta y - 2 \Delta x$$

Step 5: Repeat step 4 Δx times

9. Explain Midpoint Circle algorithm?

Algorithm

Step 1: Input radius r and circle center (X_c, Y_c) and obtain the first point on

the circumference of a circle centered on the origin as

$$(X_0, Y_0) = (0, r)$$

Step 2: Calculate the initial values of the decision parameter as

$$P_0 = 5/4 - r$$

Step 3: At each position starting at k perform the following test:

If $P_k < 0$, the next point to plot is (X_{k+1}, Y_k) and

$$P_{k+1} = P_k + 2 X_{k+1} + 1$$

Otherwise the next point is (X_{k+1}, Y_{k+1}) and

$$P_{k+1} = P_k + 2 X_{k+1} + 1 - 2 Y_{k+1}$$

Step 4: Determine symmetry points in the other seven octants

Step 5: Move each pixel position (X, Y) onto the circular path centred on

(X_c, Y_c) and plot the coordinate values as

$$X = X + X_c \quad Y = Y + Y_c$$

Step 6: Repeat steps 3 through until $X \geq Y$

$$P_k + 1 = P_k + 2 \Delta Y$$

Other wise, the next point is (X_{k+1}, Y_{k+1}) and

$$P_k + 1 = P_k + 2 \Delta Y - 2 \Delta X$$

Step 5: Repeat steps 4 ΔX times

10. Explain Ellipse generating Algorithm?

Algorithm

Step 1: Input radius r_x , r_y and ellipse center (X_c, Y_c) and obtain the first point on the circumference of a circle centered on the origin as

$$(X_0, Y_0) = (0, r_y)$$

Step 2: Calculate the initial values of the decision parameter in region 1 as

$$P1_0 = r_y^2 - r_x^2 r_y + 1/4 r_x^2$$

Step 3: At each position starting at X_k position in region 1, starting at $k = 0$, perform the following test:

If $P_k < 0$, the next point to plot is (X_{k+1}, Y_k) and

$$P1_{k+1} = P1_k + 2 r_y^2 X_{k+1} + r_y^2$$

Otherwise the next point is (X_{k+1}, Y_{k-1}) and

$$P1_{k+1} = P1_k + 2 r_y^2 X_{k+1} - 2 r_y^2 Y_{k+1} + r_y^2$$

Step 4: Calculate the initial values of the decision parameter in region 2 as

$$P2_0 = r_y^2 (X_0 + 1/2)^2 + r_x^2 (Y_0 - 1)^2 - r_x^2 r_y^2$$

Step 5: At each position starting at Y_k position in region 2, starting at $k = 0$, perform the following test:

If $P_k > 0$, the next point to plot is (X_k, Y_{k-1}) and

$$P2_{k+1} = P2_k - 2 r_y^2 Y_{k+1} + r_x^2$$

Otherwise the next point is (X_{k+1}, Y_{k-1}) and

$$P2_{k+1} = P2_k - 2 r_y^2 Y_{k+1} - 2 r_x^2 Y_{k+1} + r_x^2$$

Step 6: Determine symmetry points in the other three octants

Step 7: Move each pixel position (X, Y) onto the circular path centred on (X_c, Y_c) and plot the coordinate values as

$$X = X + X_c \quad Y = Y + Y_c$$

Step 8: Repeat steps for region 1 until $2 r_y^2 X \geq 2 r_y^2 Y$

11.Explain Boundary fill algorithm?

- . If the boundary is specified in a single color, the fill algorithm proceeds outward pixel-by-pixel until the boundary color is encountered. This is called boundary fill algorithm
- . The boundary fill procedure accepts as input the coordinates of an interior point (x, y), a fill color, and a boundary color.
- . Starting from (x, y), the procedure tests the neighboring positions to determine whether they are boundary color.
- . If not, they are painted with the fill color, and the neighbors are tested.
- . This process continues until all pixels up to the boundary color for the area have been tested
- . Two methods
 - 4- connected
 - 4 neighbouring points are connected
 - 8-connected
 - correctly fill the interior of the area defined

program

```
void boundaryfill( int x, int y, int fill, int boundary)
{
int current;

current = getpixel (x, y);
if ((current != boundary) &&(current != nil)
{
setColor ( fill);
setPixel ( x, y);
boundaryfill (x+1, y, fill, boundary);
boundaryfill (x-1, y, fill, boundary);
boundaryfill (x, y+1, fill, boundary);
boundaryfill (x, y-1, fill, boundary);
}
}
```

UNIT-II

1. Explain reflection and shear?

Reflection - It is a mirror image of an object

- Rotating about 180 degree
- Flat object moving in the xy plane
- Reflection about y axis flips x coordinates
- Reflection point as pivot point is same as above
- To obtain the transformation matrix for reflection diagonal is $y = -x$
- Sequence
- Clockwise rotation by 45 degree
- Reflection about y axis
- Counter wise rotation by 45 degree

Shear - Internal layer cause to slide over each other called shear

- Transforms coordinate position as $X'' = x + shx$, $Y'' = y$
- Shift in the position of objects relative to shearing reference lines are equivalent to translations

2. Explain Liang Barsky line clipping

- Faster line clipper of the parametric equation of a line segment
- Line parallel to one of the clipping boundaries
- Line intersects the extension of boundary k
- If $u_1 > u_2$ line is outside the clipping window
- Else inside the clipping window
- Clipping is done using the reflection in the clip window

3. Explain Sutherland Hodgeman polygon clipping

- Clipping polygon which lies inside the clipping window
- Four possible cases
- If the first vertex is outside the window boundary and the second vertex inside
- If the first vertex is inside the window boundary and the second vertex outside
- If both are outside
- If both are inside
- Repeat the process of algorithm
- Convex polygon are correctly clipped using this clipping
- Concave and convex polygon are also used

4. Explain about clipping operations

- Clip a picture from either outside or inside a region known as clipping
- Also called as clipping algorithm
- The region against the object is known as clip window
- Clipping operations on different types of objects
- Point clipping
- Polygon clipping
- Area clipping
- Line clipping
- Curve clipping
- Text clipping
- Polygon and line clipping are the standard clipping components

5. Explain the viewing pipeline

A world coordinate area selected for display is called a window. An area on a display device to which a window is mapped is called a view port. The window defines what is to be viewed the view port defines where it is to be displayed. The mapping of a part of a world coordinate scene to device coordinate is referred to as viewing transformation. The two dimensional viewing transformation is referred to as window to view port transformation of windowing transformation.

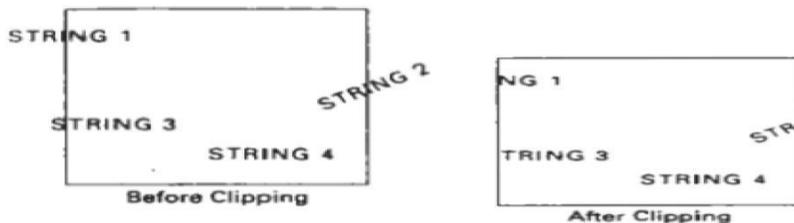
The viewing transformation in several steps, as indicated in Fig. First, we construct the scene in world coordinates using the output primitives. Next to obtain a particular orientation for the window, we can set up a two-dimensional viewing-coordinate system in the world coordinate plane, and define a window in the viewing-coordinate system. The viewing-coordinate reference frame is used to provide a method for setting up arbitrary orientations for rectangular windows. Once the viewing reference frame is established, we can transform descriptions in world coordinates to viewing coordinates.

We then define a viewport in normalized coordinates (in the range from 0 to 1) and map the viewing-coordinate description of the scene to normalized coordinates. At the final step all parts of the picture that lie outside the viewport are clipped, and the contents of the viewport are transferred to device coordinates.

6. Discuss Text Clipping?

TEXT CLIPPING

An alternative to rejecting an entire character string that overlaps a window boundary is to use the **all-or-none character-clipping** strategy. Here we discard only those characters that are not completely inside the window. In this case, the boundary limits of individual characters are compared to the window. Any character that either overlaps or is outside a window boundary is clipped.



Exterior clipping:

Procedure for clipping a picture to the interior of a region by eliminating everything outside the clipping region. By these procedures the inside region of the picture is saved. To clip a picture to the exterior of a specified region. The picture parts to be saved are those that are outside the region. This is called as exterior clipping.

UNIT III

1. Explain the three dimensional display methods?

Parallel projection

Parallel projection is a method for generating a view of a solid object is to project points on the object surface along parallel lines onto the display plane.

In parallel projection, parallel lines in the world coordinate scene project into parallel lines on the two dimensional display planes.

This technique is used in engineering and architectural drawings to represent an object with a set of views that maintain relative proportions of the object.

The appearance of the solid object can be reconstructed from the major views.



- The production of the 2D display of the 3D scene is called projection
- Project points on the object surface along the parallel lines on to the display plane
- Different 2D views of objects can be produced by projecting the visible points

Perspective projection

It is a method for generating a view of a three dimensional scene is to project points to the display plane along converging paths.

This makes objects further from the viewing position be displayed smaller than objects of the same size that are nearer to the viewing position.

In a perspective projection, parallel lines in a scene that are not parallel to the display plane are projected into converging lines.

Scenes displayed using perspective projections appear more realistic, since this is the way that our eyes and a camera lens form images.

- Done by the projecting points to the display plane along the converging points
- Causes the objects farther from the viewing point should be smaller of the same sized object present here.
- Depth CUEING
- Basic problem for visualization techniques is called depth cueing
- Some 3D objects are without depth information
- Visible line and surface identification
- To highlight the visible lines
- Display visible lines as dashed lines
- Removing the invisible lines
- Surface rendering
- Lightening conditions in the screen
- Assigned characteristics
- Degree of transparency
- How rough or smooth the surfaces are to be
- Exploded and cutaway views
- Three dimensional and stereoscopic views

2. Explain spline representation

- It is referred to a curve drawn in a different manner
- Interpolation and approximation splines
- Set of coordinate points called control points
- Curve can be translated ,rotated and scaled
- Enclosing a set of points called convex hull
- Set of connected points is often called control graph
- Parametric continuity condition
- Geometric continuity condition
- Spline specification

3. Explain Bezier curves and surfaces

- Have number of properties
- Can be fitted to any number of control points
- Polynomial functions between p_0 and p_n

$$n P(u) = \sum_{k=0}^n p_k \text{ BEZ } k,n(u)$$

$K = 0$

- Calculated $x(u), y(u), z(u)$
- Properties of Bezier curves
- Cubic Bezier curves
- Design techniques in Bezier curves
- Bezier surfaces

4. Explain general three dimensional rotations

- Transformation sequences
- $P'' = T'' \cdot R_x(\text{_____}) \cdot T \cdot P$
- Rotation in five steps
- Translate the object that rotates in parallel coordinate axis
- Rotate the object with one coordinate axis
- Apply inverse rotation to its original position
- Apply inverse translation to its original position
- $V = p_2 - p_1$
- After rotation to original position
- $R() \text{_____} = T'' \cdot M \cdot T$

5. Discuss Polygon Meshes

A single plane surface can be specified with a function such as **fillArea**. But when object surfaces are to be tiled, it is more convenient to specify the surface facets with a mesh function.

One type of polygon mesh is the **triangle strip**. A triangle strip formed with 11 triangles connecting 13 vertices.

This function produces $n-2$ connected triangles given the coordinates for n vertices.

6. Enumerate Composite Transformation

Composite three dimensional transformations can be formed by multiplying the matrix representation for the individual operations in the transformation sequence.

This concatenation is carried out from right to left, where the right most matrixes is the first transformation to be applied to an object and the left most matrix is the last transformation.

A sequence of basic, three-dimensional geometric transformations is combined to produce a single composite transformation which can be applied to the coordinate definition of an object.

7. SCAN-LINE METHOD

This image-space method for removing hidden surfaces is an extension of the scan-line algorithm for filling polygon interiors. As each scan line is processed, all polygon surfaces intersecting that line are examined to determine which are visible. Across each scan line, depth calculations are made for each overlapping surface to determine which is nearest to the view plane. When the visible surface has been determined, the intensity value for that position is entered into the refresh buffer.

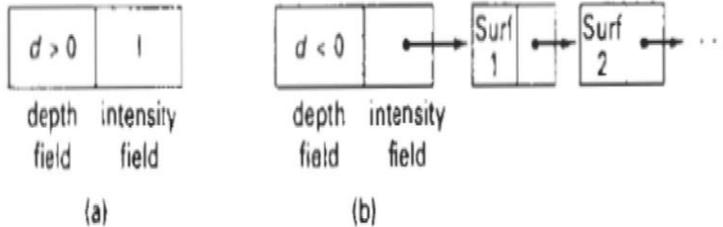
We assume that tables are set up for the various surfaces, which include both an edge table and a polygon table. The **edge table** contains coordinate endpoints for each line in-the scene, the inverse slope of each line, and pointers into the polygon table to identify the surfaces bounded by each line. The **polygon table** contains coefficients of the plane equation for each surface, intensity information for the surfaces, and possibly pointers into the edge table.

To facilitate the search for surfaces crossing a given scan line, we can set up an active list of edges from information in the edge table. This active list will contain only edges that cross the current scan line, sorted in order of increasing x . In addition, we define a flag for each surface that is set on or off to indicate whether a position along a scan line is inside or outside of the surface. Scan lines are processed from left to right. At the leftmost boundary of a surface, the surface flag is turned on; and at the rightmost boundary, it is turned off.

8.A- BUFFER METHOD

An extension of the ideas in the depth-buffer method is the A-buffer method. The A buffer method represents an **antialiased, area-averaged, accumulation-buffer method** developed by Lucasfilm for implementation in the surface-rendering system called **REYES** (an acronym for "Renders

Everything You Ever Saw"). A drawback of the depth-buffer method is that it can only find one visible surface at each pixel position. The A-buffer method expands



9. What are Three Dimensional Transformation Functions? Discuss

Some of the basic 3D transformation functions are: `translate (translateVector, matrixTranslate)` `rotateX(thetaX, xMatrixRotate)` `rotateY(thetaY, yMatrixRotate)` `rotateZ(thetaZ, zMatrixRotate)` `scale3 (scaleVector, matrixScale)` Each of these functions produces a 4 by 4 transformation matrix that can be used to transform coordinate positions expressed as homogeneous column vectors.

Parameter `translate Vector` is a pointer to list of translation distances `tx`, `ty`, and `tz`.

Parameter `scale vector` specifies the three scaling parameters `sx`, `sy` and `sz`.

`Rotate` and `scale` matrices transform objects with respect to the coordinate origin.

Composite transformation can be constructed with the following functions:

`composeMatrix3` `buildTransformationMatrix3` `composeTransformationMatrix3` The order of the transformation sequence for

the **`buildTransformationMarix3`** and **`composeTransformationMarix3`** functions, is the same as in 2 dimensions:

1. `scale`

2. `rotate`

3. `translate`

Once a transformation matrix is specified, the matrix can be applied to specified points with

`transformPoint3 (inPoint, matrix, outpoint)`

The transformations for hierarchical construction can be set using structures with the function

`setLocalTransformation3 (matrix, type)` where parameter `matrix` specifies the elements of a 4 by 4 transformation matrix and parameter `type` can be assigned one of the values of: `Preconcatenate`, `Postconcatenate`, or `replace`.

UNIT-IV

1.Explain about RGB color model?

Based on the tristimulus theory of our eyes perceive color through the stimulation of three visual pigments in the cones on the retina. These visual pigments have a peak sensitivity at wavelengths of about 630 nm (red), 530 nm (green) and 450 nm (blue).

By comparing intensities in a light source, we perceive the color of the light. This is the basis for displaying color output on a video monitor using the 3 color primaries, red, green, and blue referred to as the RGB color model. The origin represents black, and the vertex with coordinates (1,1,1) is white.

Vertices of the cube on the axes represent the primary colors, the remaining vertices represent the complementary color for each of the primary colors.

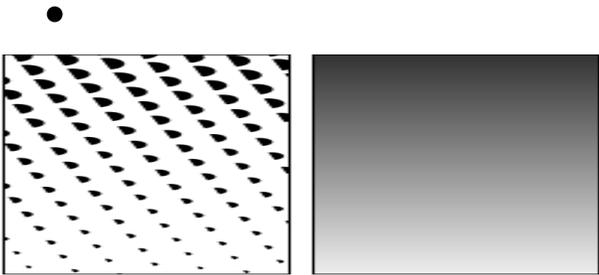
The RGB color scheme is an additive model. (i.e.,) Intensities of the primary colors are added to produce other colors. Each color point within the bounds of the cube can be represented as the triple (R,G,B) where values for R, G and B are assigned in the range from 0 to 1.

The color C_λ is expressed in RGB component as
 $C_\lambda = R\mathbf{R} + G\mathbf{G} + B\mathbf{B}$

2.Explain Halftone

- **Halftone** is the reprographic technique that simulates continuous tone imagery through the use of dots, varying either in size, in shape or in spacing, thus generating a gradient-like effect.
- "Halftone" can also be used to refer specifically to the image that is produced by this process.
- Where continuous tone imagery contains an infinite range of colors or greys, this process reduces visual reproductions to an image that is printed with only one color of ink, in dots of differing size or spacing.

- This reproduction relies on a basic optical illusion: the tiny halftone dots are blended into smooth tones by the human eye.
- At a microscopic level, developed black-and-white photographic film also consists of only two colors, and not an infinite range of continuous tones.



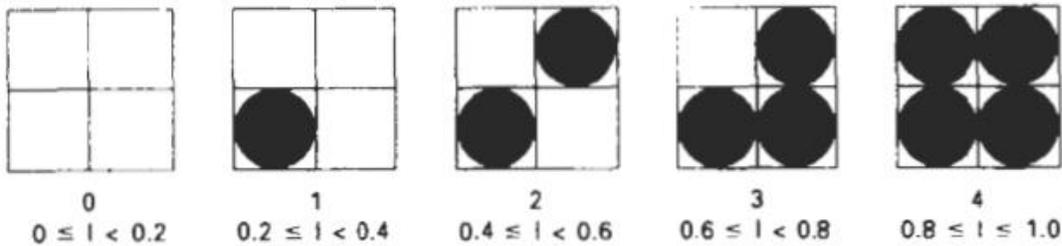
- For a black-and-white photograph, each intensity area is reproduced as a series of black circles on a white background.
- The diameter of each circle is proportional to the darkness required for that intensity region.
- Darker regions are printed with larger circles.
- lighter regions are printed with smaller circles

Halftone Dot Shape:

- There are different dot types available, each of them having their own characteristics. They can be used simultaneously to avoid the moiré effect.
- **Round dots:** most common, suitable for light images, especially for skin tones. They meet at a tonal value of 70%.
- **Elliptical dots:** appropriate for images with many objects. Elliptical dots meet at the tonal values 40% (pointed ends) and 60% (long side), so there is a risk of a pattern.
- **Square dots:** best for detailed images, not recommended for skin tones. The corners meet at a tonal value of 50%. The transition between the square dots can sometimes be visible to the human eye.

Halftone Approximations:

- halftone reproductions are approximated using rectangular pixel regions, called **halftone patterns** or **pixel patterns**.
- The number of intensity levels that we can display with this method depends on how many pixels we include in the rectangular grids and how many levels a system can display



3. What are the properties of light ? explain

Light is a narrow frequency band within the electromagnetic system. Other frequency bands within this spectrum are called radio waves, micro waves, infrared waves and x-rays. The below fig shows the frequency ranges for some of the electromagnetic bands. Each frequency value within the visible band corresponds to a distinct color. At the low frequency end is a red color (4.3×10^{14} Hz) and the highest frequency is a violet color (7.5×10^{14} Hz)

Spectral colors range from the reds through orange and yellow at the low frequency end to greens, blues and violet at the high end.

Since light is an electro magnetic wave, the various colors are described in terms of either the frequency for the wave length λ of the wave.

The wave length and frequency of the monochromatic wave are inversely proportional to each other, with the proportionality constants as the speed of light C where $C = \lambda f$

A light source such as applications:

1. Implement a color models.

2. Implement a realistic scenes and objects

3. OPENGL is a easy way to make a real objects. as the sun or a light bulb emits all frequencies within the visible range to produce white light. When white light is incident upon an object, some frequencies are reflected and some are absorbed by the object. The combination of frequencies present in the reflected light determines what we perceive as the color of the object.

If low frequencies are predominant in the reflected light, the object is described as red. In this case, the perceived light has the dominant frequency at the red end of the spectrum. The dominant frequency is also called the hue, or simply the color of the light.

Brightness is another property, which in the perceived intensity of the light.

Intensity in the radiant energy emitted per limit time, per unit solid angle, and per unit projected area of the source.

Radiant energy is related to the luminance of the source.

The next property in the purity or saturation of the light.

- Purity describes how washed out or how pure the color of the light appears.

- Pastels and Pale colors are described as less pure.

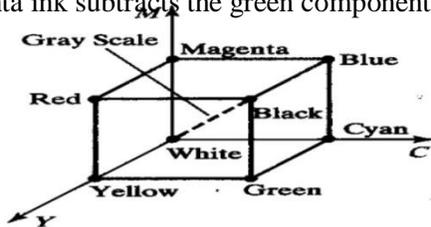
The term chromaticity is used to refer collectively to the two properties, purity and dominant frequency.

4.Explain cmy colour model

A color model defined with the primary colors cyan, magenta, and yellow (CMY) is useful for describing color output to hard copy devices.

It is a subtractive color model (i.e.,) cyan can be formed by adding green and blue light. When white light is reflected from cyan-colored ink, the reflected light must have no red component. i.e., red light is absorbed or subtracted by the ink.

Magenta ink subtracts the green component from incident light and yellow subtracts the blue component.



In CMY model, point (1,1,1) represents black because all components of the incident light are subtracted.

The origin represents white light.

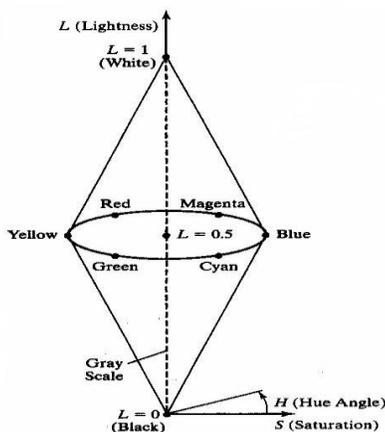
Equal amounts of each of the primary colors produce grays along the main diagonal of the cube.

A combination of cyan and magenta ink produces blue light because the red and green components of the incident light are absorbed.

The printing process often used with the CMY model generates a color point with a collection of 4 ink dots; one dot is used for each of the primary colors (cyan, magenta and yellow) and one dot in black.

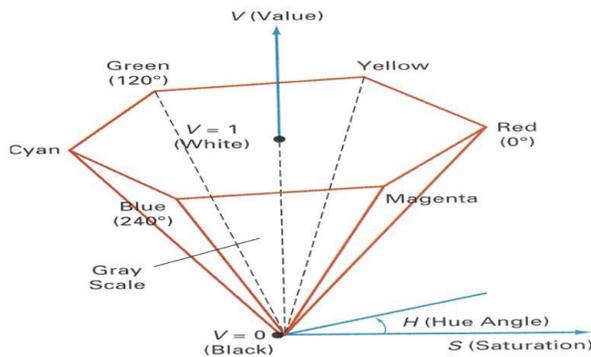
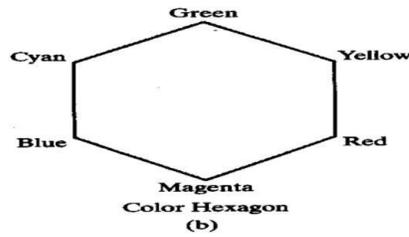
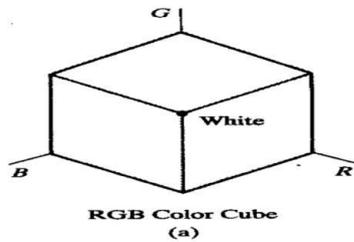
5.Explain about HLS colour model

HLS model is based on intuitive color parameters used by Tektronix. It has the double cone representation shown in the below figure. The 3 parameters in this model are called Hue (H), lightness (L) and saturation (s).



6.Discuss about HSV colour model

The HSV model uses color descriptions that have a more interactive appeal to a user. Color parameters in this model are hue (H), saturation (S), and value (V). The 3D representation of the HSV model is derived from the RGB cube. The outline of the cube has the hexagon shape.



7. Explain Dithering technique.

- Dithering is used in computer graphics to create the illusion of "color depth" in images with a limited color palette - a technique also known as color quantization.
- In a dithered image, colors that are not available in the palette are approximated by a diffusion of colored pixels from within the available palette.

Dithering technique:

- The human eye perceives the diffusion as a mixture of the colors within it.
- Dithered images, particularly those with relatively few colors, can often be distinguished by a characteristic graininess or speckled appearance.

Various Dithering Methods

- Thresholding (or) Average Dithering
- Random dithering
- Patterning dithers using a fixed pattern
- Ordered Dithering
 - Bayer Ordered Dithering
 - Void-and-Cluster Dithering
- Error-Diffusion Dithering

It refers to techniques for approximating halftones without reducing resolution, as pixel-grid patterns do. The term **dithering** is also applied to halftone approximation method using pixel grid, and something it is used to refer to color halftone approximations only. Random values added to pixel intensities to break up contours are often referred to as dither noise. Number of methods is used to generate intensity variations.

Ordered dither methods generate intensity variations with a one-to-one mapping of points in a scene to the display pixel.

Various algorithms have been used to generate the random distributions.

The effect is to add noise over an entire picture, which tends to soften intensity boundaries.

- Each pixel produces a *quantization error*

- The quality of the result may be improved by adjusting the threshold locally, so that adjacent pixels in small areas are quantized with different thresholds.
- This reduces the **average** local quantization error. Matrices of these threshold are called **dither** matrices.
- Dither is an intentionally applied form of noise used to randomize quantization error, preventing large-scale patterns such as color banding in images.
- A typical use of dither is converting a greyscale image to black and white, such that the density of black dots in the new image approximates the average grey level in the original.

7.Explain about YIQ colour model?

The National Television System Committee (NTSC) color model for forming the composite video signal in the YIQ model.

In the YIQ color model, luminance (brightness) information is contained in the Y parameter, chromaticity information (hue and purity) is contained into the I and Q parameters.

A combination of red, green and blue intensities are chosen for the Y parameter to yield the standard luminosity curve.

Since Y contains the luminance information, black and white TV monitors use only the Y signal.

Parameter I contains orange-cyan hue information that provides the flesh-tone shading and occupies a bandwidth of 1.5 MHz.

Parameter Q carries green-magenta hue information in a bandwidth of about 0.6 MHz.

An RGB signal can be converted to a TV signal

UNIT V

1. Explain key frame systems

Key frame systems

- Motion path can be given with a kinematics description
- Physically based on force acting object
- Frame in to individual component or object called cells

Morphing

- Transformation of object shapes from one form to another is called morphing
- Two key frames for an object transformation
- Preprocessing using vertex count Simulating accelerations
- Time interval between keyframe is divided into $n + 1$

$$t = (t_2 - t_1) / n + 1$$

$$t_{Bj} = t_1 + jt, j = 1, 2, 3, \dots$$

- Time for j th in-between is

$$t_{Bj} = t_1 + t[(1 - \cos[j / (n + 1)]) / 2], j = 1, 2, 3, \dots$$

2.Explain about animation

Computer animation refers to any time sequence of visual changes in a scene.

Computer animations can also be generated by changing camera parameters such as position, orientation and focal length.

Applications of computer-generated animation are entertainment, advertising, training and education.

Example : Advertising animations often transition one object shape into another. **Frame-by-Frame animation** Each frame of the scene is separately generated and stored. Later, the frames can be recoded on film or they can be consecutively displayed in "real-time playback" mode **Design of Animation Sequences** An animation sequence is designed with the following steps: Story board layout

Object definitions

Key-frame specifications

Generation of in-between frames.

Story board

The story board is an outline of the action.

It defines the motion sequences as a set of basic events that are to take place.

Depending on the type of animation to be produced, the story board could consist of a set of rough sketches or a list of the basic ideas for the motion.

Object Definition

An object definition is given for each participant in the action.

Objects can be defined in terms of basic shapes such as polygons or splines.

The associated movements of each object are specified along with the shape.

KEY FRAME

A key frame is detailed drawing of the scene at a certain time in the animation sequence.

Within each key frame, each object is positioned according to the time for that frame.

Some key frames are chosen at extreme positions in the action; others are spaced so that the time interval between key frames is not too much.

Keyframe Systems

Each set of in-betweens are generated from the specification of two keyframes.

For complex scenes, we can separate the frames into individual components or objects called cells, an acronym from cartoon animation.

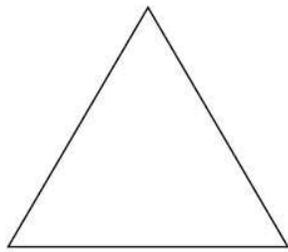
3. Write notes on fractal geometry

Fractals are very complex pictures generated by a computer from a single formula. They are created using iterations. This means one formula is repeated with slightly different values over and over again, taking into account the results from the previous iteration.

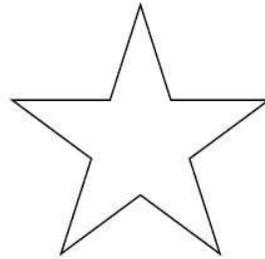
Fractals are used in many areas such as –

- **Astronomy** – For analyzing galaxies, rings of Saturn, etc.
- **Biology/Chemistry** – For depicting bacteria cultures, Chemical reactions, human anatomy, molecules, plants,

- **Others** – For depicting clouds, coastline and borderlines, data compression, diffusion, economy, fractal art, fractal music, landscapes, special effect, etc.
- Geometric fractals deal with shapes found in nature that have non-integer or fractal dimensions. To geometrically construct a deterministic (nonrandom) self-similar fractal, we start with a given geometric shape, called the **initiator**. Subparts of the initiator are then replaced with a pattern, called the **generator**.



Initiator



Generator

4. Discuss Koch curve

The **Koch snowflake** (also known as the **Koch curve**, **Koch star**, or **Koch island**) is a mathematical curve and one of the earliest fractal curves to have been described. It is based on the Koch curve, which appeared in a 1904 paper titled "On a continuous curve without tangents, constructible from elementary geometry."

The progression for the area of the snowflake converges to $\frac{8}{5}$ times the area of the original triangle, while the progression for the snowflake's perimeter diverges to infinity. Consequently, the snowflake has a finite area bounded by an infinitely long line.

Construction:

The Koch snowflake can be constructed by starting with an equilateral triangle, then recursively altering each line segment as follows:

1. divide the line segment into three segments of equal length.
2. draw an equilateral triangle that has the middle segment from step 1 as its base and points outward.
3. remove the line segment that is the base of the triangle from step 2.

After one iteration of this process, the resulting shape is the outline of a hexagram.

The Koch snowflake is the limit approached as the above steps are followed over and over again. The Koch curve originally described by Helge von Koch is constructed with only one of the three sides of the original triangle. In other words, three Koch curves make a Koch snowflake.

5. Write notes on ray tracing

Ray Tracing is a global illumination based rendering method. It traces rays of light from the eye back through the image plane into the scene. Then the rays are tested against all objects in the scene to determine if they

intersect any objects. If the ray misses all objects, then that pixel is shaded the background color. Ray tracing handles shadows, multiple specular reflections, and texture mapping in a very easy straight-forward manner.

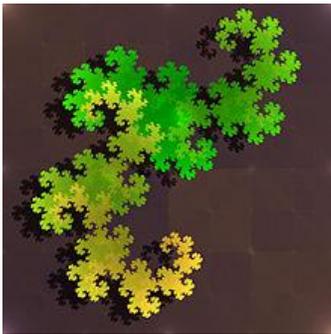
Note that ray tracing, like scan-line graphics, is a point sampling algorithm. We sample a continuous image in world coordinates by shooting one or more rays through each pixel. Like all point sampling algorithms, this leads to the potential problem of aliasing, which is manifested in computer graphics by jagged edges or other nasty visual artifacts.

In ray tracing, a ray of light is traced in a backwards direction. That is, we start from the eye or camera and trace the ray through a pixel in the image plane into the scene and determine what it hits. The pixel is then set to the color values returned by the ray.

6. Write notes on dragon curves

A **dragon curve** is any member of a family of self-similar fractal curves, which can be approximated by recursive methods such as Lindenmayer systems.

A dragon curve is a recursive nonintersecting curve whose name derives from its resemblance to a certain mythical creature.



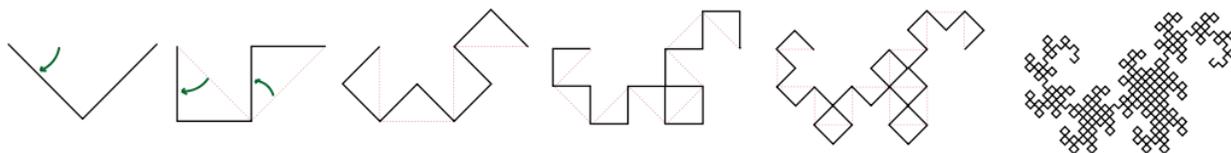
Construction

It can be written as a Lindenmayer system with

- angle 90°
- initial string FX
- string rewriting rules
 - $X \mapsto X+YF+$
 - $Y \mapsto -FX-Y.$

That can be described this way: starting from a base segment, replace each segment by 2 segments with a right angle and with a rotation of 45° alternatively to the right and to the left

The dragon curve can be drawn starting from a base segment, replace each segment by 2 segments with a right angle and with a rotation of 45 degree alternatively to the right and to the left as shown in the below figure.



V.S.B. ENGINEERING COLLEGE, KARUR
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
Academic Year 2018-19(ODD Semester)

Class: III Year/ V Semester B.E., CSE “A”Section
Name of the Subject: CS6504/Computer Graphics
Name of Faculty Member:M.Maharasi

ASSIGNMENT TOPICS

S.No	Assignment Topic
1.	Explain a method to rotate an object about an axis that is not parallel to the coordinate axis with neat block diagram and derive the transformation matrix for the same
2.	Solve using Liang Barsky line clipping algorithm where $(X_{wmin}, Y_{wmin})=(1,9)$ and $(X_{wmax}, Y_{wmax})=(2,8)$ for line segments P1(3,7) to P2(3,10), P3(6,6) to P4(8,9), P5(-1,7) to P6(11,1) and explain the algorithm
3.	Show how shear transformation may be expressed in terms of rotation and scaling
4.	Calculate the pixel position along the circle path with radius $r=14$ centered on the origin using Bresenham's circle drawing algorithm from point (0,4) to point $x=y$.
5.	Derive the ellipse drawing algorithm
6.	Explain the following: Colour selection, RGB and YIQ models
7.	Discuss about filled area primitives
8.	Give the single point perspective projection transformation matrix when projectors are placed on the z axis and also give the two point perspective transformation matrix
9.	Discuss the inverse transformation
10.	Explain about 3D transformations
11.	Explain the Illumination models.
12.	Test the Liang-Barsky algorithm for line P1P2 where $P1=(10,10)$ and $P2(60,30)$ against the window with $(X_{wmin}, Y_{wmin})=(15,15)$ & $(X_{wmax}, Y_{wmax})=(25,25)$
13.	Define Ray tracing. Analyze the ray tracing techniques
14.	What are fractals. Explain the different types of fractals
15.	Analyze the Morphing techniques

16.	Explain about various dithering techniques
17.	Discuss various color models used in computer graphics
18.	Explain the various visible surface detection methods
19.	Define projection. Explain the different types of projection
20.	Derive the Bezier curves parametric equation and explain
21.	Discuss about various techniques used to provide text clipping in computer graphics
22.	Mention the limitations of Cohen Sutherland line clipping algorithm. How it can be overcome
23.	Explain Window to view port transformation
24.	Find the points on the Bezier curves which have start & points $P_0(2,3)$ and $P_3(4,-3)$ & it is controlled by $P_1(5,1)$ & $P_2(7,1)$ for $u=0.9$
25.	Explain Sutherland-Hodgeman polygon algorithm
26.	Analyze the various display devices
27.	Clip the line PQ having coordinates $P(4,1)$ & $Q(6,4)$ against the clip window having vertices $A(3,2)$ $B(7,2)$, $C(7,6)$ & $D(3,6)$ using Cohen Sutherland line clipping algorithm. And explain the algorithm
28.	What are 2 D geometric transformations? Explain
29.	Discuss about various hard copy devices
30.	Explain about Sutherland & Cohen subdivision line clipping algorithm
31.	Derive the ellipse drawing algorithm
32.	Explain the following: Colour selection, RGB and YIQ models
33.	What are 2 D geometric transformations.? Explain
34.	Explain 3D object representations.
35.	Explain a method to rotate an object about an axis that is not parallel to the coordinate axis with neat block diagram and derive the transformation matrix for the same
36.	Solve using Liang barsky line clipping algorithm where $(X_{wmin}, Y_{wmin})=(1,9)$ and $(X_{wmax}, Y_{wmax})=(2,8)$ for line segments. $P_1(3,7)$ to $P_2(3,10)$, $P_3(6,6)$ to $P_4(8,9)$, $P_5(-1,7)$ to $P_6(11,1)$ and explain the algorithm
37.	Show how shear transformation may be expressed in terms of rotation and scaling
38.	Calculate the pixel position along the circle path with radius $r=14$ centered on the origin using Bresenham's circle drawing algorithm from point

	(0,4) to point $x=y$.
39.	Derive the ellipse drawing algorithm
40.	Explain the following: Colour selection, RGB and YIQ models
41.	Discuss about filled area primitives
42.	Give the single point perspective projection transformation matrix when projectors are placed on the z axis and also give the two point perspective transformation matrix
43.	Discuss the inverse transformation
44.	Explain about 3D transformations
45.	Define projection. Explain the different types of projection
46.	Derive the Bezier curves parametric equation and explain
47.	Discuss about various techniques used to provide text clipping in computer graphics
48.	Mention the limitations of Cohen Sutherland line clipping algorithm. How it can be overcome
49.	Explain window to view port transformation
50.	Define Ray tracing. Analyze the ray tracing techniques
51.	Derive the ellipse drawing algorithm
52.	Explain the following: Colour selection, RGB and YIQ models
53.	Explain the Illumination models
54.	Test the Liang-Barsky algorithm for line P_1P_2 where $P_1=(10,10)$ and $P_2(60,30)$ against the window with $(X_{wmin}, Y_{wmin})=(15,15)$ & $(X_{wmax}, Y_{wmax})=(25,25)$

Staff in charge

HoD

TWO MARK QUESTION & ANSWERS

Unit - I

1. Define hypothesis.

The formal proof can be using deductive proof and inductive proof. The deductive proof consists of sequence of statements given with logical reasoning in order to prove the first or initial statement. The initial statement is called hypothesis.

2. Define inductive proof.

It is a recursive kind of proof which consists of sequence of parameterized statements that use the statement itself with lower values of its parameter.

3. Define Set, Infinite and Finite Set.

Set is Collection of various objects. These objects are called the elements of the set.

$$\text{Eg : } A = \{ a, e, i, o, u \}$$

Infinite Set is a collection of all elements which are infinite in number.

$$\text{Eg: } A = \{ a \mid a \text{ is always even number} \}$$

Finite Set is a collection of finite number of elements.

$$\text{Eg : } A = \{ a, e, i, o, u \}$$

4. Give some examples for additional forms of proof.

1. Proofs about sets
2. Proofs by contradiction
3. Proofs by counter examples.

5. Prove $1+2+3+\dots+n = n(n+1)/2$ using induction method.

Consider the two step approach for a proof by method of induction

1. Basis of induction :

Let $n = 1$ then $LHS = 1$ and $RHS = 1 + 1 / 2 = 1$ Hence $LHS = RHS$.

2. Induction hypothesis :

To prove $1 + 2 + 3 + \dots + n = n(n + 1) / 2 + (n + 1)$

Consider $n = n + 1$

$$\begin{aligned} \text{then } 1 + 2 + 3 + \dots + n + (n + 1) &= \frac{n(n + 1)}{2} + (n + 1) \\ &= \frac{n^2 + 3n + 2}{2} \\ &= \frac{(n + 1)(n + 2)}{2} \end{aligned}$$

Thus it is proved that $1 + 2 + 3 + \dots + n = n(n + 1) / 2$

6. Write down the operations on set.

i) $A \cup B$ is Union Operation

If $A = \{ 1, 2, 3 \}$ $B = \{ 1, 2, 4 \}$ then

$$A \cup B = \{ 1, 2, 3, 4 \}$$

i.e. combination of both the sets.

ii) $A \cap B$ is Intersection operation

If $A = \{ 1, 2, 3 \}$ $B = \{ 1, 2, 4 \}$ then

$$A \cap B = \{ 2, 3 \}$$

i.e. Collection of common elements from both the sets.

iii) $A - B$ is the difference operation

If $A = \{ 1, 2, 3 \}$ $B = \{ 1, 2, 4 \}$ then

$$A - B = \{ 3 \}$$

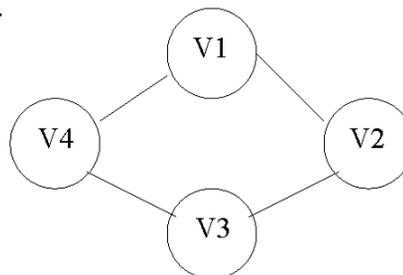
i.e. elements which are there in set A but not in set B.

7. Define Graph, Directed graph and give example.

Graph is consists of finite set of Vertices (Node) V and set of Edges E, edges are nothing but pair of vertices.

It denoted $G = (V, E)$

Eg. :

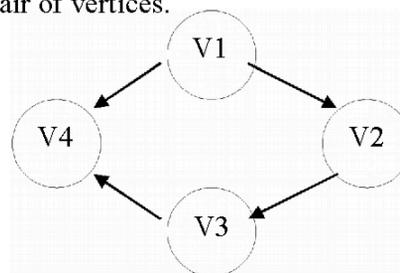


E1 is a edge connecting the vertices V1 and V2.

Directed Graph is consists of finite set of Vertices (Node) V and set of Edges E, edges are nothing but pair of vertices.

It denoted $G = (V, E)$

Eg.



The edge E1 shows the direction to V2 from V1.

8. Write any three applications of Automata Theory.

1. It is base for the formal languages and these formal languages are useful of the programming languages.
2. It plays an important role in complier design.
3. To prove the correctness of the program automata theory is used.
4. In switching theory and design and analysis of digital circuits automata theory is applied.
5. It deals with the design finite state machines.

9. Define Finite Automata.

A finite automata is a collection of 5 tuples $(Q, \Sigma, \delta, q_0, F)$

where Q is a finite set of states, which is non empty.

Σ is a input alphabet, indicates input set.

δ is a transition function or a function defined for going to next state.

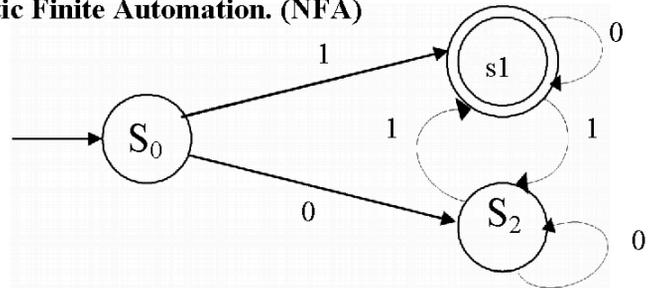
q_0 is an initial state (q_0 in Q)

F is a set of final states.

Two types :

Deterministic Finite Automata (DFA)

Non-Deterministic Finite Automata. (NFA)



10. Define Deterministic Finite Automata.

- The finite automata is called DFA if there is **only one path for a specific input from current state to next state.**

- A finite automata is a collection of 5 tuples $(Q, \Sigma, \delta, q_0, F)$

where Q is a finite set of states, which is non empty.

Σ is a input alphabet, indicates input set.

δ is a transition function or a function defined for going to next state.

q_0 is an initial state (q_0 in Q)

F is a set of final states.

11. Define Non-Deterministic Finite Automata.

The finite automata is called NFA when there exists **many paths for a specific input from current state to next state.**

A finite automata is a collection of 5 tuples $(Q, \Sigma, \delta, q_0, F)$

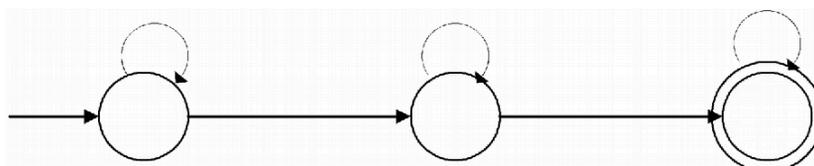
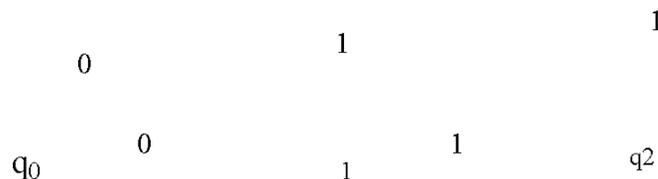
where Q is a finite set of states, which is non empty.

Σ is a input alphabet, indicates input set.

δ is a transition function or a function defined for going to next state.

q_0 is an initial state (q_0 in Q)

F is a set of final states.



12. Define NFA with ϵ transition.

The ϵ is a **character** used to indicate null string.

i.e the string which is used simply for transition from **one state to other state without any input.**

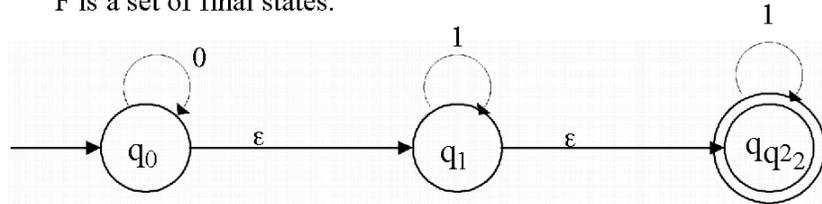
A Non Deterministic finite automata is a collection of 5 tuples $(Q, \Sigma, \delta, q_0, F)$ where Q is a finite set of states, which is non empty.

Σ is a input alphabet, indicates input set.

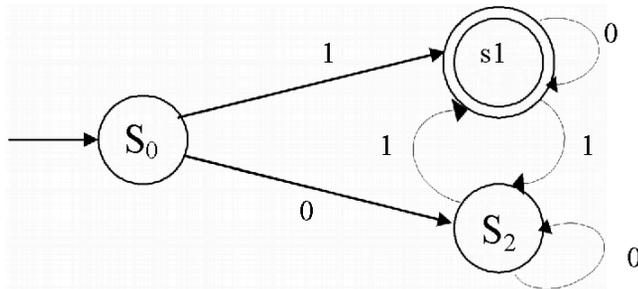
δ is a transition function or a function defined for going to next state.

q_0 is an initial state ($q_0 \in Q$)

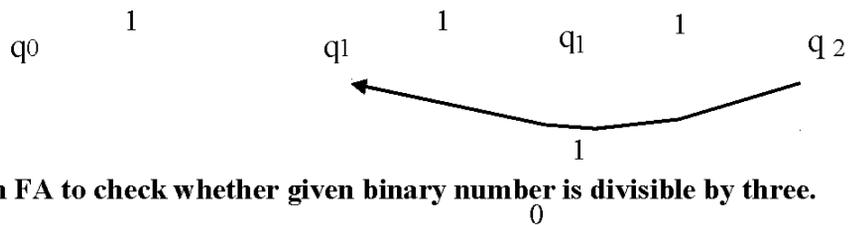
F is a set of final states.



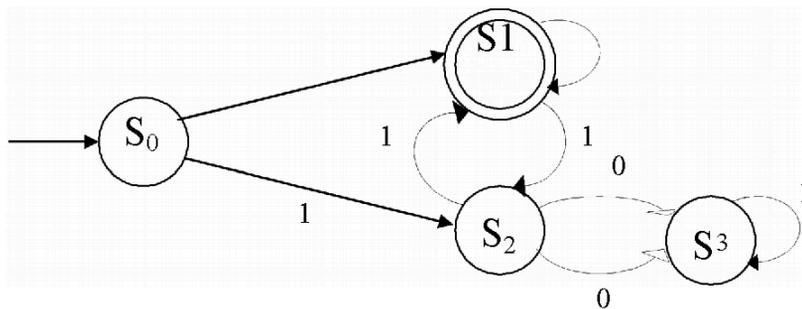
13. Design FA which accepts odd number of 1's and any number of 0's.



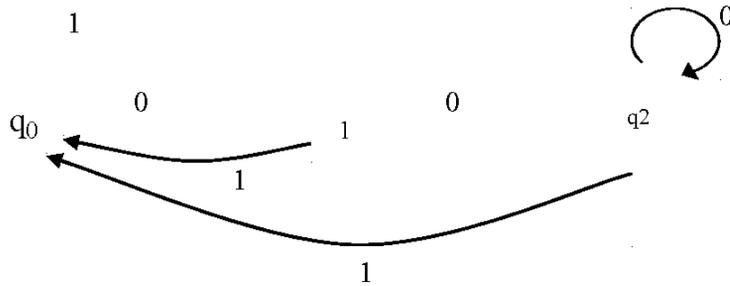
14. Design FA to check whether given unary number is divisible by three.



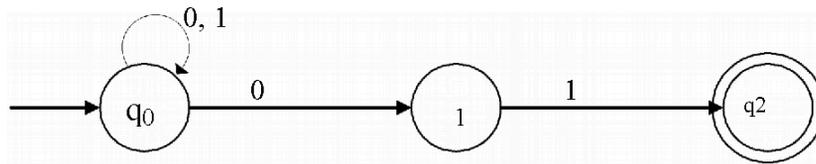
15. Design FA to check whether given binary number is divisible by three.



16. Design FA to accept the string that always ends with 00.



17. Obtain the DFA equivalent to the following NFA.



Solution :

The transition table for given NFA can be drawn as follows

States	Input	
	0	1
{q0}	{q0}{q1}	{q0}
{q1}	-	{q2}
{q2}	-	-

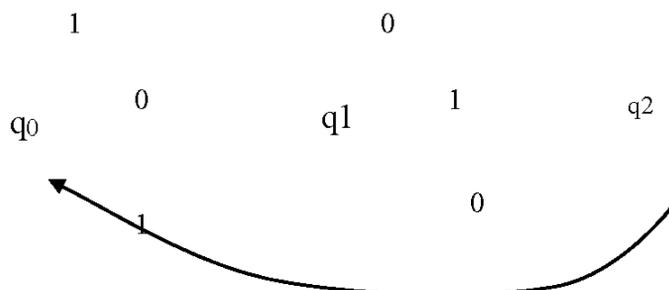
To construct equivalent DFA

- $\delta(q_0, 0) = \{q_0, q_1\}$ a new state - **A**
- $\delta(q_0, 1) = \{q_0\}$ $\delta(q_1, 0) =$
- $\delta(q_1, 1) = \{q_2\}$
- $\delta(q_2, 0) = -$
- $\delta(q_2, 1) = -\delta(\{q_0, q_1\}, 0) = \{q_0, q_1\}$
- $\delta(\{q_0, q_1\}, 1) = \{q_0, q_2\}$ a new state - **B**
- $\delta(\{q_0, q_2\}, 0) = \{q_0, q_1\}$
- $\delta(\{q_0, q_2\}, 1) = \{q_0\}$

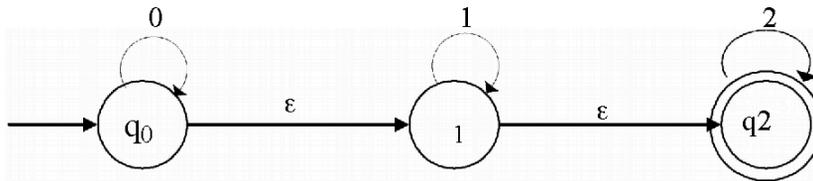
The transition table for DFA

States	Input	
	0	1
{q0}	{q0, q1}	{q0}
{q1}	-	{q2}
{q2}	-	-
{q0, q1}	{q0, q1}	{q0, q2}
{q0, q2}	{q0, q1}	{q0}

The transition diagram for DFA

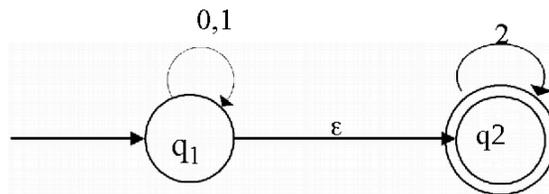


18. Obtain the NFA without ϵ transition to the following NFA with ϵ transition.

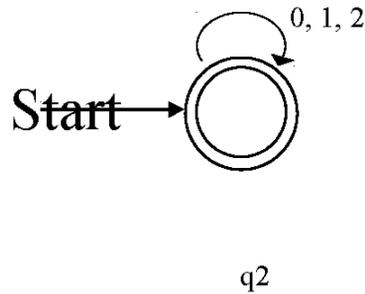


Solution :

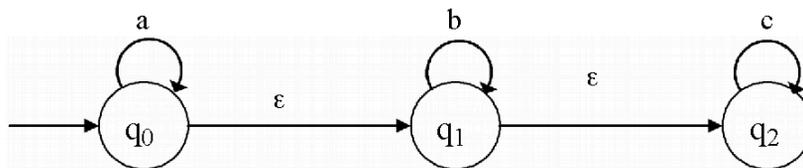
Remove ϵ transition from q_0 to q_1



Now remove transition from q_0 to q_2 . As q_0 to q_2 is transition q_0 will become start and final state both.



19. Obtain the ϵ closure of states q_0 and q_1 in the following NFA with ϵ transition.

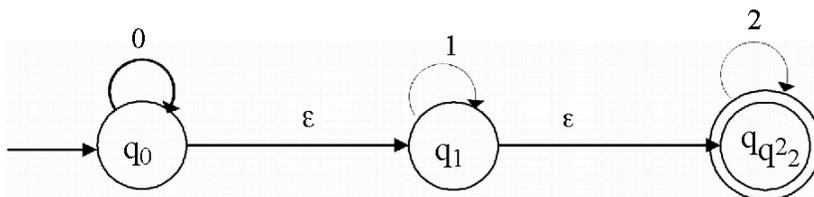


Solution:

ϵ - CLOSURE $\{q_0\} = \{q_0, q_1, q_2\}$

ϵ - CLOSURE $\{q_1\} = \{q_1, q_2\}$

20. Obtain ϵ closure of each state in the following NFA with ϵ move.



Solution:

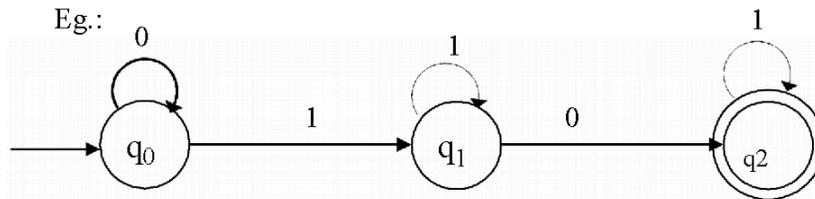
ϵ - CLOSURE $\{q_0\} = \{q_0, q_1, q_2\}$

ϵ - CLOSURE $\{q_1\} = \{q_1, q_2\}$

ϵ - CLOSURE $\{q_2\} = \{q_2\}$

21. Explain a transition diagram.

It is a 5-tuple graph used state and edges represent the transitions from one state to other state.



22. Explain a transition diagram.

It is the tabular representation of the DFA. For a transition table the transition function is used.

Eg.:

States	Input	
	0	1
{q0}	{q1}	{q0}
{q1}	-	{q2}
{q2}	-	-

23. Explain the transition function.

The mapping function or transition function denoted by δ . Two parameters are passed to this transition function : (i) current state and (ii) input symbol. The transition function returns a state which can be called as next state.

Eg.:

$$\delta (q_0, a) = q_1$$

24. Differentiate DFA and NFA?

Sl. No	DFA	NFA
1.	DFA is Deterministic Finite Automata	NFA is Non-Deterministic Finite Automata
2.	For given state, on a given input we reach to deterministic and unique state.	For given state, on a given input we reach to more than one state.
3.	DFA is a subset of NFA	Need to convert NFA to DFA in the design of complier.

25. Write short notes on Minimization of DFA?

- Reducing the number of states from given FA
- First find out which two states are equivalent we than replace those two states by one representative state.
- For finding the equivalent states we will apply the following rule
 - The two states S1 & S2 are equivalent if and only if both the states are final or non-final states.

Unit - II

1. State regular expression.

Let Σ be an alphabet. The regular expressions over Σ and the sets that they denote are defined recursively as follows

- a. \emptyset is a regular expression and denotes the empty set.
- b. ϵ is a regular expression and denotes the set $\{\epsilon\}$
- c. For each 'a' $\in \Sigma$, 'a' is a regular expression and denotes the set $\{a\}$.
- d. If 'r' and 's' are regular expressions denoting the languages L_1 and L_2 respectively then

$r + s$ is equivalent to $L_1 \cup L_2$ i.e. union
 rs is equivalent to L_1L_2 i.e. concatenation
 r^* is equivalent to L_1^* i.e. closure

2. How the kleen's closure or closure of L can be denoted?

$$L^* = \bigcup_{i=0}^n L^i \quad (\text{e.g. } a^* = \{\epsilon, a, aa, aaa, \dots\})$$

3. How do you represent positive closure of L?

$$L^+ = \bigcup_{i=1}^n L^i \quad (\text{e.g. } a^+ = \{a, aa, aaa, \dots\})$$

4. Write the regular expression for the language accepting all combinations of a's over the set $\Sigma = \{a\}$.

$$L = \{ a, aa, aaa, \dots \}$$

$$R = a^* \quad (\text{i.e. kleen closure})$$

5. Write regular expression for the language accepting the strings which are starting with 1 and ending with 0, over the set $\Sigma = \{0,1\}$.

$$L = \{ 10, 1100, 1010, 100010, \dots \}$$

$$R = 1(0+1)^* 0$$

6. Show that $(0^*1^*)^* = (0+1)^*$.

$$\text{LHS} : (0^*1^*)^* = \{ \epsilon, 0, 1, 00, 11, 0011, 011, 0011110, \dots \}$$

$$\text{RHS} : (0+1)^* = \{ \epsilon, 0, 1, 00, 11, 0011, 011, 0011110, \dots \}$$

Hence

$$\text{LHS} = \text{RHS is proved}$$

7. Show that $(r+s)^* \neq r^* + s^*$.

$$\text{LHS} : (r+s)^* = \{ \epsilon, r, s, rs, rr, ss, rrrsss, \dots \}$$

$$\text{RHS} : r^* + s^* = \{ \epsilon, r, rr, rrr, \dots \} \cup \{ \epsilon, s, ss, sss, \dots \}$$

$$= \{ \epsilon, r, rr, rrr, s, ss, ssss, \dots \}$$

Hence

$$\text{LHS} \neq \text{RHS is proved}$$

8. What do you mean by homomorphism?

A string homomorphism is a function on strings that works by substituting a particular string for each symbol.

Eg. $h(0) = ab$

$h(1) = \epsilon$ is a homomorphism, where replace all 0's by ab and replace all 1's by ϵ .

Let $w = 0011$

$h(w) = abab$

9. Explain the application of the pumping lemma.

Pumping Lemma is used to prove the language is not regular.

10. Describe the following by regular expression

a. $L1 =$ the set of all strings of 0's and 1's ending in 00.

b. $L2 =$ the set of all strings of 0's and 1's beginning with 0 and ending with 1.

$r1 = (0+1)^*00$

$r2 = 0(0+1)^*1$

11. Show that $(r^*)^* = r^*$ for a regular expression r.

LHS = $r^* = \{ \epsilon, r, rr, rrr, \dots \}$

$(r^*)^* = \{ \epsilon, r, rr, rrr, \dots \}^*$

$(r^*)^* = \{ \epsilon, r, rr, rrr, \dots \} = r^*$

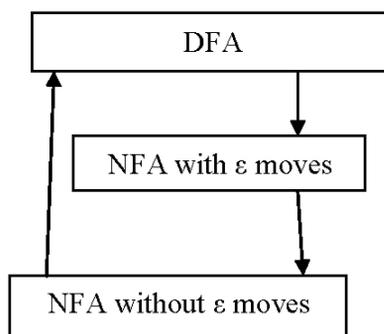
LHS = RHS

12. Write down the closure properties of regular language.

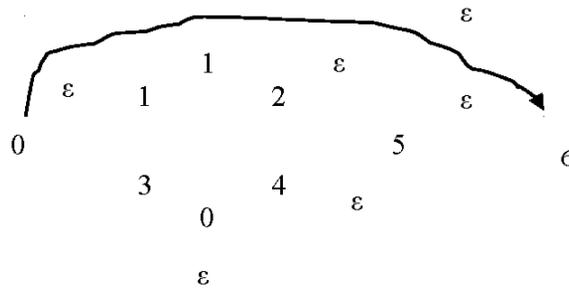
The regular languages are closed under

1. Union
2. Intersection
3. Complement
4. Difference
5. Reversal
6. Closure
7. Concatenation
8. Homomorphism
9. Inverse Homomorphism

13. Write down the relationship between FA and regular expression.



14. Construct NFA with ϵ moves for the regular expression $(0+1)^*$.



15. What is pumping lemma?

Let L be a regular language. Then there exists a constant n such that for every string w in L such that $|w| \geq n$, $w = xyz$ such that

- i) $y \neq \epsilon$
- ii) $|xy| \leq n$
- iii) For all $i \geq 0$, $xy^i z \in L$

16. If $L = \{ \text{The language starting and ending with 'a' and having any combinations of b's in between, that what is r?} \}$

$$r1 = a b^* a$$

17. Give regular expression for

- $L = L1 \cap L2$ over alphabet $\{a,b\}$
- where $L1 = \text{all strings of even length}$
- $L2 = \text{all strings starting with 'b'}$.

Solution :

$$r = r1 + r2$$

$$r = a^n b^n + b (a+b)^*$$

18. State Arden's theorem.

Let P and Q be two regular expression over alphabet Σ . If P does not contain null string ϵ , then

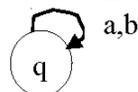
$$R = Q + RP$$

It has the solution

$$R = QP^*$$

19. What is dead state?

All the non final states which transmit to itself for all input symbols in Σ , are called Dead state.



q – non final state, also the dead state.

20. Let $\Sigma = \{0,1\}$ and $\Sigma^1 = \{0,1,2\}$ with $h(0) = 01$ and $h(1) = 112$. Find $h(010)$ and homomorphic image of $L = \{00, 010\}$.

Solution :

$$\Sigma = \{0,1\} \text{ and } \Sigma^1 = \{0,1,2\}$$

h is defined as :

$$h(0) = 01 \text{ and } h(1) = 112$$

$$h(010) = 0111201$$

The homomorphic image of $L = \{00, 010\}$ is $h(L) = \{0101, 0111201\}$.

Unit - III

1. Let $G = (\{S,C\}, \{a,b\}, P, S)$ where P consists of $S \rightarrow aCa, C \rightarrow aCa, C \rightarrow aCa$, Find $L(G)$?

Solution:

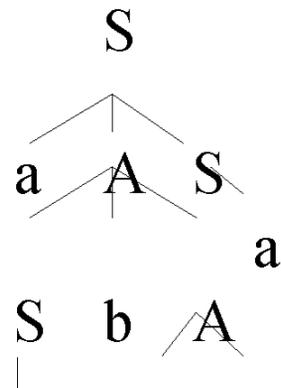
$S \rightarrow aCa$
 $\rightarrow aaCaa \quad C \rightarrow aCa$
 \vdots
 $\rightarrow a^n C a^n$
 $\rightarrow a^n b a^n \quad C \rightarrow b$

$$L(G) = \{ a^n b a^n ; n > 0 \}$$

2. Consider G whose productions are $S \rightarrow aAS / a, A \rightarrow SbA / SS / ba$, show that $\rightarrow aabbaa$ and construct a derivation tree.

Solution:

$S \rightarrow aAs$
 $\rightarrow aSbAs \quad A \rightarrow SbA$
 $\rightarrow aabAS \quad S \rightarrow a$
 $\rightarrow aabbaS \quad A \rightarrow ba$
 $\rightarrow aabbaa \quad S \rightarrow a$



3. Find $L(G)$ where $G = (\{S\}, \{0,1\}, \{S \rightarrow 0S1, s \rightarrow \epsilon\}, S)$

Solution:
S

leaf and is the only son of its father.

5. Construct CFG $L = \{ a^n b^n ; n \geq 1 \}$.

Solution:

The Production are
 $S \rightarrow aSb / \epsilon$ where $G = (\{S\}, \{a,b,\epsilon\}, P, S)$

6. Find a LM derivation for $aaabbabbba$ with the productions.

$P : S \rightarrow aB / bA, A \rightarrow$
 $a / S / bAA, B \rightarrow b / bS$
 $/ aBB$

Solution:

$S \rightarrow aB \rightarrow aaBB \rightarrow$
 $aaaBBB \rightarrow aaabBB \rightarrow$
 $aaaabbB \rightarrow aaabbaBB \rightarrow aaabbabB$
 $\rightarrow aaabbabbS$
 \rightarrow
 $aaabbabbbA S$
 $\rightarrow aaabbabbba$

7. Find $L(G)$, $S \rightarrow aSb, S \rightarrow ab$.

Solution:

$S \rightarrow aSb$
 $\rightarrow aaSbb$
 \vdots
 \rightarrow
 $a^n S b^n$
 $\rightarrow a^n b^n \quad C$
 $\rightarrow ab$

$$L(G) = \{ a^n b^n ; n \geq 1 \}$$

8. Show that $id^* id$ can be generated by two distinct leftmost derivation in the grammar

$$E \rightarrow E+E / E * E / (E) / id$$

Solution:

$$\begin{aligned}
 \text{(i) } E &\rightarrow E + E \\
 &\rightarrow id + E \\
 &\rightarrow id + E * E \\
 &\rightarrow id + id * E & \text{ (ii) } E &\rightarrow E \\
 &\rightarrow id + id * id & &\rightarrow E \\
 & & &\rightarrow E \\
 & & &\rightarrow E \\
 & & &\rightarrow id
 \end{aligned}$$

We showed that $id+id*id$ can be generated by two distinct LMD.

9. Define pushdown automaton.

A Pushdown Automata is a finite automation with extra resource called stack. It consists of 7 tuples.

$$P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$$

Where

Q – Finite set of states

Σ - Finite set of input symbols

Γ - Finite set of stack symbols

δ -

Transition

function q_0

– Start

State

Z_0 – Start symbol of the stack

F – Final State.

10. What are the different ways of language acceptances by a PDA and define them.

i) Acceptance by final state

$L(M) = \{ w \mid (q_0, w, z_0) \vdash^* (p, \varepsilon, \gamma) \text{ for some } p \in F \text{ and } \gamma \in \{\varepsilon\} \}$

ii) Acceptance by empty stack

$N_e(M) = \{ w \mid (q_0, w, z_0) \vdash^* (p, \varepsilon, \varepsilon) \text{ for some } p \in Q \}$

11. Write a CFG for the set of strings which does not produce any palindromes.

Here the grammar should be designed in such a way that $w \neq w^R$

$$\begin{aligned}
 S &\rightarrow \\
 &aSa / \\
 &bSb / C \\
 C &\rightarrow \\
 &aAb / \\
 &bAa \\
 A &\rightarrow aA \\
 &/ bA / \varepsilon
 \end{aligned}$$

12. Find the language for the CFG S

$$\begin{aligned}
 S &\rightarrow aSa / \\
 &aAb \\
 A &\rightarrow bAa / b a
 \end{aligned}$$

Solution:

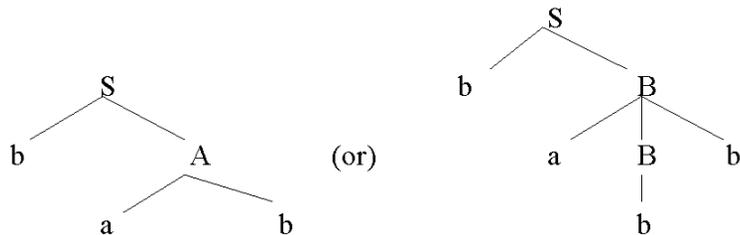
$$\begin{aligned}
 S &= aSb \Rightarrow aaSbb \Rightarrow \dots \Rightarrow a^n S b^n \Rightarrow a^n a A b b^n \Rightarrow a^n a b A a b b^n \\
 &\Rightarrow a^n a b n a a b b^n \Rightarrow a^n b^n a^n b^n \\
 L &= \{ a^n b^n a^n b^n ; m, n \geq 1 \}
 \end{aligned}$$

13. Find the derivation tree for the grammar

$$G = (\{S,$$

$\{A, B\}$,
 $\{a, b\}$, P , S
 Where P is
 given by
 $S \rightarrow Aa / bB$
 $A \rightarrow ab$
 $B \rightarrow aBb / a$

Solution:



**14. Construct a PDA that accepts
 the language generated by the
 grammar**

$S \rightarrow aSbb / aab$

Solution:

The PDA $A =$

$(\{q\}, \{a, b\},$
 $\{S, a, b\}, \delta, q, S)$

Where $\delta :$

i) $\delta(q, z_0, S) = \{(q, aSbb), (q,$
 $abb)\}$

ii) $\delta(q, a, a) = \{(q, \epsilon)\}$

iii) $\delta(q, b, b) = \{(q, \epsilon)\}$

**15. Construct a PDA that accepts
 the language generated by the
 grammar**

$S \rightarrow aABB, A \rightarrow aB / a, B \rightarrow bA /$
 b

Solution:

The PDA $A = (\{q\},$
 $\{a,b\},$

$\{S,A,B,Z,a,b\},\delta,$

$q,S\}$ Where $\delta :$

- i) $\delta(q, z, S) = \{(q, aABB)\}$
- ii) $\delta(q, z, A) = \{(q, aB, (q,a))\}$
- iii) $\delta(q, z, B) = \{(q, bA, (q,b))\}$
- iv) $\delta(q, a, a) = \{(q, \varepsilon)\}$
- v) $\delta(q, b, b) = \{(q, \varepsilon)\}$

16. Define parse tree.

A data structure to represent the source program in a compiler is called parse tree. Parse tree can have nodes and edges.

17. How do you convert CFG to a PDA.

Let $G = (V, T, P, S)$ be a CFG. Then construct a PDA P that accepts $L(G)$ by empty stack as follows :

$$P = (\{q\}, \\ T, V \cup T, \\ \delta, q, S)$$

Where δ is given by

- 1) For each variable A ,
 $\delta(q, \epsilon, A) = \{(q, \beta) \mid A \rightarrow \beta \text{ is a production of } P\}$
- 2) For each terminal a ,
 $\delta(q, a, a) = \{(q, \epsilon)\}$

18. Define Deterministic PDA.

A PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ to be deterministic iff
1) $\delta(q, a, X)$ has at most one member for any q in Q , a in Σ or $a = \epsilon$ and X in Γ^* .

2) If $\delta(q, a, X)$ is not empty, for some a in Σ , then $\delta(q, \epsilon, X)$ must be empty.

19. Is it true that NPDA is more powerful than that of DPDA? Justify your answer.

No, NPDA is not powerful than DPDA. Because NPDA may produce ambiguous grammar by reaching its

final state or by emptying its stack. But DPDA produces only unambiguous grammar.

20. What is the additional feature PDA has when compared with NFA? Is PDA superior over NFA in the sense L acceptance? Justify your answer.

PDA is superior NFA by having the following additional features.

- Stack which is used to store the necessary tape symbols and use the state to remember the conditions.
- Two ways of L acceptances, one by reaching its final state and another by emptying its stack.

1. What are the two major normal forms for context-free grammar?

The two Normal forms are

i. Chomsky Normal Form (CNF)

ii. Greibach Normal Form (GNF)

2. What is a useless symbol?

A symbol x is useful if there is a derivation

$$S \Rightarrow^* \alpha x \beta \Rightarrow^* w \text{ for some } \alpha, \beta, w \in \Sigma^*$$

or else, it is useless.

3. How do you simplify the context-free grammar?

- First eliminate useless symbols, where the variable or terminals that do not appear in any derivation of a terminal string from the start symbol.
- Next eliminate ϵ - productions which is of the form $A \rightarrow \epsilon$ for some variable A .
- Eliminate unit productions, which are of the form $A \rightarrow B$ for variables A, B .
- Finally use any of the normal forms to get the simplified CFG.

4. Define Nullable Variable?

Nullable variable in a CFG G

$G = (V, T, P, S)$ can be defined as follows.

- Any variable A for which P contains the production $A \rightarrow A$, is nullable.
- If P contains the production $A \rightarrow B_1, B_2, \dots, B_n$ and B_1, B_2, \dots, B_n are nullable variables, then A is nullable.
- No other variables in V are nullable.

5. Define generating symbol?

Let $G = (V, T, P, S)$ is

generating, if $X \Rightarrow^* w$ for some terminal string w . e.g.

$$\begin{array}{l} A \rightarrow aAB / \epsilon \\ B \rightarrow b \end{array}$$

Then A is a generating symbol since $A \Rightarrow^* ab$

6. Let $G = (V, T, P, S)$ with the productions given by

$$S \rightarrow aSbS / B / \epsilon$$

$B \rightarrow abB$ Eliminate the useless production.

Solution:

Remove B is useless production because of Variable is not reachable.

$$S \rightarrow aSbS / \epsilon$$

7. What is substitution Rule?

A production $A \rightarrow x_1 B x_2$ can be eliminated from a grammar if B is replaced by all strings derived by B in one step, provided A and B are variables.

8. What is a useful production?

Let $G = (V, T, P, S)$ be a CFG. A variable $A \in V$ is said to be useful iff there is at least one $w \in L(G)$ such that $S \Rightarrow^* xAy \Rightarrow^* w$ with $x, y \in (V \cup T)^*$.

9. Determine whether the grammar G has a useless production?

$$\begin{aligned} S &\rightarrow A \\ A &\rightarrow aA / \varepsilon \\ B &\rightarrow bA \end{aligned}$$

The variable B is useless, since it is used by the start variable or by the variable in the start production.

$B \rightarrow bA$ is a useless production.

10. Write a procedure to eliminate ϵ production.

i) For all productions $A \rightarrow \epsilon$, put A into V

ii) Repeat the following steps until no new variable are added.

- a. For all productions

$$B \rightarrow A_1 A_2 A_3 \dots A_n,$$

where $A_1 A_2 A_3 \dots A_n$ are

b. ^{in V} Put B into V

11. Write the procedure to eliminate the unit productions.

i) Find all variables B, for each A such that

$$A \Rightarrow^* B$$

ii) The new grammar G is obtained by letting into P all non-unit productions of P.

iii) For all A and B satisfying $A \Rightarrow^* B$, add to p

$$A \rightarrow y_1 / y_2 / y_3 / \dots / y_n,$$

where $B \rightarrow y_1 / y_2 / y_3 / \dots / y_n$

is the set of productions in P.

12. Define CNF.

A CFG without any ϵ -production is generated by a grammar in which the productions are of the form.

$$A \rightarrow BC \quad \text{or} \quad A \rightarrow a, \text{ where } A, B \in V \text{ and } a \in T.$$

13. What is GNF?

Every CFL L without ϵ can be generated by a grammar for which every production is of the form $A \rightarrow \alpha$, where $A \in V$, $\alpha \in T^+$, is a string of variables.

14. What is a Turing Machine?

A finite state machine with storage is called a Turing Machine.

15. Define a Turing Machine.

The Turing Machine is denoted by

$$M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$$

Where

Q – Finite set of states

Σ - Finite set of input symbols

Γ - Finite set of stack symbols

δ - Transition function - $Q \times \Sigma \times \Gamma \rightarrow Q \times \Sigma \times \Gamma \times \{L, R\}$, Where L, R –

Directions. q_0 – Start State

B – a Start symbol of the Σ , a blank

F – Final State.

16. What are the required fields of an instantaneous description or configuration of a TM.

It requires

- The state of the TM
- The contents of the tape
- The position of the tape head on the tape.

17. What is multiple tracks Turing machine?

UNIT – V

1. What is the weak-form of Turing thesis?

A Turing Machine can compute anything that can be computed by a general purpose digital computer.

2. What is the strong-form of Turing thesis?

A Turing Machine can compute anything that can be computed. This is the strong form of Turing thesis.

3. When a language is said to be recursively enumerable?

A language is recursively enumerable if there exists a Turing Machine that accepts every string of the language and does not accept strings that are not in the language.

4. When a language is said to be recursive?

A language L is said to be recursive if there exists a Turing machine M that accepts L , and goes to halt state or else M rejects L .

5. What is diagonalization language?

The language L_d . Which consists of all those strings w such that the Turing machine represented by w does not accept the input w . $L_d = \{ w_i \mid w_i \notin L(M_i) \}$

6. Define decidability (or) decidable problems?

A problem is said to be decidable if there exists a Turing machine which gives one 'yes' or 'no' answer for every input in the language.

(or)

A problem is said to be decidable if it is a recursive language.

7. Define Undecidable problem?

If a problem is not a recursive language, then it is called undecidable problem.

8. Define Universal language?

A Universal Turing Machine M_u is an automation, that given as input the description of any Turing Machine M and a string w , can simulate the computation of M on w .

9. What are the reasons for a TM not accepting its input?

- i) The TM may halt in a non final state.
- ii) The TM may enter into an indefinite loop.

10. Define trivial property?

A property is trivial if it is either empty or is all RE languages.

11. Define Rice Theorem?

Every non-trivial property of the RE languages is undecidable.

12. Define post's correspondence problem?

An instance of PCP consists of two lists, $A = w_1, w_2, w_3, \dots, w_k$

$B = x_1, x_2, x_3, \dots, x_k$ of strings over some Σ . This instance of PCP has a solution if there is any sequence of integers i_1, i_2, \dots, i_m with $m \geq 1$, such that

$$w_{i_1} w_{i_2} w_{i_3} \dots w_{i_m} = x_{i_1} x_{i_2} x_{i_3} \dots x_{i_m}$$

..... X_m

The sequence of i_1, i_2, \dots, i_m is a solution to this instance of PCP.

13. Let A and B be lists of three strings each, as defined in the following table?

A		B
w		x
1	1	111
2	10111	10
3	10	0

Find the instance of post correspondence Problem.

Solution :

Apply $w_{i_1}, w_{i_2}, w_{i_3}, \dots, w_{i_m} = x_{i_1}, x_{i_2}, x_{i_3}, \dots, x_{i_m}$ to this problem.

Take $M = 4$

$$w_2 w_1 w_1 w_3 = x_2 x_1 x_1 x_3$$

$$10 111 111 0 = 10 111 111 0$$

Instance = 2,1,1,3.

14. Define modified post's correspondence problem?

Given lists A and B, of K strings each from Σ^* , say

$$A = w_1, w_2, w_3, \dots, w_k$$

$$B = x_1, x_2, x_3, \dots, x_k$$

Does there exist a sequence of integers i_1, i_2, \dots, i_r such that

$$w_{i_1} w_{i_2} w_{i_3} \dots w_{i_m} = x_{i_1} x_{i_2} x_{i_3} \dots x_{i_m}$$

The sequence of i_1, i_2, \dots, i_m is a solution to this instance of PCP.

15. Define problem solvable in polynomial Time?

A Turing Machine M is said to be of time complexity $T(n)$ if whenever m given an input w of length n,

m halts after making atmost $T(n)$ moves, regardless of whether or not m accepts.

16. Define the classes P and NP?

P consists of all those languages or problems accepted by some Turing Machine that runs in some polynomial amount of time, as a function of its input length.

NP is the class of languages or problems that are accepted by Nondeterministic TM's with a polynomial bound on the time taken along any sequence of non – deterministic choices.

17. Define NP – Complete Problem?

A language L is NP – complete if the following statements are true. a. L is in NP b. For every language L_1 in NP there is a polynomial time reduction of L_1 to L

18. What are tractable problems?

The problems which are solvable by polynomial time algorithms are called tractable problems.

19. What are the properties of recursive enumerable sets Which are undecidable?

i) Emptiness ii) Finiteness iii) Regularity iv) Context – freedom

20. What are the properties of recursive and Recursively Enumerable Language?

1. The complement of a Recursive language is Recursive.

2. The union of two recursive languages are recursive.

The union of two RE languages are RE.

3. If a language L and complement L are both RE, then L is recursive.

V.S.B. ENGINEERING COLLEGE, KARUR
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
Academic Year 2018-19(ODD Semester)

Class: III Year/ V Semester B.E., CSE

Name of the Subject: CS6503/Theory of Computation

Name of Faculty Member: Mr.K.Suresh Kumar

ASSIGNMENT TOPICS

S.No	Reg.No	Name of the Student	Assignment Topic
1.	922516104061	PADMAPRIYA S	Explain about automaton and its types.
2.	922516104062	PAVITHRA L	Elucidate the importance of DFA with real time example.
3.	922516104063	PAVITHRA M	Explain transition table and transition diagram by taking an example.
4.	922516104064	PAVITHRA T	Mention the fields where theory of computation is being used and narrate the relations with each fields
5.	922516104065	PRADEEP S	Justify NFA is abnormal, elucidate its application in real world.
6.	922516104066	PRADEEPA S	Design DFA accepting the following language over the alphabet {0,1}. i) Number of 0's is a multiple of 2 ii) Number of 1's is not a multiple of 3
7.	922516104068	PRAVEEN T	Construct a minimized DFA for the RE (10+0+11)0*1.
8.	922516104069	PRAVEENKUMAR T	Explain extended transition function by taking an example.
9.	922516104070	PREETHI S	Briefly discuss about the construction of DFA from NFA.
10.	922516104071	PRIYADHARSHINI B	Prove mathematical induction $\sum_{x=1}^n x^2 = n(n+1)(2n+1)/6$. Discuss about inductive proofs.
11.	922516104072	PRIYANKA R	Convert the following grammar into GNF. S \rightarrow XY 0 X \rightarrow 00X 1 X1 Y \rightarrow 1X1
12.	922516104073	PRIYANKA S	Design DFA accepting the following language over the alphabet {a,b}. i) Number of a's is a multiple of 2 ii) Number of b's is a multiple of 3
13.	922516104074	RAHUL R	Write short notes on the following: (i) Proof by contradiction. (ii) Proof by contra positive.
14.	922516104075	RAJKUMAR D	Narrate the influence of automaton in electronic devices.
15.	922516104077	RANJITHKUMAR R	Construct a minimized DFA for the RE (0+1)0*1+1*.
16.	922516104078	ROSHINI M	Prove the equivalence of CFL and PDA.
17.	922516104079	ROSHINI S	Convert the following grammar into GNF. S \rightarrow XA BB B \rightarrow b SB X \rightarrow b A \rightarrow a Explain GNF.

18.	922516104080	SAKTHI SANTHOSH K	Discuss about inductive and deductive proofs.
19.	922516104081	SAKTHIVEL P	Explain CNF and GNF.
20.	922516104082	SAKTHIVEL S	Explain acceptance by final state and acceptance by empty stack of a pushdown automata.
21.	922516104083	SANKAVI M	Discuss about DFA and NFA with examples.
22.	922516104084	SANTHANAM R	Construct a minimized DFA for the RE $b^*(a+b) a^*(a+b)$.
23.	922516104085	SANTHIYA V	Justify DFA is having normal state transitions, elucidate its application in real world
24.	922516104086	SANTHIYA V	Construct a minimized DFA for the RE $11(0+1)^*01$.
25.	922516104089	SARANYA K	Briefly discuss about the construction of PDA and NPDA
26.	922516104090	SARANYA T	Narrate the influence of automaton in electronic devices.
27.	922516104091	SAVITHA S	Construct a NFA accepting the set of strings over $\{0, 1\}$ ending in 110. Use it to construct a DFA accepting the same set of strings.
28.	922516104092	SELVAPRIYA S	Prove that, if L is accepted by NFA with ϵ transitions, then L is accepted by NFA without ϵ transitions. Discuss Arden's theorem.
29.	922516104093	SELVAPRIYA N S	Discuss an ambiguous grammar with example. Explain about homomorphism.
30.	922516104094	SHAJITHA SANOFER K	Discuss the closure properties of regular languages.
31.	922516104096	SHEELA AROCKIA MARY P	Construct a PDA to accept the language $L = \{ a^n b^{2n} / n > 1 \}$ by empty state and final state.
32.	922516104097	SHIYAM V	Construct a minimized DFA for the RE $(0+1)^*001$.
33.	922516104098	SIBI SIDDHARTH J	Give formal pushdown automata for $L = \{ \{wcw^r \mid w \text{ in } (0+1)^*\} \}$ by empty stack.
34.	922516104099	SIVAKUMAR M	$L = \{ \{ WW^R \mid w \text{ is in } (0+1)^* \} \}$. PDA P that accepts string x by final state if and only if x is of the form WW^R
35.	922516104100	SIVASARATHY S	Construct a PDA to accept the language $L = \{ a^n b^n / n > 1 \}$ by empty state and final state.
36.	922516104101	SOBIYA D	Construct a NFA accepting the set of strings over $\{a,b\}$ ending in ba. Use it to construct a DFA accepting the same set of strings.
37.	922516104102	SOUNDARYA N	Prove for every $n \geq 1$ by mathematical induction $\sum i^3 = \{n(n+1)/2\}^2$.
38.	922516104103	SOWMYA M	Using pumping lemma for the regular sets, prove that the language $L = \{ a^n / n \geq 1 \}$.

39.	922516104104	SRI NANTHINI K	Construct a PDA for $L=\{(a+b)^* \mid n_{aw} < n_{bw}\}$ Construct a DPDA for even length palindrome.
40.	922516104105	SURIYAKUMAR S	Construct an NFA equivalent to $(0+1)^*(00+11)(0+1)^*$
41.	922516104106	SURYA C	Construct a minimized DFA for the RE $(a+b)a^*(a+b)$.
42.	922516104107	SWATHILAKSHMI M	Construct a NFA accepting the set of strings over $\{0,1\}$ starting as 01. Use it to construct a DFA accepting the same set of strings.
43.	922516104108	THILAGA R	Construct the following grammar in GNF $S \rightarrow ABC BaB$ $A \rightarrow aA BaC aaa$ $B \rightarrow bBb aD$ $C \rightarrow CA AC$ $D \rightarrow \epsilon$
44.	922516104110	UDHAYARANJANI R	Prove equivalence of DFA and NFA with examples.
45.	922516104111	VAANMATHI P	Construct a PDA for $L=\{(1+0)^* \mid n_{1w} < n_{0w}\}$ Construct a DPDA for even length palindrome.
46.	922516104112	VEERAKRITHIKA R.D	Construct DFA to accept language having even no. of 1's. Construct DFA to accept language having odd no. of 0's.
47.	922516104113	VEERASUKANYA V	Construct DFA and NFA for acceptance of STRAT and REGULAR
48.	922516104114	VELAN N	Using pumping lemma for the regular sets, prove that the language $L=\{a^m b^n \mid m > n\}$.
49.	922516104115	VENGATESHKUMAR R	Construct a minimized DFA for the RE $1^*(10+0)01$.
50.	922516104116	VIDHYASRI .T	Construct a DPDA for even length palindrome.
51.	922516104117	VIGNESH L	Discuss about types of automaton briefly.
52.	922516104118	VIGNESH S P	Construct a PDA for $L=\{(01)^* \mid n_{1w} < n_{0w}\}$ Construct a DPDA for odd length palindrome.
53.	922516104119	VINITHA M	Construct a NFA accepting the set of strings over $\{0,1\}$ ending in 110. Use it to construct a DFA accepting the same set of strings.
54.	922516104120	YOGESH.A	Prove the equivalence of CFL and PDA.
55.	922516104301	SOWMIYA G	Using pumping lemma for the regular sets, prove that the language $L= \{a^n b^n \mid n \geq 1\}$.
56.	-	ABARNA.S	Construct a minimized DFA for the RE $(0+1)^*(0+11)0^*1$.

Staff in charge

HoD

UNIT I

- 1 List any four ways of theorem proving.
- 2 Define Alphabets.
- 3 Write short notes on Strings.
- 4 What is the need for finite automata?
- 5 What is a finite automaton? Give two examples.
- 6 Define DFA.
- 7 Explain how DFA process strings.
- 8 Define transition diagram.
- 9 Define transition table.
10. Define the language of DFA.
11. Construct a finite automata that accepts $\{0,1\}^+$.
12. Give the DFA accepting the language over the alphabet 0,1 that have the set of all strings ending in 00.
13. Give the DFA accepting the language over the alphabet 0,1 that have the set of all strings with three consecutive 0's.
14. Give the DFA accepting the language over the alphabet 0,1 that have the set of all strings with 011 as a substring.
15. Give the DFA accepting the language over the alphabet 0,1 that have the set of all strings whose 10th symbol from the right end is 1.
16. Give the DFA accepting the language over the alphabet 0,1 that have the set of all strings such that each block of 5 consecutive symbol contains at least two 0's.
17. Give the DFA accepting the language over the alphabet 0,1 that have the set of all strings that either begins or end(or both) with 01.
18. Give the DFA accepting the language over the alphabet 0,1 that have the set of all strings such that the no of zero's is divisible by 5 and the no of 1's is divisible by 3.
19. Find the language accepted by the DFA given below.
20. Define NFA.
21. Define the language of NFA.
22. Is it true that the language accepted by any NFA is different from the regular language? Justify your Answer.
23. Define ϵ -NFA.
24. Define ϵ closure.
25. Find the ϵ closure for each state from the following automata.

Part B

1.a) If L is accepted by an NFA with ϵ -transition then show that L is accepted by an NFA without ϵ -transition.

b) Construct a DFA equivalent to the NFA.

$$M = (\{p, q, r\}, \{0, 1\}, \delta, p, \{q, s\})$$

Where δ is defined in the following table.

δ	0	1
p	{q,s}	{q}
q	{r}	{q,r}
r	{s}	{p}
s	-	{p}

2. a) Show that the set $L = \{a^n b^n / n \geq 1\}$ is not a regular. (6) b) Construct a DFA equivalent to the NFA given below: (10)

	0	1
p	{p,q}	P
q	r	R
r	s	-
s	s	S

3.a) Check whether the language $L = \{0^n 1^n / n \geq 1\}$ is regular or not? Justify your answer.

b) Let L be a set accepted by a NFA then show that there exists a DFA that accepts L .

4. Define NFA with ϵ -transition. Prove that if L is accepted by an NFA with ϵ -transition then L is also accepted by a NFA without ϵ -transition.

5.a) Construct a NFA accepting all string in $\{a, b\}^+$ with either two consecutive a's or two consecutive b's.

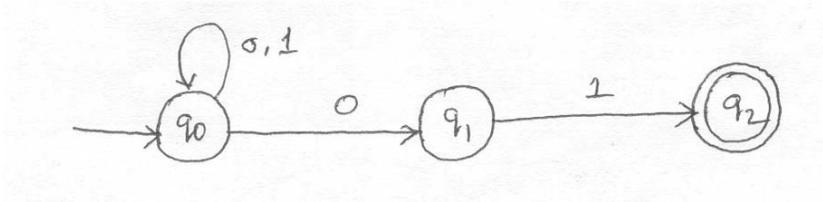
b) Give the DFA accepting the following language: set of all strings beginning with a 1 that when interpreted as a binary integer is a multiple of 5.

6. Draw the NFA to accept the following languages.

- (i) Set of Strings over alphabet $\{0,1,\dots,9\}$ such that the final digit has appeared before. (8)
- (ii) Set of strings of 0's and 1's such that there are two 0's separated by a number of positions that is a multiple of 4.

7.a) Let L be a set accepted by an NFA. Then prove that there exists a deterministic finite automaton that accepts L . Is the converse true? Justify your answer. (10)

b) Construct DFA equivalent to the NFA given below: (6)



8.a) Prove that a language L is accepted by some ϵ -NFA if and only if L is accepted by some DFA. (8)

b) Consider the following ϵ -NFA. Compute the ϵ -closure of each state and find its equivalent DFA. (8)

	ϵ	A	b	C
p	{q}	{p}	Φ	Φ
q	{r}	ϕ	{q}	Φ
*r	Φ	ϕ	ϕ	{r}

9.a) Prove that a language L is accepted by some DFA if L is accepted by some NFA.

b) Convert the following NFA to its equivalent DFA

	0	1
p	{p,q}	{p}
q	{r}	{r}
r	{s}	ϕ

*s	{s}	{s}
----	-----	-----

10.a) Explain the construction of NFA with ϵ transition from any given regular expression.

b) Let $A = (Q, \Sigma, \delta, q_0, \{q_f\})$ be a DFA and suppose that for all $a \in \Sigma$ we have $\delta(q_0, a) = \delta(q_f, a)$. Show that if x is a non empty string in $L(A)$, then for all $k > 0$, x^k is also in $L(A)$.

UNIT II

- 1 Define Regular expression. Give an example.
- 2 What are the operators of RE.
- 3 Write short notes on precedence of RE operators.
Write Regular Expression for the language that have the set of strings over $\{a, b, c\}$ containing at least one a and at least one b .
- 4 Write Regular Expression for the language that have the set of all strings of 0's and 1's whose 10th symbol from the right end is 1.
- 6 Write Regular Expression for the language that has the set of all strings of 0's and 1's with at most one pair of consecutive 1's.
- 7 Write Regular Expression for the language that have the set of all strings of 0's and 1's such that every pair of adjacent 0's appears before any pair of adjacent 1's.
- 9 Write Regular Expression for the language that have the set of all strings of 0's and 1's whose no of 0's is divisible by 5.
- 11 Write Regular Expression for the language that has the set of all strings of 0's and 1's not containing 101 as a substring.
- 12 Write Regular Expression for the language that have the set of all strings of 0's and 1's such that no prefix has two more 0's than 1's, not two more 1's than 0's.
13. Write Regular Expression for the language that have the set of all strings of 0's and 1's whose no of 0's is divisible by 5 and no of 1's is even.
14. Give English descriptions of the languages of the regular expression $(1 + \epsilon)(00^*1)^*0^*$.
15. Give English descriptions of the languages of the regular expression $(0^*1^*)^*000(0+1)^*$.
16. Give English descriptions of the languages of the regular expression

- $(0+10)^*1^*$.
17. Convert the following RE to ϵ -NFA. 01^* .
 18. State the pumping lemma for Regular languages.
 19. What are the application of pumping language?
 20. State the closure properties of Regular language.
 21. Prove that if L and M are regular languages then so is LUM.
 22. What do you mean by Homomorphism?

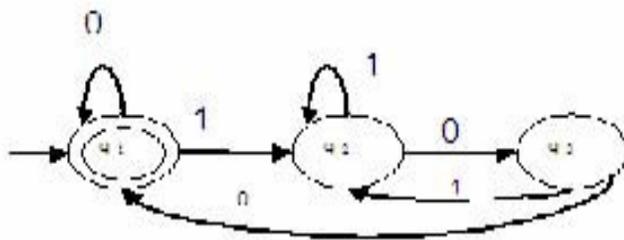
Suppose H is the homomorphism from the alphabets $\{0,1,2\}$ to the alphabets $\{a,b\}$ defined by $h(0)=a$ $h(1)=ab$ $h(2)=ba$. What is $h(0120)$ and $h(21120)$.

23. Suppose H is the homomorphism from the alphabets $\{0,1,2\}$ to the alphabets $\{a,b\}$ defined by $h(0)=a$ $h(1)=ab$ $h(2)=ba$. If L is the language $L(01^*2)$ what is $h(L)$.

24. Let R be any set of regular languages is $\cup R$ regular? Prove it.
25. Show that the compliment of regular language is also regular.
26. What is meant by equivalent states in DFA.

PART-B

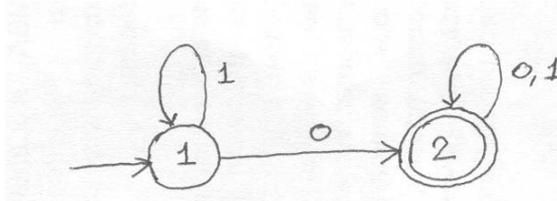
- 1.a) Construct an NFA equivalent to $(0+1)^*(00+11)$
- 2.a) Construct a Regular expression corresponding to the state diagram given in the following figure.



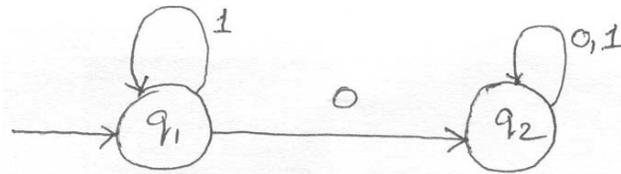
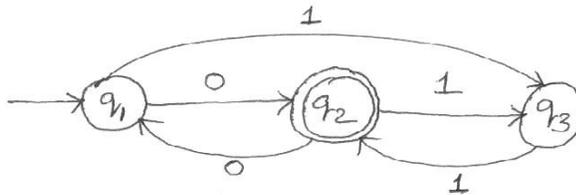
- b) Show that the set $E = \{0^i 1^i \mid i \geq 1\}$ is not Regular. (6)

- 3.a) Construct an NFA equivalent to the regular expression $(0+1)^*(00+11)(0+1)^*$.

- b) Obtain the regular expression that denotes the language accepted by the following DFA.



- 4.a) Construct an NFA equivalent to the regular expression $((0+1)(0+1)(0+1))^*$
- b) Construct an NFA equivalent to $10+(0+11)0^*1$
- 5.a) Obtain the regular expression denoting the language accepted by the following DFA (8) b) Obtain the regular expression denoting the language accepted by the following DFA by using the formula R_{ij}^k



6. a) Show that every set accepted by a DFA is denoted by a regular Expression
- b) Construct an NFA equivalent to the following regular expression 01^*+1 .
7. a) Define a Regular set using pumping lemma Show that the language $L=\{0^i / i \text{ is an integer, } i \geq 1\}$ is not regular
- b) Construct an NFA equivalent to the regular expression $10+(0+11)0^*1$
8. a) Show that the set $L=\{0^{n^2/n} \text{ is an integer, } n \geq 1\}$ is not regular.
- b) Construct an NFA equivalent to the following regular expression $((10)^*(0+1)^*01 \cdot (10)^9$
- 9.a) Prove that if $L=L(A)$ for some DFA A, then there is a regular expression R such that $L=L(R)$.

b) Show that the language $\{0^p, p \text{ is prime}\}$ is not regular.

10. Find whether the following languages are regular or not.

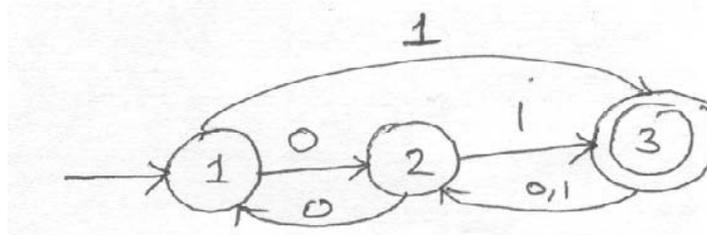
(i) $L = \{w \in \{a,b\}^R \mid w = w^R\}$.

(ii) $L = \{0^n 1^m 2^{n+m}, n, m \geq 1\}$

(iii) $L = \{1^k \mid k = n, n \geq 1\}$. (4)

(iv) $L_1/L_2 = \{x \mid \text{for some } y \in L_2, xy \in L_1\}$, where L_1 and L_2 are any two languages and L_1/L_2 is the quotient of L_1 and L_2 .

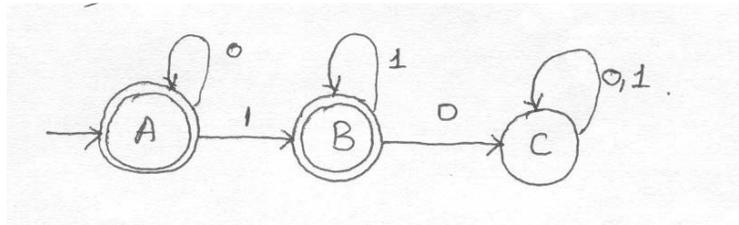
11.a) Find the regular expression for the set of all strings denoted by R^{13} from the deterministic finite automata given below:



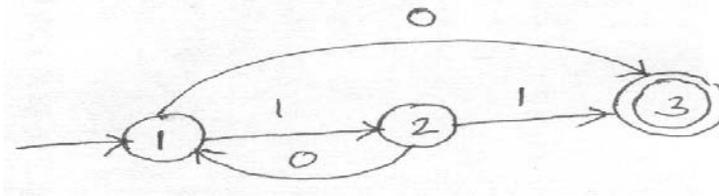
b) Verify whether the finite automata M_1 and M_2 given below are equivalent over $\{a,b\}$.

12.a) Construct transition diagram of a finite automaton corresponding to the regular expression $(ab+cb)^*$.

13.a) Find the regular expression corresponding to the finite automaton given below.



b) Find the regular expression for the set of all strings denoted by R^{23} from the deterministic finite automata given below.



14.a) Find whether the languages $\{ww, w \text{ is in } (1+0)^*\}$ and $\{1^k \mid k=n^2, n \geq 1\}$ are regular or not.

b) Show that the regular languages are closed under intersection and reversal.

UNIT III

PART-A

1. Define CFG. 2. Find $L(G)$ where $G = (\{S\}, \{0,1\}, \{S \rightarrow 0S1, S \rightarrow \epsilon\}, S)$.
2. Define derivation tree for a CFG (or) Define parse tree.
3. Construct the CFG for generating the language $L = \{a^n b^n \mid n \geq 1\}$.
4. Let G be the grammar $S \rightarrow aB/bA, A \rightarrow aA/bAA, B \rightarrow b/bS/aBB$. for the string $aaabbabbba$ find the left most derivation.
5. Let G be the grammar $S \rightarrow aB/bA, A \rightarrow aA/bAA, B \rightarrow b/bS/aBB$. obtain parse tree for the string $aaabbabbba$.
6. For the grammar $S \rightarrow aCa, C \rightarrow aCa/b$. Find $L(G)$.
7. Show that $id+id*id$ can be generated by two distinct leftmost derivation in the grammar $E \rightarrow E+E \mid E*E \mid (E) \mid id$.
8. For the grammar $S \rightarrow A1B, A \rightarrow 0A \mid \epsilon, B \rightarrow 0B \mid 1B \mid \epsilon$, give leftmost and rightmost derivations for the string 00101 .
9. Find the language generated by the CFG $G = (\{S\}, \{0,1\}, \{S \rightarrow 0/1/ \epsilon, S \rightarrow 0S0/1S1\}, S)$.
10. obtain the derivation tree for the grammar $G = (\{S,A\}, \{a,b\}, P, S)$ where P consist of $S \rightarrow aAS / a, A \rightarrow SbA / SS / ba$.
11. Consider the alphabet $\Sigma = \{a,b,(,),+,* ,., \epsilon\}$. Construct the context free grammar that generates all strings in Σ^* that are regular expression over the alphabet $\{a,b\}$.
12. Write the CFG to generate the set $\{a^m b^n c^p \mid m+n=p \text{ and } p \geq 1\}$.
13. Construct a derivation tree for the string 0011000 using the grammar $S \rightarrow A0S \mid 0 \mid SS, A \rightarrow S1A \mid 10$.
14. Give an example for a context free grammar.
15. Let the production of the grammar be $S \rightarrow 0B \mid 1A, A \rightarrow 0 \mid 0S \mid 1AA, B \rightarrow 11S \mid 0BB$. for the string 0110 find the right most derivation.
16. What is the disadvantages of unambiguous parse tree. Give an example.
17. Give an example of PDA.
18. Define the acceptance of a PDA by empty stack. Is it true that the language accepted by a PDA by empty stack or by that of final state are different

- languages.
20. What is additional feature PDA has when compared with NFA? Is PDA superior over NFA in the sense of language acceptance? Justify your answer.
 21. Explain what actions take place in the PDA by the transitions (moves) $\delta(q, a, Z) = \{(p_1, \gamma_1), (p_2, \gamma_2), \dots, (p_m, \gamma_m)\}$ and $\delta(q, \epsilon, Z) = \{(p_1, \gamma_1), (p_2, \gamma_2), \dots, (p_m, \gamma_m)\}$.
 22. What are the different ways in which a PDA accepts the language? Define them. Is it true that non deterministic PDA is more powerful than that of deterministic PDA? Justify your answer.
 23. Explain acceptance of PDA with empty stack.
 24. Is it true that deterministic push down automata and non deterministic push down automata are equivalent in the sense of language of acceptances? Justify your answer.
 25. Define instantaneous description of a PDA.
 26. Give the formal definition of a PDA.
 27. Define the languages generated by a PDA using final state of the PDA and empty stack of that PDA.
 28. Define the language generated by a PDA using the two methods of accepting a language.
 29. Define the language recognized by the PDA using empty stack.

PART-B

- 1.a) Let G be a CFG and let $a \Rightarrow w$ in G . Then show that there is a leftmost derivation of w .
 - b) Let $G = (V, T, P, S)$ be a Context free Grammar then prove that if $S \Rightarrow \alpha$ then there is a derivation tree in G with yield α .
2. Let G be a grammar $s \rightarrow OB/1A, A \rightarrow O/OS/1AA, B \rightarrow 1/1S/OBB$. For the string 00110101 find its leftmost derivation and derivation tree.
- 3) a) If G is the grammar $S \rightarrow Sbs/a$, Show that G is ambiguous.
 - b) Give a detailed description of ambiguity in Context free grammar
- 4.a) Show that $E \rightarrow E+E/E * E/(E)/id$ is ambiguous. (6) b) Construct a Context free grammar G which accepts $N(M)$, where $M = (\{q_0, q_1\}, \{a, b\}, \{z_0, z\}, \delta, q_0, z_0, \Phi)$ and where δ is given by

$$\delta(q_0, b, z_0) = \{(q_0, z z_0)\}$$

$$\delta(q_0, \epsilon, z_0) = \{(q_0, \epsilon)\}$$

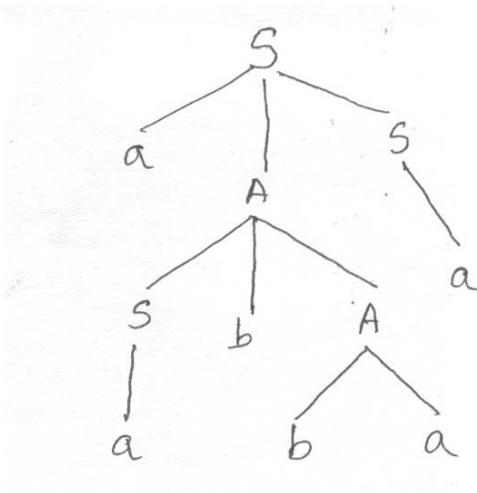
$$\delta(q_0, b, z) = \{(q_0, zz)\}$$

$$\delta(q_0, a, z) = \{(q_1, z)\}$$

$$\delta(q_1, b, z) = \{(q_1, \epsilon)\}$$

$$\delta(q_1, a, z_0) = \{(q_0, z_0)\}$$

- 5.a) If L is Context free language then prove that there exists PDA M such that $L = N(M)$.
- b) Explain different types of acceptance of a PDA. Are they equivalent in sense of language acceptance? Justify your answer.
6. Construct a PDA accepting $\{a^n b^m a^n \mid m, n \geq 1\}$ by empty stack. Also construct the corresponding context-free grammar accepting the same set.
- 7.a) Prove that L is $L(M_2)$ for some PDA M_2 if and only if L is $N(M_1)$ for some PDA M_1 .
- b) Define deterministic Push Down Automata DPDA. Is it true that DPDA and PDA are equivalent in the sense of language acceptance is concern? Justify Your answer.
- 8.a) Construct a equivalent grammar G in CNF for the grammar G_1 where $G_1 = (\{S, A, B\}, \{a, b\}, \{S \rightarrow bA/aB, A \rightarrow bAA/aS/a, B \rightarrow aBB/bS/b\}, S)$
- b) Find the left most and right most derivation corresponding to the tree.



9.a) Find the language generated by a grammar

$$G = (\{S\}, \{a, b\}, \{S \rightarrow aSb, S \rightarrow ab\}, S) \quad (4)$$

- b) Given $G = (\{S, A\}, \{a, b\}, P, S)$ where $P = \{S \rightarrow AaS|S|SS, A \rightarrow SbA|ba\}$
 S-Start symbol. Find the left most and right most derivation of the string $w = aabbbaaa$. Also construct the derivation tree for the string w .
 c) Define a PDA. Give an Example for a language accepted by PDA by empty stack.

10. G denotes the context-free grammar defined by the following rules. $S \rightarrow ASB/ab/SS$ $A \rightarrow aA/A$ $B \rightarrow bB/A$

- (i) Give a left most derivation of $aaabb$ in G. Draw the associated parse tree.
 (ii) Give a right most derivation of $aaabb$ in G. Draw the associated parse tree.
 (iii) Show that G is ambiguous. Explain with steps.
 (iv) Construct an unambiguous grammar equivalent to G. Explain.

11 a) Construct the grammar for the following PDA.

$M = (\{q_0, q_1\}, \{0, 1\}, \{X, z_0\}, \delta, q_0, Z_0, \Phi)$ and where δ is given by

$$\delta(q_0, 0, z_0) = \{(q_0, XZ_0)\}, \delta(q_0, 0, X) = \{(q_0, XX)\}, \delta(q_0, 1, X) = \{(q_1, \epsilon)\}, \\ \delta(q_1, 1, X) = \{(q_1, \epsilon)\}, \delta(q_1, \epsilon, X) = \{(q_1, \epsilon)\}, \delta(q_1, \epsilon, Z_0) = \{(q_1, \epsilon)\}. \quad (12)$$

b) Prove that if L is $N(M_1)$ for some PDA M_1 then L is $L(M_2)$ for some PDA M_2 .

12.a) Construct a PDA that recognizes the language

$$\{a^i b^j c^k \mid i, j, k > 0 \text{ and } i=j \text{ or } i=k\}.$$

b) Discuss about PDA acceptance

- (1) From empty Stack to final state.
- (2) From Final state to Empty Stack.

UNIT IV

PART-A

- 1 Define multitape Turing Machine.
- 2 Explain the Basic Turing Machine model and explain in one move. What are the actions that take place in TM?
- 3 Explain how a Turing Machine can be regarded as a computing device to compute integer functions.
- 4 Describe the non-deterministic Turing Machine model. Is it true that non-deterministic Turing Machine models are more powerful than the basic Turing Machines? (In the sense of language Acceptance).
- 5 Explain the multi-tape Turing Machine model. Is it more powerful than the basic Turing machine? Justify your answer.
- 6 Using Pumping lemma Show that the language $L = \{ a^n b^n c^n \mid n \geq 1 \}$ is not a CFL.
- 7 What is meant by a Turing Machine with two-way infinite tape.
- 8 Define instantaneous description of a Turing Machine.
- 9 What is the class of language for which the TM has both accepting and rejecting configurations? Can this be called a Context-free Language?
10. The binary equivalent of a positive integer is stored in a tape. Write the

necessary transition to multiply that integer by 2.

- 1 What is the role of checking off symbols in a Turing Machine?
- 2 State Pumping lemma for Context free language.
- 3 Define a Turing Machine.
- 4 Mention any two problems which can only be solved by TM.
- 5 State Pumping lemma and its advantages.
- 6 What are useless symbols in a grammar.

PART-B

1.a) Find a grammar in Chomsky Normal form equivalent to $S \rightarrow aAD; A \rightarrow aB/bAB; B \rightarrow b, D \rightarrow d$. (6)

b) Convert to Greibach Normal Form the grammar $G = (\{A1, A2, A3\}, \{a, b\}, P, A1)$ where P consists of the following.
 $A1 \rightarrow A2 A3, A2 \rightarrow A3 A1 / b, A3 \rightarrow A1 A2 / a$. (10)

2.a) Show that the language $\{0^n 1^{n^2} / n \geq 1\}$ is not a Context free language. (6)

b) Convert the grammar $S \rightarrow AB, A \rightarrow BS/b, B \rightarrow SA/a$ into Greibach Normal Form. (10)

3.a) Construct an equivalent grammar G in CNF for the grammar $G1$ where $G1 = (\{S, A, B\}, \{a, b\}, \{S \rightarrow bA/aB, A \rightarrow bAA/aS/a, B \rightarrow aBB/bS/b\}, S)$ (12)

b) Obtain the Chomsky Normal Form equivalent to the grammar $S \rightarrow bA/aB, A \rightarrow bAA/aS/a, B \rightarrow aBB/bS/b$. (4)

4.a) Begin with the grammar

$$S \rightarrow 0A0/1B1/BBA \rightarrow CB \rightarrow S/AC \rightarrow S/\epsilon$$

and simplify using the safe order Eliminate ϵ -Productions Eliminate unit production Eliminate useless symbols Put the (resultant) grammar in Chomsky Normal Form (10)

b) Let $G = (V, T, P, S)$ be a CFG. Show that if $S = \alpha$, then there is a derivation tree in a grammar G with yield α . (6)

5.a) Let G be the grammar $S \rightarrow aS/aSbS/\epsilon$. Prove that $L(G) = \{x / \text{each prefix of } x \text{ has at least as many a's as b's}\}$ (6)

b) Explain the Construction of an equivalent grammar in CNF for the grammar $G = (\{S, A, B\}, \{a, b\}, P, S)$

where $P = \{S \rightarrow bA/aB, A \rightarrow bAA/aS/a, B \rightarrow aBB/bS/b\}$ (10)

6.a) Find a Context free grammar with no useless symbol equivalent to

$S \rightarrow AB/CA, B \rightarrow BC/ABA \rightarrow a, C \rightarrow aB/b. (6)$

b) Show that any CFL without ϵ can be generated by an equivalent grammar in Chomsky Normal Form. (10)

7.a) Convert the following CFG to CNF $S \rightarrow ASA|aB, A \rightarrow BIS, B \rightarrow bl, \epsilon (12)$

b) Explain about Greibach Normal Form. (4)

8.a) Is $L = \{a^n b^n c^n / n \geq 1\}$ a context free language? Justify Your answer. (8)

b) Prove that for every context free language L without ϵ there exists an equivalent grammar in Greibach Normal Form. (8)

9. State and Prove pumping lemma for Context free languages. (16)

10.a) State Pumping Lemma for context free language. Show that $\{0^n 1^n 2^n / n \geq 1\}$ is not a Context free language. (6)

b) State Pumping lemma for context free language σ show that language $\{a^i b^j c^i d^j / i \geq 1, \text{ and } j \geq 1\}$ is not context-free. (6)

11.a) Design a Turing Machine M to implement the function “multiplication” using the subroutine ‘copy’. (12)

b) Explain how a Turing Machine with the multiple tracks of the tape can be used to determine the given number is prime or not. (4)

12.a) Design a Turing Machine to compute $f(m+n)=m+n, \forall m, n \geq 0$ and simulate their action on the input 0100. (10)

b) Describe the following Turing machine and their working. Are they more powerful than the Basic Turing Machine? Multi-tape Turing Machine Multi-Dimensional Turing Machine

(3) Non-Deterministic Turing Machine. (6)

13.a) Define Turing machine for computing $f(m,n)=m-n$ (proper subtraction).

(10)

b) Explain how the multiple tracks in a Turing Machine can be used for testing given positive integer is a prime or not. (6)

14.a) Explain in detail:” The Turing Machine as a Computer of integer functions”. (8)

b) Design a Turing Machine to accept the language $L = \{0^n 1^n / n \geq 1\}$ (8)

15.a) What is the role of checking off symbols in a Turing Machine? (4)

b) Construct a Turing Machine that recognizes the language $\{wcw/w \text{ in } \{a+b\}^+\}$ (12)

16. Prove that the language L is recognized by a Turing Machine with a two way infinite tape if and only if it is recognized by a Turing Machine with a one way infinite tape. (16)

17. For each of the following Context free languages L , find the smallest pumping length that will satisfy the statement of the Context free pumping lemma. In each case, Your answer should include a number (the minimum pumping length), a detailed explanation of why that the number is indeed a valid pumping length for the given language L , and a detailed explanation of why no smaller number qualifies as a valid pumping length for that particular language L .

(i) $L = \{a^n b^n / n \geq 0\}$ (6)

(ii) $L = \{w \text{ in } \{a,b\}^* / w \text{ has the same number of } a\text{'s and } b\text{'s}\}$ (6) (iii) $L = \{w \text{ in } \{a,b\}^* / w \text{ has twice as many } a\text{'s as } b\text{'s}\}$ (4)

18. Design a Turing Machine M that decides $A = \{0^k / n > 0 \text{ and } k = 2^n\}$ the language consisting of all strings of 0's whose length is a power of 2. (16)

19.a) Give a High level implementation description with a neat sketch of a Turing Machine M that performs the following computation. M on input w : writes a copy of w on the tape immediately after w , leaving the string $w\#w$ on the tape. Assume that the input string initially appears at the left most end of the tape and that the input alphabet does not contain the blank character ' ': The end of the input string is therefore determined by the location of the first blank cell on the input tape. The symbol $\#$ is assumed to be in the tape alphabet, and the input alphabet is $\{a,b\}$.

(12)

b) Demonstrate the working of your TM with an example. (4)

20.a) Show that the language $\{0^n 1^n 2^n / n \geq 1\}$ is not context free. (8)

b) Show that the context free languages are closed under union operation but not under intersection. (8)

UNIT-V

PART-A

- 1 When a recursively enumerable language is said to be recursive.
 - 2 Is it true that the language accepted by a non deterministic Turing Machine is different from recursively enumerable language?
 - 3 When we say a problem is decidable? Give an example of undecidable problem?
 - 4 Give two properties of recursively enumerable sets which are undecidable.
 - 5 Is it true that complement of a recursive language is recursive? Justify your answer.
 - 6 When a language is said to be recursive or recursively enumerable?
 - 7 When a language is said to be recursive? Is it true that every regular set is not recursive?
 - 8 When a problem is said to be decidable or undecidable? Give an example of an undecidable.
 - 9 What do you mean by universal Turing Machine?
10. When a problem is said to be undecidable? Give an example of an undecidable problem.
 11. Show that the union of recursive language is recursive.
 12. Show that the union of two recursively enumerable languages is recursively enumerable.
 13. What is undecidability problem?
 14. Show that the following problem is undecidable. "Given two CFG's G_1 and G_2 , is $L(G_1) \cap L(G_2) = \Phi$?".
 15. Define L_d .
 16. Define recursively enumerable language.
 17. Give an example for a non recursively enumerable language.
- 1 Differentiate between recursive and recursively enumerable languages.
 - 2 Mention any two undecidability properties for recursively enumerable language.
21. Define Diagonal languages.
 22. Give an example for an undecidable problem.

PART-B

- 1.a) Show that union of recursive languages is recursive.
(4)
- b) Define the language L_d and show that L_d is not recursively enumerable language. (8)

c) Explain the Halting problem. Is it decidable or undecidable problem
(4)

2. Define Universal language L_u . Show that L_u is recursively enumerable but not recursive.

3.a) Obtain the code for the TM
 $M = (\{q_1, q_2, q_3\}, \{0, 1\}, \{0, 1, B\}, \delta, q_1, B, \{q_2\})$ With the moves
 $\delta(q_1, 1) = (q_3, 0, R)$ $\delta(q_3, 0) = (q_1, 1, R)$ $\delta(q_3, 1) = (q_2, 0, R)$
 $\delta(q_3, B) = (q_3, 1, L)$ $\delta(q_2, B) = (q_3, 1, L)$

b) Show that L_n is recursively enumerable.

4.a) Define L_d and show that L_d is not recursively enumerable.
(12)

b) Whether the problem of determining given recursively enumerable language is empty or not? Is decidable? Justify your answer.

(4) 5. Define the language L_u . Check whether L_u is recursively enumerable? or L_u is recursive? Justify your answer. (16) 6.a) Show that the language L_d is neither recursive nor recursively enumerable. (12) b) Describe how a Turing Machine can be encoded with 0 and 1 and give an example. (4) 7.a) Show that any non trivial property J of the recursively enumerable

languages is undecidable. (8) b) Show that if L and L are recursively enumerable then L and L recursive.

8. Define the universal language and show that it is recursively enumerable but not recursive. (16)

9. Prove that the universal language L_u is recursively enumerable. (16)

10. State and Prove Rice's Theorem for recursive index sets. (16)

11.a) Show that the following language is not decidable. $L = \{ \langle M \rangle \mid M \text{ is a TM that accepts the string } aab \}$. (8)

b) Discuss the properties of Recursive and Recursively enumerable languages. (8)

12.a) Define Post correspondence problem with an example. (8)

b) Prove that the function $f(n) = 2^n$ does not grow at a polynomial rate, in other words, it

- does not satisfy $f(n) = O(n^p)$ for any finite exponent p .
- 13.a) Define the language L_d . Show that L_d is neither recursive nor recursively enumerable. (12)
- b) Show that if a language L and its complement \bar{L} are both recursively enumerable then L is recursive. (4)
- 14.a) What are the features of a Universal Turing Machine? (4)
- b) Show that "If a language L and its complement \bar{L} are both recursively enumerable, then both languages are recursive". (6)
- c) Show that halting problem of Turing Machine is undecidable. (6)

- 15.a) Does PCP with two lists $x = (b, bab, ba)$ and $y = (b, ba, a)$ have a solution? (6)
- b) Show that the characteristic function of the set of all even numbers is recursive. (6)
- c) Let $\Sigma = \{0, 1\}$. Let A and B be the lists of three strings each, defined as List A List B

i Wi Xi1 1 1112 10111 10310 0

Does this PCP have a solution? (4)

- 16.a) Show that it is undecidable for arbitrary CFG's G_1 and G_2 whether $L(G_1) \cap L(G_2)$ is a CFL. (8)
- b) Show that "finding whether the given CFG is ambiguous or not" is undecidable by reduction technique. (8)
17. Find whether the following languages are recursive or recursively enumerable.
- (i) Union of two recursive languages. (4)
- (ii) Union of two recursively enumerable languages. (4)
- (iii) L if L and complement of L are recursively enumerable. (4)
- (iv) L^u (4)

18. Consider the Turing Machine M and $w = 01$, where $M = (\{q_1, q_2, q_3\}, \{0, 1\}, \{0, 1, B\}, \delta, q_1, B, \{q_3\})$ and δ is given by
- Reduce the above problem to Post's correspondence Problem and find whether that PCP has a solution or not. (16)
- 2 Explain the Post's Correspondence Problem with an example (16)

3 Find the languages obtained from the following operations:

qi	$\delta(q_i, 0)$	$\delta(q_i, 1)$	$\delta(q_i, B)$
Q1	(q2,1,R)	(q2, 0,L)	(q2, 1,L)
q2	(q3, 0,L)	(q1, 0,R)	(q2, 0,R)
q3			

- (i) Union of two recursive languages. (6)
- (ii) Union of two recursively enumerable languages (6)
- (iii) L if L and complement of L are recursively enumerable (4)

V.S.B ENGINEERING COLLEGE, KARUR – 639 111
DEPARTMENT OF MATHEMATICS

IMPORTANT UNIVERSITY PROBLEMS FOR
MA 6566 - DISCRETE MATHEMATICS

UNIT I
LOGIC AND PROOFS
PART – A

1. Express the statement “Good food is not cheap” in symbolic form.

Solution : P : food is Good Q : food is Cheap.

Symbolic form is $P \rightarrow \neg Q$

2. Obtain PDNF for $\neg P \vee Q$.

Solution : PDNF is $(P \wedge Q) \vee (\neg P \wedge Q) \vee (\neg P \wedge \neg Q)$

3. If P, Q & R are statement variables, prove that $P \wedge ((\neg P \wedge Q) \vee (\neg P \wedge \neg Q)) \Rightarrow R$.

Solution : Consider $P \wedge ((\neg P \wedge Q) \vee (\neg P \wedge \neg Q)) \rightarrow R$

$$\Leftrightarrow (F \wedge Q) \vee (F \wedge \neg Q) \rightarrow R$$

$$\Leftrightarrow F \rightarrow R$$

$$\Leftrightarrow \neg F \vee R$$

$$\Leftrightarrow T \vee R$$

$$\Leftrightarrow T$$

4. Prove that whenever $A \wedge B \Rightarrow C$, we also have $A \Rightarrow (B \rightarrow C)$ and vice versa.

Proof:

To prove $A \Rightarrow (B \rightarrow C)$ we have to prove $A \rightarrow (B \rightarrow C)$ is a Tautology

$A \wedge B \rightarrow C$ is a Tautology

$$\Leftrightarrow \neg(A \wedge B) \vee C \text{ is a Tautology}$$

$$\Leftrightarrow \neg A \vee \neg B \vee C \text{ is a Tautology}$$

$$\Leftrightarrow \neg A \vee (B \rightarrow C) \text{ is a Tautology}$$

$$\Leftrightarrow A \rightarrow (B \rightarrow C) \text{ is a Tautology. Hence the proof.}$$

5. Define functionally complete set of connectives and give an example.

A collection of logical operators is called functionally complete if every compound proposition is logically equivalent to a compound proposition involving only those logical operators

6. Define Contra positive of a statement.

For any statement formula $P \rightarrow Q$, the statement formula $Q \rightarrow P$ is called its converse, $\neg P \rightarrow \neg Q$ is called its inverse and $\neg Q \rightarrow \neg P$ is called its contrapositive.

7. Give the converse and the Contra positive of the implication “ If it is raining then I get wet”.

Solution :

P : It is raining Q : I get wet

Converse : $Q \rightarrow P$: If I get wet, then it is raining.

Contrapositive : $\neg Q \rightarrow \neg P$: If I do not get wet, then it is not raining

8. Show that $\neg(P \wedge Q) \rightarrow (\neg P \vee (\neg P \vee Q)) \Leftrightarrow (\neg P \vee Q)$. (Use only the laws).

Solution :

$$\begin{aligned} & \neg(P \wedge Q) \rightarrow (\neg P \vee (\neg P \vee Q)) \\ \Leftrightarrow & \neg\neg(P \wedge Q) \vee (\neg P \vee (\neg P \vee Q)) \\ \Leftrightarrow & (P \wedge Q) \vee (\neg P \vee Q) \quad (\text{Associative law}) \\ \Leftrightarrow & (P \vee \neg P \vee Q) \wedge (Q \vee \neg P \vee Q) \quad (\text{Distributive law}) \\ \Leftrightarrow & (T \vee Q) \wedge (\neg P \vee Q) \quad (\text{Negation law}) \\ \Leftrightarrow & T \wedge (\neg P \vee Q) \quad (\text{Domination law}) \\ \Leftrightarrow & (\neg P \vee Q) \quad (\text{Identity law}) \end{aligned}$$

9. Write the following statement in symbolic form

- (a) Mark is poor but unhappy
 (b) mark is rich or unhappy
 (c) Mark is neither rich nor happy
 (d) Mark is poor (or) he is both rich and poor.

Solution :

M : Mark is poor H : Mark is happy

The symbolic forms are

(a) $M \wedge \neg H$ (b) $M \vee \neg H$ (c) $\neg M \vee \neg H$ (d) $M \vee (\neg M \wedge \neg H)$

10. Write the following statement in symbolic form

“x is the father of mother of y” .

Solution :

P(x) x is a person

F(x,y) : x is the father of y

M(x,y) : x is the mother of y

The symbolic form is $(\exists z)P(z) \wedge F(x, z) \wedge M(z, y)$

11. Write in symbolic form the statement “ The house will be destroyed if there is a flood”.

Solution :

P : House will be destroyed Q : There is flood

The symbolic form is $Q \rightarrow P$.

12. Construct the truth table for $(P \rightarrow Q) \wedge (Q \rightarrow P)$.

P	Q	$P \rightarrow Q$	$Q \rightarrow P$	$(P \rightarrow Q) \wedge (Q \rightarrow P)$
T	T	T	T	T
T	F	F	T	F
F	T	T	F	F
F	F	T	T	T

13. Write the dual of $P \nabla Q$.

Solution :

$$P \nabla Q \Leftrightarrow (P \wedge \neg Q) \vee (\neg P \wedge Q)$$

Dual of $P \nabla Q$ is $(P \vee \neg Q) \wedge (\neg P \vee Q)$

14. Represent $P \rightarrow Q$ using \uparrow only.

$$\begin{aligned} P \rightarrow Q &\Rightarrow \neg P \vee Q \\ &\Rightarrow \neg(P \wedge \neg Q) \\ &\Rightarrow P \uparrow \neg Q \end{aligned}$$

15. Write the converse, inverse and contra positive of the following
“If today is labour day, then tomorrow is Tuesday”.

Solution :

P : Today is Labour day Q : Tomorrow is Tuesday

Converse : $Q \rightarrow P$: If tomorrow is Tuesday then today is labour day.

Inverse : $\neg P \rightarrow \neg Q$: If today is not labour day then tomorrow is not Tuesday.

Contrapositive : $\neg Q \rightarrow \neg P$: If tomorrow is not Tuesday then today is not labour day

16. For any statements P, Q prove that $\neg(P \downarrow Q) \Leftrightarrow \neg P \uparrow \neg Q$; $\neg(P \downarrow Q) \Leftrightarrow \neg P \downarrow \neg Q$.

Solution :

$$\begin{aligned} \neg(P \downarrow Q) &\Leftrightarrow \neg(\neg(P \vee Q)) & \neg(P \uparrow Q) &\Leftrightarrow \neg(\neg(P \wedge Q)) \\ &\Leftrightarrow \neg(\neg P \wedge \neg Q) & &\Leftrightarrow \neg(\neg P \vee \neg Q) \\ &\Leftrightarrow \neg P \uparrow \neg Q & &\Leftrightarrow \neg P \downarrow \neg Q \end{aligned}$$

17. Determine the truth value of the following

a) If $3+4=12$, then $3+2=6$.

b) If $3+3=6$, then $3+4=9$.

a) Here $3+4=12$: F & $3+2=6$: F. ie $F \rightarrow F$ so truth value is True

b) Here $3+3=6$: T & $3+4=9$: F. ie $T \rightarrow F$ so truth value is False

18. Write the dual of (a) $Q \rightarrow P$ (b) $P \rightarrow (Q \wedge R)$ (c) $P \leftrightarrow Q$.

Solution :

$$\begin{array}{lll} \text{a) } Q \rightarrow P & \text{b) } P \rightarrow (Q \wedge R) & \text{c) } P \leftrightarrow Q \\ \Rightarrow \neg Q \vee P & \Rightarrow \neg P \vee (Q \vee R) & \Rightarrow (P \rightarrow Q) \wedge (Q \rightarrow P) \\ & & \Rightarrow (\neg P \vee Q) \wedge (\neg Q \vee P) \\ \text{Dual : } \neg Q \wedge P & \text{Dual : } \neg P \wedge (Q \vee R) & \text{Dual : } (\neg P \wedge Q) \vee (\neg Q \wedge P) \end{array}$$

19. Show that $\{\wedge, \vee\}, \{\vee\}$ & $\{\neg\}$ are not functionally complete set.

Solution :

$\neg P$ cannot be expressed using the connectives $\{\vee, \wedge\}$, since no such contribution of statement exist with $\{\vee, \wedge\}$ as input is T and the output is F.

20. Express $P \uparrow Q$ interms of \downarrow only.

Solution :

$$\begin{aligned} P \uparrow Q &\Leftrightarrow \neg(P \wedge Q) \\ &\Leftrightarrow \neg P \vee \neg Q \Leftrightarrow (P \downarrow P) \vee (Q \downarrow Q) \Leftrightarrow (P \downarrow P) \downarrow (Q \downarrow Q) \end{aligned}$$

21. Show that $P \rightarrow (Q \rightarrow R) \Leftrightarrow P \rightarrow (\neg Q \vee R) \Leftrightarrow (P \wedge Q) \rightarrow R$.

$$\begin{aligned} P \rightarrow (Q \rightarrow R) &\Leftrightarrow P \rightarrow (\neg Q \vee R) \\ &\Leftrightarrow \neg P \vee \neg Q \vee R \Leftrightarrow \neg(P \wedge Q) \vee R \Leftrightarrow (P \wedge Q) \rightarrow R \end{aligned}$$

22. Demonstrate that R is a valid inference from the premises $P \rightarrow Q, Q \rightarrow R$ & P .

Step	Derivation	Rule
(1)	$P \rightarrow Q$	P
(2)	$Q \rightarrow R$	P
(1,2)	$P \rightarrow R$	T
(3)	P	P
(1,2,3)	R	T

23. Show that $\neg Q, P \rightarrow Q \Rightarrow \neg P$.

Step	Derivation	Rule
(1)	$\neg Q$	P
(2)	$P \rightarrow Q$	P
(1,2)	$\neg Q \rightarrow \neg P$	T
(1,2)	$\neg P$	T

24. Show that a) $P \vee (P \wedge Q) \Rightarrow P$ b) $P \vee (\neg P \wedge Q) \Leftrightarrow P \vee Q$.

$$\begin{aligned} \text{a) } P \vee (P \wedge Q) \rightarrow P &\Leftrightarrow [\neg(P \vee (P \wedge Q))] \vee P \Leftrightarrow [\neg P \vee \neg(P \wedge Q)] \vee P \\ &\Leftrightarrow [\neg P \vee (\neg P \vee \neg Q)] \vee P \Leftrightarrow (\neg P \wedge \neg P) \vee (\neg P \wedge \neg Q) \vee P \\ &\Leftrightarrow \neg P \vee (\neg P \wedge \neg Q) \vee P \Leftrightarrow \neg P \vee P \vee (\neg P \wedge \neg Q) \\ &\Leftrightarrow T \vee (\neg P \wedge \neg Q) \Leftrightarrow T \end{aligned}$$

$$\text{b) } P \vee (\neg P \wedge Q) \Leftrightarrow (P \vee \neg P) \wedge (P \vee Q) \Leftrightarrow T \wedge (P \vee Q) \Leftrightarrow (P \vee Q)$$

25. Give an example to show that $(\exists x)(A(x) \wedge B(x))$ need not be a conclusion from $(\exists x)A(x)$ & $(\exists x)B(x)$.

Solution :

Let $A(x): x \in A$. Similarly $B(x)$ is defined.

Let $A = \{1\}$ and $B = \{2\}$.

Since A and B are non-empty $(\exists x)A(x)$ and $(\exists x)B(x)$ is True.

But $(\exists x)(A(x) \wedge B(x))$ is False.

26. Let $p(x)$ denote the statement “ $x > 4$ ”. What are the truth values of $P(5)$ and $P(2)$?

Solution :

We obtain the statement $P(5)$ by setting $x=5$ in the statement “ $x > 4$ ”. Hence $P(5)$, which is the statement “ $5 > 4$ ” is true. However $P(2)$, which is the statement “ $2 > 4$ ” is false.

27. Let $Q(x,y)$ denote the statement “ $x=y+2$ ”, what are the truth values of the propositions $Q(1,2)$ and $Q(2,0)$.

Solution :

To obtain $Q(1,2)$, set $x=1$ and $y=2$ in the statement $Q(x,y)$.

Hence $Q(1,2)$ is the statement “ $1=2+2$ ” which is false.

Similarly for $Q(2,0)$, the statement is “ $2=0+2$ ” which is True.

28. What are the truth values of the proposition $R(1,2,3)$ and $R(0,0,1)$?

Solution :

The proposition $R(1,2,3)$ is obtained by setting $x=1$, $y=2$ and $z=3$ in the statement $R(x,y,z)$ which denote “ $x+y=z$ ”.

The proposition $R(1,2,3)$ is “ $1+2=3$ ” which is True.

The proposition $R(0,0,1)$ is “ $0+0=1$ ” which is False.

29. Find the truth value of $(x)(P \rightarrow Q(x)) \vee (\exists x)R(x)$ where

$P : 2 > 1, Q(x) : x > 3, R(x) : x > 4$ with universe of discourse E being $E=\{2,3,4\}$.

Solution :

$P : 2 > 1$ is True , $Q(4)$ is False. Therefore $(x)(P \rightarrow Q(x))$ is False.

Since $R(2), R(3), R(4)$ are all False, $(\exists x)R(x)$ is also False.

Hence $(x)(P \rightarrow Q(x)) \vee (\exists x)R(x)$ is also False.

30. Express the statement “ For every x there exist a y such that $x^2 + y^2 \geq 100$ ” in symbolic form.

The symbolic form is $(\forall x)(\exists y)(x^2 + y^2 \geq 100)$

31. Define Simple statement function. (OR)

Define statement function of one variable. When it will become a statement?

A simple statement function of a variable is defined to be an expression consisting of a predicate symbol and an individual variable. Such a statement function becomes a statement when the variable is replaced by the name of any object.

Example : If “ X is a Teacher” is denoted by $T(x)$, it is a statement function. If X is replaced by John, then “John is a teacher”.

32. Give the symbolic form of the statement

“Every book with a blue cover is a Mathematics book”.

The symbolic form is $(\forall x)(S(x)) \rightarrow P(x)$

where $S(x) : x$ is every book with a blue cover

$P(x) : Mathematics book.$

33. Define Quantifiers.

Certain declarative sentences involve words that indicate quantity such as “all, some, none, one”.

These words help to determine the answer to the question “How many?” Since such words indicate quantity they are called quantifiers.

34. Write the following sentence in a symbolic form “Every one who is healthy can do all kinds of work”.

$H(x) : x$ is a healthy person

$H(y) : y$ is a kind of work

$D(x,y) : x$ can do y

$$(x)(y)[H(x) \wedge H(y) \rightarrow D(x,y)]$$

35. Symbolize the following statement with and without using the set of positive integer on the universe of discourse

“Given any positive integers , there is a greater positive integer”.

Solution:

Let the variable x and y be restricted to the set of positive integers. Then the above statement can be Paraphrased as follows:

For all x, there exist a y such that y is greater than x. If $G(x,y)$ is “x is greater than y”, then the given statement is $(x)(\exists y)G(y,x)$.

If we do not impose the restriction on the universe of discourse and if we write $P(x)$ for “x is a positive integer”, then we can symbolize the given statement is $(x)(P(x) \rightarrow (y)(P(y) \wedge G(y,x)))$

36. Rewrite the following using quantifiers. “Some men are genius”

$M(x)$: x is a man $G(x)$: x is genius

$$(\exists x)(M(x) \wedge G(x))$$

37. Symbolize the expression “ All the world loves a lover”

In first note that the quotation really means that everybody loves a lover. Now

Let $P(x)$: x is a person $L(x)$: x is a lover $R(x,y)$: x loves y

The required expression is $(x)(P(x)) \rightarrow (y)(P(y) \wedge (L(y) \rightarrow R(x,y)))$

38. Identify the bound variables and the free variables in each of the following expressions

(a) $(x)(\exists z)(\cos(x+y)) = \sin(z-x)$. (b) $(\exists x)(\exists y)(x^2 - y^2 = z)$.

Solution:

In (a) the scope of $(x)(\exists z)$ is $(\cos(x+y))$, while the occurrence of y is a free occurrence and $\sin(z-x)$ is free.

In (b) the scope of $(\exists x)(\exists y)$ is $x^2 - y^2$, while the occurrence of z is a free occurrence.

39. Use quantifiers to express the associate law for multiplication of real numbers.

Solution: $\forall x \forall y \forall z ((x \cdot y) \cdot z = x \cdot (y \cdot z))$ where the universe of discourse for x,y,z is the set of real numbers.

40. Let the universe of discourse be $E=\{5,6,7\}$. Let $A=\{5,6\}$ and $B=\{6,7\}$. Let $P(x)$: x is in A; $Q(x)$: x is in B and $R(x,y)$: $x+y < 12$. Find the truth value of $((\exists x)(P(x) \rightarrow Q(x))) \rightarrow R(5,6)$.

Solution: $R(x,y)$: $x+y < 12$ the only possibility is

$$5 + 6 < 12$$

$\therefore x(P(x) \rightarrow Q(x))$ is true.

$$\therefore ((\exists x)(P(x) \rightarrow Q(x))) \rightarrow R(5,6)$$

41. Give an example in which $(\exists x)(P(x) \rightarrow Q(x))$ is true but $((\exists x)P(x)) \rightarrow ((\exists x)Q(x))$ is false.

Solution : Let $E = \{2,3,5\}$

Let $P(x)$: $x < 4$, $Q(x)$: $x > 6$

$P(2)$ is true. $\therefore (\exists x)P(x)$ is true.

For any x in E, $Q(x)$ is false.

Hence $((\exists x)P(x)) \rightarrow ((\exists x)Q(x))$ is false.

$P(5)$ is false and $Q(5)$ is false.

$\therefore P(5) \rightarrow Q(5)$ is true.

$\therefore (\exists x)(P(x) \rightarrow Q(x))$ is true.

42. Construct a truth table for $(p \rightarrow q) \rightarrow (q \rightarrow p)$

p	q	$p \rightarrow q$	$q \rightarrow p$	$(p \rightarrow q) \rightarrow (q \rightarrow p)$
T	T	T	T	T
T	F	F	T	T
F	T	T	F	F
F	F	T	T	T

UNIT-I PART-B

1. Prove that $(P \rightarrow Q) \wedge (Q \rightarrow R) \Rightarrow (P \rightarrow R)$.
2. Find the PDNF and PCNF of the formula $S \Leftrightarrow (P \rightarrow (Q \wedge R)) \wedge (\neg P \rightarrow (\neg Q \wedge \neg R))$.
3. Using Conditional proof, prove that $\neg P \vee Q, \neg Q \vee R, R \rightarrow S \Rightarrow P \rightarrow S$.
4. By using truth table verify whether the following specifications are Consistent.
Whenever the system software is being upgraded users cannot access the file system. If users can access file system, then they can save new files. If users cannot save new files then the system software is not being upgraded.
5. Without using truth tables, show that $\neg P \wedge (\neg Q \wedge R) \vee (Q \wedge R) \vee (P \wedge R) \Leftrightarrow R$.
6. Obtain PDNF & PCNF of $(\neg P \rightarrow R) \wedge (Q \leftrightarrow P)$.
7. Show that S is a valid inference from the premises $P \rightarrow \neg Q, Q \vee R, \neg S \rightarrow P \ \& \ \neg R$.
8. Without using truth tables, show that $\neg P \wedge (\neg Q \wedge R) \vee (Q \wedge R) \vee (P \wedge R) \Leftrightarrow R$.
9. Obtain PDNF & PCNF of $(\neg P \rightarrow R) \wedge (Q \leftrightarrow P)$
10. Show that $R \rightarrow S$ is a valid inference from the set of premises $P \rightarrow (Q \rightarrow S), \neg R \vee P \ \& \ Q$.
11. Show that the following premises are inconsistent:
 1. If Jack misses many classes through illness and reads a lot of books.
 2. If Jack fails high school, then he is uneducated.
 3. If Jack reads a lot of books, then he is not uneducated.
 4. Jack misses many classes through illness and reads a lot of book.
12. "If there was a ball game, then traveling was difficult. If arrived on time, then travelling was difficult. If they arrived on time, then traveling was not difficult. They arrived on time. Therefore there was no ball game". Show that these statements constitute a valid argument.
13. Show the validity of the argument:
If the band could not play rock music (or) the refreshments were not delivered on time, then the new year's party would have been cancelled and Alice would have angry. If the party were cancelled, then refunds would have to be made. No refunds were made. Therefore the band could play rock music.
14. Establish the validity of the argument.
 $U \rightarrow R, (R \wedge S) \rightarrow (P \vee T), (Q \rightarrow (U \wedge S)), \neg T \Rightarrow Q \rightarrow P$.
15. Show that without using truth table
 $((P \vee Q) \wedge \neg(\neg P \wedge (\neg Q \vee \neg R))) \vee (\neg P \vee \neg Q) \vee (\neg P \wedge \neg R)$ is a tautology

16. Establish the validity of the argument

$$P \rightarrow Q, Q \rightarrow (R \wedge S), \neg R \vee (\neg T \vee U), P \wedge T \Rightarrow U.$$

17. Establish the validity of the argument

$$P, P \vee Q, Q \rightarrow (R \rightarrow S), T \rightarrow R \Rightarrow \neg S \rightarrow \neg T.$$

18. Establish the validity of the argument

$$(\neg P \vee Q) \rightarrow R, R \rightarrow (S \vee T), \neg S \wedge \neg U, \neg U \rightarrow \neg T \Rightarrow P.$$

19. Establish the validity of the argument $\neg P \leftrightarrow Q, Q \rightarrow R, \neg R \Rightarrow P$.

20. Use Indirect method of proof show that $(x)(P(x) \vee Q(x)) \Rightarrow (x)P(x) \vee (\exists x)Q(x)$.

21. Prove that $(\exists x)P(x) \rightarrow (x)Q(x) \Rightarrow (x)(P(x) \rightarrow Q(x))$.

22. Use Conditional proof to prove that $(x)(P(x) \rightarrow Q(x)) \Rightarrow (x)P(x) \rightarrow (x)Q(x)$.

23. Prove that $(\exists x)(A(x) \vee B(x)) \Leftrightarrow (\exists x)A(x) \vee (\exists x)B(x)$.

24. Show that $(x)(P(x) \rightarrow Q(x)) \wedge (x)(Q(x) \rightarrow R(x)) \Rightarrow (x)(P(x) \rightarrow R(x))$.

25. Is the following conclusion validly derivable from the premises given?

$$\text{If } (x)(P(x) \rightarrow Q(x)), (\exists y)P(y) \text{ then } (\exists z)Q(z).$$

26. Verify the validity of the inference: If one person is more successful than another, then he has worked harder to deserve success. John not worked harder than Peter. Therefore John is not successful than Peter.

27. Prove that $(\exists x)(P(x) \wedge Q(x)) \Rightarrow (\exists x)P(x) \wedge (\exists x)Q(x)$. Verify Whether the converse is true.

28. Show that $(\exists x)(F(x) \wedge S(x)) \rightarrow (y)(M(y) \rightarrow W(y)), (\exists y)(M(y) \wedge \neg W(y)) \Rightarrow (x)(F(x) \rightarrow \neg S(x))$.

29. No junior (or) senior is enrolled in a physical education class. Mary is enrolled in a physical education class. Therefore Mary is

30. Show that $(x)(P(x) \vee Q(x)), (x)(\neg P(x) \wedge Q(x)) \rightarrow R(x) \Rightarrow (x)(\neg R(x) \rightarrow P(x))$.

31. Express “ $\sqrt{2}$ is an irrational number” using quantifiers.

32. Using CP or otherwise, show that the following implications

(a) $(\exists x)P(x) \rightarrow (x)Q(x) \Rightarrow (x)(P(x) \rightarrow Q(x))$ (b) $(x)(P(x) \rightarrow Q(x)) \Rightarrow (x)P(x) \rightarrow (x)Q(x)$.

UNIT-II
COMBINATORICS
PART A

1. Define the product rule.

If one job can be done in m ways and following this another job can be done in n ways then the total number of ways in which both the jobs can be done in the stated order is mn .

2. Define the sum rule.

If one job can be done in m ways and another job can be done in n ways and if there is no way common to both jobs then the total number of ways in which either of the two jobs can be done is equal to $m + n$.

3. How many different 8-bit strings are there that begin and end with one.

Solution: A 8-bit string that begins and ends with 1 can be constructed in 6 steps, (i.e.,) By selecting Ind bit, IIIrd bit, Vth bit, VIIth bit and each bit can be selected in 2 ways.

Hence, the total number of 8-bit strings that begins and end with 1 is equal to $2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 = 2^6 = 64$.

4. How many different 8-bit strings are there that end with 0111?

Solution: A 8-bit strings that end with 0111 can be constructed in 4 steps.

By selection Ist bit, IInd bit, IIIrd bit and IVth bit and each bit can be selected in 2 Ways.

The total number of 8-bit strings that end with 0111 is equal to $2 \cdot 2 \cdot 2 \cdot 2 = 2^4 = 16$.

5. What is Inclusion and Exclusion principle.

Let P_1, P_2, \dots, P_n are finite sets.

$$\text{Then } |P_1 \cup P_2 \cup \dots \cup P_n| = \sum_{1 \leq i \leq n} |P_i| - \sum_{1 \leq i < j \leq n} |P_i \cap P_j| + \sum_{1 \leq i < j < k \leq n} |P_i \cap P_j \cap P_k| - \dots + (-1)^{n-1} |P_1 \cap P_2 \cap \dots \cap P_n|$$

6. Define Pigeonhole principle.

It states that if there are more pigeons (objects) than the pigeonholes (boxes), then some pigeonhole (box) must contain two or more pigeons (objects). The pigeonhole principle is also called the Dirichlet drawer principle or shoe box principle.

7. Give two examples based on pigeonhole principle.

Solution:

1. Among any group of 365 people, there must be at least two with the same birthday, because there are only 366 maximum possible birthdays.

2. In any group of 27 English words, there must be at least two that start with the same letter, since there are 26 letters in the English alphabet.

8. Show that among 13 children, there are at least two children who were born in the same month.

Solution: Let us assume that 13 children are pigeons and the 12 months (January, ..., December) are the pigeonholes. Then by the pigeonhole principle there will be at least two children who were born in the same month.

9. Prove the statement: If $m = K_{n+1}$ pigeons (where $K \geq 1$) occupy n pigeonholes then at least one pigeonhole must contain $K+1$ or more pigeons.

Solution: Let us assume that the conclusion of the given statement is false.

Then every pigeonhole contains K or less number of pigeons. Then, the total number of pigeons would be nK . This is a contradiction. Hence, the assumption made is wrong, and the given statement is true.

10. Show that if seven colors are used to paint 50 cars, at least eight cars will have the same colour.

Solution: Assume that 50 cars (pigeons) are assigned 7 colors (pigeonholes). Hence, by the generalised pigeonhole principle, at least $\left\lceil \frac{50-1}{7} \right\rceil + 1 = 8$ cars will have the same colour.

11. Show that if any 5 numbers from 1 to 8 are chosen, then two of them will have their sum equal to 9.

Solution: let us consider the following sets :

$$A_1 = \{1, 8\}, A_2 = \{2, 7\}, A_3 = \{3, 6\}, A_4 = \{4, 5\}$$

These are the only sets containing two numbers from 1 to 8, whose sum is 9.

Because every number from 1 to 8 belongs to one of the above sets, each of the 5 numbers chosen must belong to one of the sets.

Since there are only 4 sets, two of the 5 chosen numbers have to belong to the same set (by the pigeonhole principle). These two numbers have their sum equal to 9.

12. Define permutation:

A permutation of a set of distinct objects is an ordered arrangement of these objects.

Note: Permutation means selection and arrangement of factors.

Notation: nP_r (or) $P(n, r)$ (or) $P_{n,r}$ (or) P_n^r (or) $(n)_r$

13. Define r-permutation.

An r-permutation of n (distinct) elements x_1, x_2, \dots, x_n is an ordering of an r-elements subset $\{x_1, x_2, \dots, x_n\}$. The number of r-permutations of a set of n distinct elements is denoted by $P(n, r)$.

14. Define combinations and give an example.

A combination is a selection of objects without regard to order. (or) A combination is an unordered collection of distinct objects.

Example: abc is the combination of three objects a, b and c.

15. Prove that for let n and r be the non-negative integers with $r \leq n$. Then $C(n, r) = C(n, n-r)$.

Proof: We know that $C(n, r) = \frac{n!}{r!(n-r)!}$

$$C(n, n-r) = \frac{n!}{(n-r)!(n-(n-r))!} = \frac{n!}{(n-r)!(r!)} = C(n, r)$$

Hence, $C(n, r) = C(n, n-r)$.

16. Determine the value of n if $20C_{n+1} = 20C_{2n-1}$.

Solution: Given: $20C_{n+1} = 20C_{2n-1}$

Formula $nC_x = nC_y \Rightarrow n = x + y$ or $x = y$

$$n+1 = 2n-1 \Rightarrow 3 = n \Rightarrow n = 3$$

17. How many possibilities are there for the win, place and show (first, second and third) positions in a horse race with 12 horses if all orders of finish are possible?

Solution: The number of ways to pick the three winners is the number of ordered selections of three elements from 12. (i.e.,) $P(12, 3) = (12)(11)(10) = 1320$.

18. Determine the value of n if $(nP_3) = (n+1)P_3$

Solution: Given:

$$(4)(nP_3) = (n+1)P_3$$

$$(4) \left[\frac{n!}{(n-3)!} \right] = \frac{(n+1)!}{(n+1-3)!} = \frac{(n!)(n+1)}{(n-3)!(n-2)!}$$

$$\Rightarrow 4 = \frac{n+1}{n-2} \Rightarrow 4(n-2) = n+1 \Rightarrow 4n-8 = n+1$$

$$\Rightarrow 3n=9 \quad \Rightarrow n=3$$

19. In how many ways can a set of five letter to be selected from the English alphabet?

Solution: The positions of r 1s in a bit string of length n from an r -combination of the set $\{1, 2, 3, \dots, n\}$.

Hence, there are $C(n, r)$ bit strings of length n that contain exactly r 1s.

20. Define Recurrence Relations.

Recurrence Relation:(Sometimes called difference equation).

A recurrence relation for the sequence $\{a_n\}$ is an equation that shows a_n in terms of one or more of the previous terms of the sequence a_0, a_1, \dots, a_{n-1} , for all integers n with $n \geq n_0$, where n_0 is a non-negative integer.

21. Let $\{A_n\}$ be a sequence that satisfies the recurrence relation $a_n = a_{n-2} + a_{n-1}$ for $n=1, 2, 3, 4, 5, \dots$ and suppose that $a_0 = 3$ and $a_1 = 5$. What are a_2 and a_3 ?

Solution: Given

$$a_n = a_{n-2} + a_{n-1}$$

$$a_2 = a_0 + a_1 = 3 + 5 = 8$$

$$a_3 = a_1 + a_2 = 5 + 8 = 13$$

22. Let $a_n = 2^n + (5)(3^n)$ for $n=0, 1, 2, \dots$

(a) Find a_0, a_1 and a_2 . (b) Show that $a_4 = 5a_3 - 6a_2$.

Solution: Given: $a_n = 2^n + (5)(3^n)$

$$a_0 = 2^0 + (5)(3^0) = 1 + (5)(1) = 6$$

$$a_1 = 2^1 + (5)(3^1) = 2 + (5)(3) = 17$$

$$a_2 = 2^2 + (5)(3^2) = 4 + (5)(9) = 49$$

(b) Given: $a_n = 2^n + 5(3^n)$

$$a_3 = 2^3 + 5(3^3) = 8 + 5(27) = 143$$

To prove: $a_4 = 5a_3 - 6a_2$

$$L.H.S = a_4 = 2^4 + 5(3^4) = 16 + 5(81) = 421$$

$$R.H.S = 5a_3 - 6a_2 = 5(143) - 6(49) = 715 - 294 = 421$$

$$L.H.S = R.H.S.$$

Hence the proof.

23. What are the three methods to solve recurrence relation?

1. Iteration, 2. Characteristic roots and 3. Generating functions.

24. What are the solution of the recurrence relation $a_n = 2a_{n-1}$ for $n \geq 1, a_0 = 3$

Solution: Given : $a_n = 2a_{n-1}$

$$\text{(i.e.,)} a_n - 2a_{n-1} = 0 \quad \dots\dots\dots (1)$$

Let $a_n = r^n$ be a solution of (1)

$$\Rightarrow r^n - 2r^{n-1} = 0 \Rightarrow r^n \left[1 - \frac{2}{r} \right] = 0 \Rightarrow r^n \left[\frac{r-2}{r} \right] = 0$$

The characteristic equation is $r-2=0$

$$r=2$$

$$\text{By theorem } a_n = \alpha 2^n \quad \dots\dots\dots(2)$$

$$a_0 = 3 \Rightarrow a_0 = \alpha 2^0 = 3$$

$$\text{Given } \Rightarrow \alpha = 3$$

$$(2) \Rightarrow a_n = 3(2^n)$$

25. Define Generating functions.

The generating function for the sequence $a_0, a_1, \dots, a_k, \dots$ of real numbers is the infinite series.

$$G(x) = a_0 + a_1x + \dots + a_kx^k + \dots = \sum_{k=0}^{\infty} a_kx^k.$$

26. Find the generating function for the finite sequence 2,2,2,2,2.

Solution: The generating function of 2,2,2,2,2. is $2 + 2x + 2x^2 + 2x^3 + 2x^4$

$$= 2 \left[1 + x + x^2 + x^3 + x^4 \right] = 2 \left[\frac{x^5 - 1}{x - 1} \right]$$

When $x \neq 1$. Consequently, $G(x) = 2 \left[\frac{x^5 - 1}{x - 1} \right]$ is the G.F. of the sequence 2,2,2,2,2.

27. Find the generating function for $(1+x)^{-n}$, where n is a positive integer.

Solution: $(1+x)^{-n} = \sum_{k=0}^n \binom{-n}{k} x^k$ by extended binomial theorem.

We know that $\binom{-n}{r} = (-1)^r c(n+r-1, r)$

$$(1+x)^{-n} = \sum_{k=0}^n (-1)^k c(n+k-1, k) x^k.$$

28. Find a closed form for the generating function of 3,-3,3,-3,3,-3,....

Solution: We have $\frac{3}{1+x} = 3(1+x)^{-1} = 3(1-x+x^2-x^3+\dots)$

$$= 3 + (-3)x + 3x^2 + (-3)x^3 + \dots = \sum_{n=0}^{\infty} (-3)^n x^n$$

Hence the required G.F is $\frac{3}{1+x}$

29. Find the co-efficient of x^{10} in $(1+x^5+x^{10}+x^{15}+\dots)^3$

Solution: We know that $(1+x^5+x^{10}+x^{15}+\dots)^3 = \left[(1-x^5)^{-1} \right]^3 = (1-x^5)^{-3} = \sum c(3+r-1, r) x^{5r}$

To find the coefficient of x^{10} , put $5r=10 \Rightarrow r=2$

The required coefficient is $c(3+2-1, 2) = c(4, 2)$

$$=4c_2 = 6$$

30. Define Inclusion and Exclusion.

Let X and Y be two finite subsets of a universal set U. If X and Y are disjoint, then

$$|X \cup Y| = |X| + |Y|. \text{ If X and Y are not disjoint then } |X \cup Y| = |X| + |Y| - |X \cap Y|$$

This is called the principle of inclusion and exclusion.

31. Give a formula for the number of elements in the union of four sets.

Solution: By the principle of the inclusion and exclusion we get

$$\begin{aligned} |A_1 \cup A_2 \cup A_3 \cup A_4| &= |A_1| + |A_2| + |A_3| + |A_4| - |A_1 \cap A_2| \\ &\quad - |A_1 \cap A_3| - |A_1 \cap A_4| - |A_2 \cap A_3| - |A_2 \cap A_4| - |A_3 \cap A_4| \\ &\quad + |A_1 \cap A_2 \cap A_3| + |A_1 \cap A_2 \cap A_4| + |A_1 \cap A_3 \cap A_4| \\ &\quad + |A_2 \cap A_3 \cap A_4| - |A_1 \cap A_2 \cap A_3 \cap A_4| \end{aligned}$$

32. Find a formula for the probability of the union of n events in a sample space.

Solution: the probability of n events in a sample space is

$$P\left(\bigcup_{i=1}^n E_i\right) = \sum_{1 \leq i \leq n} P(E_i) - \sum_{1 \leq i < j \leq n} P(E_i \cap E_j) + \sum_{1 \leq i < j < k \leq n} P(E_i \cap E_j \cap E_k) - \dots + (-1)^{n+1} P\left(\bigcap_{i=1}^n E_i\right)$$

UNIT II PART - B

1. Prove, by mathematics induction, that for all $n \geq 1$, $n^3 + 2n$ is a multiple of 3.
2. Use Mathematics induction to prove the inequality $n < 2^n$ for all positive integer n
3. Prove by the principle of mathematical induction, for 'n' a positive integer,

$$1^2 + 2^2 + 3^2 + \dots + n^2 = \frac{n(n+1)(2n+1)}{6}$$
4. Use Mathematical induction to show that $\frac{1}{\sqrt{1}} + \frac{1}{\sqrt{2}} + \dots + \frac{1}{\sqrt{n}} > \sqrt{n}$, $n \geq 2$
5. Using mathematical induction, prove that $1^2 + 2^2 + 3^2 + \dots + (2n-1)^2 = \frac{n(2n-1)(2n+1)}{3}$
6. Prove by mathematical induction that $6^{n+2} + 7^{2n+1}$ is divisible by 43 for each positive integer n.
7. Prove that in a group of six people, atleast three must be mutual friends or atleast three must be mutual strangers.
8. From a club consisting of six men and seven women, in how many ways we select a committee of
 - (1) 3 men and four women?
 - (2) 4 person which has at least one women?
 - (3) 4 person that has atmost one man?
 - (4) 4 persons that has children of both sexes?
9. How many bits of string of length 10 contain
 - (i) exactly four 1's
 - (ii) atmost four 1's
 - (iii) atleast four 1's
 - (iv) an equal number of 0's and 1's
10. Find the number of distinct permutations that can be formed from all the letters of each word (1) RADAR (2) UNUSUAL.
11. Suppose that there are 9 faculty members in the mathematics department and 11 in the computer science department. How many ways are there to select a committee to develop a discrete mathematics course at a school if the committee is to consist of

three faculty members form the mathematics department and four from the computer science department?

12. If n Pigeonholes are occupied by $(kn+1)$ pigeons, where k is positive integer, prove that at least one pigeonhole is occupied by $k+1$ or more pigeons. Hence, find the minimum number of m integers to be selected from $S=\{1,2,\dots,9\}$ so that the sum of the m integers are even.
13. How many positive integers n can be formed using the digits 3,4,4,5,5,6,7 if n has to exceed 5000000?
14. Find the number of integers between 1 and 250 both inclusive that are divisible by any of the integers 2,3,5,7.
15. There are six men and five women in a room. Find the number of ways four person can be drawn from the room if (i) they can be male or female, (ii) two must be men and women, (iii) they must all are of the same sex.
16. What is the maximum number of students required in a discrete mathematics class to be sure that at least six will receive the same grade if there are five possible grades A, B,C,D and E?
17. Find the number of positive integers ≤ 1000 and not divisible by any of 3,5,7 and 22.
18. A box contains six white balls and five red balls. Find the number of ways four balls can be drawn from the box if
 - (1) They can be any colour
 - (2) Two must be white and two red
 - (3) They must all be the same colour.
19. Solve the recurrence relation $a_{n+1}-a_n=3n^2-n$, $n \geq 0$, $a_0=3$
20. Solve the recurrence relation, $S(n)=5(n-1)+2(n-1)$, with $S(0)=3$, $S(1)=1$, by finding its generating function.
21. Solve $G(k)-7G(k-1)+10G(k-2)=8k+6$, for $k \geq 2$.
22. A factory makes custom sports cars at an increasing rate. In the first month only one car is made, in the second month two cars are made, and so on, with n cars made in the n^{th} month.
 - (i) Set up recurrence relation for the number of cars produced in the first n months by this factory.
 - (ii) How many cars are produced in the first year?
23. Solve the recurrence relation $a_n = -3a_{n-1} - 3a_{n-2} - a_{n-3}$ given that $a_0 = 5$, $a_1 = -9$, $a_2 = 15$
24. Solve the recurrence relation $a_n = 3a_{n-1} + 2$, $n \geq 1$ with $a_0 = 1$ by method of generating Functions.
25. Use generating functions to solve the recurrence relation $a_n+3a_{n-1}-4a_{n-2}=0$, $n \geq 2$ with the initial condition $a_0=3$, $a_1= -2$ (or) Solve $S_n+3S_{n-1}-4S_{n-2}=0$, $n \geq 2$ with the initial condition $S_0=3$, $S_1= -2$.
26. Find the generating function of Fibonacci sequence.
27. Use generating function to solve the recurrence relation $S(n+1)-2S(n)=4^n$ with $S(0)=1$, $n \geq 0$.
28. Using method of generating function solve the recurrence relation

$$a_n=4a_{n-1}-4a_{n-2}+4^n; n \geq 2, \text{ given that } a_0=2 \text{ and } a_1=8.$$
29. A total of 1232 students have taken a course in Spanish, 879 have taken a course in French, and 114 have taken a course in Russian. Further, 103 have taken courses in both Spanish and French, 23 have taken courses in both Spanish and Russian, and 14 have taken courses in both French and Russian. If 2092 students have taken atleast one of Spanish, French and Russian, how many students have taken a course in all three languages?
30. There are 2500 students in a college, of these 1700 have taken a course in C, 1000 have taken a course Pascal and 550 have taken a course in Networking. Further 750 have taken courses in both C and Pascal. 400 have taken courses in both C and Networking, and 275 have taken courses in both Pascal and

Networking. If 200 of these students have taken courses in C, Pascal and Networking.

- (i) How many of these 2500 students have taken a course in any of these three courses C, Pascal and Networking? (ii) How many of these 2500 students have not taken a course in any of these three courses C, Pascal and Networking?

31. A valid code word is an n -digit decimal number containing even number of 0's. If a_n denotes the number of valid code words of length n then find an explicit formula for a_n using generating functions.

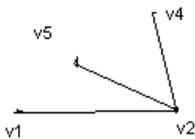
UNIT-III
GRAPHS
PART A

1. Define Graph.

A graph $G = (V(G), E(G))$ consists of V , a non empty set of vertices (nodes or points) and E , a set of edges (also called lines).

2. Define adjacent vertices.

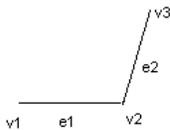
Any pair of vertices which are connected by an edge in a graph is called adjacent vertices.



Here $v_1, v_2; v_2, v_4; v_2, v_3$ are adjacent vertices $v_1, v_3; v_3, v_4; v_1, v_4$ are not adjacent.

3. Define adjacent edges.

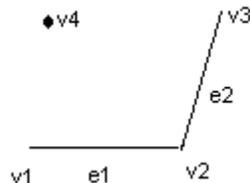
If two distinct edges are incident with a common vertex then they are called adjacent edges.



Here e_1 and e_2 are incident with a common vertex v_2 .

4. Define isolated vertex

In any graph, a vertex which is not adjacent to any other vertex is called an isolated vertex. Otherwise the vertex has no incident edge.



Here v_3 has no incident edge. Therefore the vertex v_3 is called isolated vertex.

5. Define Label graph.

A graph in which each vertex is assigned a unique name or label is called a label graph.

6. Define Directed graph and undirected graph.

In a graph $G(V, E)$, an edge which is associated with an ordered pair of vertices is called a directed edge of graph G , While an edge which is associated with an unordered pair of vertices is called an undirected edge. A graph in which every edge is directed is called a directed graph or simply a digraph.

7. Draw a diagram for the following graph

Solution:

$$G = G(V, E)$$

$$V = \{v_1, v_2, v_3, v_4\}$$

$$E = \{(v_1, v_2), (v_4, v_1), (v_3, v_1), (v_3, v_4)\}$$

8. Define Niche overlap Graphs in Ecology.

A niche overlap graph is a simple graph because no loops or multiple edge are needed in this model.

9. Define the degree of a vertex.

The degree of a vertex in an undirected graph is the number of edges incident with it, except that a loop at a vertex contributes twice to the degree of that vertex.

The degree of the vertex is denoted by $\deg(\)$.

10. Define adjacent vertices in undirected graph.

Two vertices u and v in an undirected graph G are called adjacent (or neighbors) in G if u, v are endpoints of an edge of G .

11. How many edges are there in a graph with 10 vertices each of degree six?

Solution: Sum of the degree of the 10 vertices is $(6)(10)=60$ i.e., $2e=60$ $e=30$.

12. Show that the sum of degree of all the vertices in a graph G , is even.

Proof: Each edge contributes two degree in a graph.

Also, each edge contributes one degree to each of the vertices on which it is incident.

Hence, if there are N edges in G , then

$$2N = d(v_1) + d(v_2) + \dots + d(v_N)$$

Thus, $2N$ is always even.

13. Define In degree.

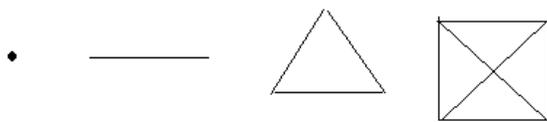
In a directed graph G , the in degree of v denoted by $\text{in deg } G(v)$ or $\text{deg}_G^{-1}(v)$, is the number of edges ending at v .

14. Define out degree.

In a directed graph G , the out degree of v of G denoted by $\text{out deg}_G(v)$ or $\text{deg}_G^+(v)$, is the number of edges beginning at v .

15. Draw Graphs K_n for $1 \leq n \leq 4$

Solution:

**16. What is the degree sequence of K_n , Where n is a positive integer? Explain your answer.**

Solution: Each of the n vertices is adjacent to each of the other $n-1$ vertices, so the degree sequence is $n-1, n-1, \dots, n-1$ (n terms)

17. Define Cycle Graph.

A cycle graph of order 'n' is a connected graph whose edges form a cycle of length 'n' and denoted by C_n .

18. Define Wheel graph.

A Wheel graph of order n is obtained by joining a new vertex called 'Hub' to each vertex of a cycle graph of order n-1, denoted by W_n .

19. Define Regular graph.

A graph in which all vertices are of equal degree is called a regular graph.

If the degree of each vertex is r, then the graph is called a regular graph of degree r.

20. For Which value of n are these graphs regular?

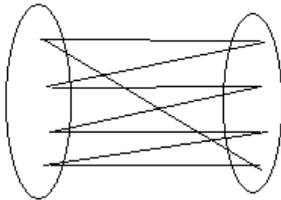
(a) K_n (b) C_n (c) W_n (d) Q_n

Solution: (a) For all $n \geq 1$ (b) For all $n \geq 3$ (c) For $n=3$ (d) For all $n \geq 0$

21. Define bipartite graph.

A bipartite graph is an undirected graph whose set vertices can be partitioned into two sets M and N in such a way that each edge joins a vertex in M to a vertex in N and no edge joins either two vertices in M or two vertices in N.

$$\begin{aligned} V &= M \cup N & M \cap N &= \phi \\ \text{Here } M &= \{v_1, v_3, v_5, v_7\} & N &= \{v_2, v_4, v_6, v_8\} \end{aligned}$$



22. Define Complete Bipartite graph.

A complete bipartite graph is a bipartite graph in which every vertex of M is adjacent to every vertex of N. The complete bipartite graphs that may be partitioned into sets M and N as above s.t $|M|=m$ and $|N|=n$ are denoted by $K_{m,n}$

23. Define star graph.

Any graph that is $K_{1,n}$ is called a star graph.

24. Prove that a graph which contains a triangle can not be bipartite.

Proof: At least two of the three vertices must lie in one of the bipartite sets because there two are joined by edge, the graph can not be bipartite.

25. Define Graph coloring.

The assign of colors to the vertices of G, one color to each vertex, so that adjacent vertices are assigned different color is called the proper coloring of G or simply vertex coloring.

If G has n coloring, then G is said to be n-colorable.

26. Define Subgraph.

A subgraph of a graph $G=(V,E)$ is a graph $H=(W,F)$, where $W \subseteq V$ and $F \subseteq E$. A subgraph H of G is a proper subgraph of G if $H \neq G$.

27. Define Complement.

The complement \overline{G} of G is defined as a simple graph with the same vertex set as G and value two vertices u and v are adjacent only when they are not adjacent in G .

28. Define Adjacency matrix.

Let $G(V, E)$ be a simple graph with n vertices ordered from V_1 and V_n , then the adjacency

matrix $A = [a_{ij}]_{n \times n}$ of G is an $n \times n$ symmetric matrix defined by the elements.

$$a_{ij} = \begin{cases} 1 & \text{when } V_i \text{ is adjacent to } V_j \\ 0 & \text{Otherwise} \end{cases}$$

it is denoted by $A(G)$ or A_G

$$A_G = \begin{matrix} & \begin{matrix} V_1 & V_2 & V_3 & V_4 \end{matrix} \\ \begin{matrix} V_1 \\ V_2 \\ V_3 \\ V_4 \end{matrix} & \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix} \end{matrix}$$

29. Write any properties of adjacency matrix.

1. An adjacency matrix completely defines a simple graph.
2. The adjacency matrix is symmetric.

30. Write the adjacency matrix of C_4 .

Solution:

C_4 graph is

$$\begin{matrix} & \begin{matrix} V_1 & V_2 & V_3 & V_4 \end{matrix} \\ \begin{matrix} V_1 \\ V_2 \\ V_3 \\ V_4 \end{matrix} & \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \end{matrix}$$

31. Define Incidence matrix.

Let G be a graph with n vertices, Let $V = \{V_1, V_2, \dots, V_n\}$ and $E = \{e_1, e_2, \dots, e_m\}$. Define $n \times m$ matrix.

$$I_G = [m_{ij}]_{n \times m} \quad \text{Where } m_{ij} = \begin{cases} 1 & \text{when } V_i \text{ is incident with } e_j \\ 0 & \text{Otherwise} \end{cases}$$

$$I_G = \begin{matrix} & \begin{matrix} e_1 & e_2 & e_3 & e_4 & e_5 \end{matrix} \\ \begin{matrix} V_1 \\ V_2 \\ V_3 \\ V_4 \\ V_5 \end{matrix} & \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{matrix}$$

32. What is the sum of the entries in a row of the incidence matrix for an undirected graph?

Solution: Sum is 2 if e is not a loop, 1 if e is a loop.

33. Define Isomorphic Graphs.

Two graphs D and G' are isomorphic if there is a function

$f: V(G) \rightarrow V(G')$ from the vertices of G to the vertices of G' such that

- f is one-to-one
- f is onto and
- For each pair of vertices u and v of G

$$[u, v] \in E(G) \Leftrightarrow [f(u), f(v)] \in E(G')$$

Any function f with the above three properties is called an isomorphism from G to G'

34. Prove that any 2 simple connected graphs with n vertices all of degree 2 are isomorphic.

Solution: We know that total degree of a graph is given by

$$\sum_{i=1}^n d(V_i) = 2|E|$$

then $|V| = \text{number of vertices } n$ and $|E| = \text{number of edges}$

Further the degree of every vertex is 2.

$$\text{Therefore } \sum_{i=1}^n 2 = 2|E| \Rightarrow 2((n) - 1 + 1) = 2|E| \Rightarrow n = |E|$$

Number of edges = number of vertices. Therefore the graphs are cycle graphs Hence they are isomorphic.

35. Can a simple graph with 7 vertices be isomorphic to its complement?

Solution: A graph with 7 vertices can have a maximum number of edges.

$$= \frac{7(7-1)}{2} = \frac{7 \times 6}{2} = 21 = 21 \text{ edges}$$

21 edges cannot be split into 2 equal integers. Therefore, G and \bar{G} cannot equal number of edges. Hence a graph with 7 vertices cannot be isomorphic to its complement.

36. Define path.

A path in a multigraph G consists of an alternating sequence of vertices and edges of the form

$V_0, e_1, V_1, e_2, V_2, \dots, e_{n-1}, V_{n-1}, e_n, V_n$ Where each edge e_i contains the length of the V_{i-1} and V_i

The number n of edges is called the length of the path.

37. Define circuit.

A path of length ≥ 1 with no repeated edges and whose end vertices are same is called a circuit.

38. Define path graph.

A path graph of order 'n' is obtained by removing one edge from a C_n graph, denoted by P_n .

39. Define trail.

A trail from v to w is a path from v to w that does not contain a repeated edge.

40. Define connected and disconnected graphs.

A graph G is a connected graph if there is at least one path between every pair of vertices in G . otherwise G is a disconnected graph.

41. Define Euler circuit.

An Euler circuit in a graph G is a simple circuit containing every edge of G .

42. Define Euler path.

An Euler path in G is a simple path containing every degree of G .

43. Define Euler line and Euler graph.

A closed walk which contains all edges of the graph G is called an Euler line, and the graph containing at least one Euler line is said to be an Euler graph.

44. Show that values of n is the graph K_n Eulerian?

Solution: We know that K_n , the complete graph of n vertices is a connected graph in which degree of each vertex is $n-1$. Because a graph is Eulerian if and only if it is connected and degree of each vertex is even, we conclude that K_n is an Euler graph if and only if n is odd.

45. Define Hamilton path.

A simple path in a graph G that passes through every vertex exactly once is called a Hamilton path. That is, the simple path $x_0, x_1, \dots, x_{n-1}, x_n$ in the graph $G=(V,E)$ is a Hamilton path if $V = \{x_0, x_1, \dots, x_{n-1}, x_n\}$ and $x_i \neq x_j$ for $0 \leq i < j \leq n$.

46. Define Hamilton circuit.

A simple circuit in a graph G that passes through every vertex exactly once is called a Hamilton circuit. And the simple circuit $x_0, x_1, \dots, x_{n-1}, x_n, x_0$ (with $n > 0$) is a Hamilton circuit if $x_0, x_1, \dots, x_{n-1}, x_n$ is a Hamilton path.

47. Show that K_n has a Hamilton circuit whenever $n \geq 3$.

Solution: Now we can form a Hamilton circuit in K_n beginning at any vertex.

Such a circuit can be built by visiting vertices in any order we choose, as long as the path begins and ends at the same vertex and visits each other vertex exactly once

It is possible since there are edges in K_n between any two vertices.

48. State Dirac's theorem.

If G is a simple graph with r. vertices with $n \geq 3$ such that the degree of every vertex in G is at least $n/2$, then G has a Hamilton circuit.

49. State Ore's theorem.

If G is a simple graph with number of vertices $n(\geq 3)$ and if

$$\text{deg}(u) + \text{deg}(v) \geq n \dots\dots\dots(1)$$

for every pair of non-adjacent vertices u and v, then G is Hamiltonian.

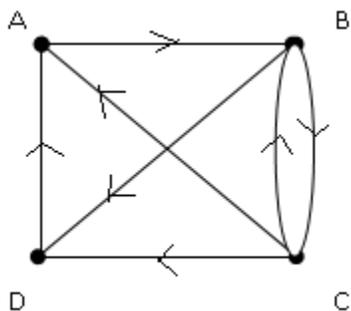
50. Define Gray code.

A Gray code is a labeling of the arcs of the circles such that adjacent arcs are labeled with bit strings that differ in exactly one bit.

NIT-III
PART-B

1. Prove that the number of vertices of odd degree in a graph is always even.
2. Let G be a graph with exactly two vertices has odd degree. Then prove that there is a path between those two vertices.
3. Define a complete graph K_n Draw a complete graph K_6 . What is the degree of each vertex in K_n ? What is the total number of edges in K_n ?
4. Draw the graph with 5 vertices A,B,C,D and E such that $\text{deg}(A)=3$, B is an odd vertex, $\text{deg}(C)=2$ and D and E are adjacent.
5. State and prove hand shaking theorem. Also prove that maximum number of edges in a connected graph with n vertices is $\frac{n(n-1)}{2}$.
6. If G is self complementary graph, then prove that G has $n \equiv 0$ (or) $1 \pmod{4}$ vertices.

7. How many paths of length four are there from A to D in the simple graph G given below.

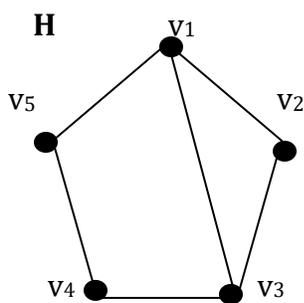
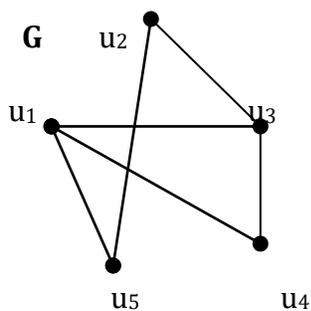


8. Define (i) Adjacency matrix and (ii) Incidence matrix of a graph with example.

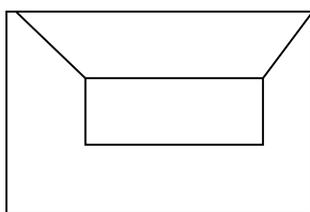
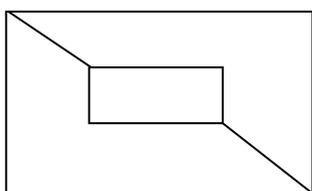
9. Discuss the various graph invariants preserved by isomorphic graphs.

10. Show that isomorphism of simple graphs is an equivalence relation.

11. Determine whether the following graphs G and H are isomorphic. Give reason



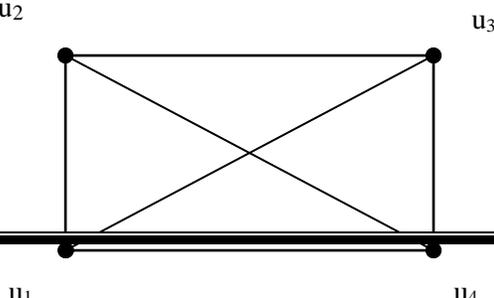
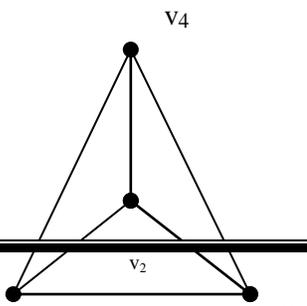
12. Examine whether the following pairs of graphs G1 and G2 given in figures are isomorphic or not.



G1

G2

13. Define Isomorphism. Establish isomorphism between the following graphs.



14. Define isomorphism between two graphs G_1 and G_2 . Are the simple graphs with the following adjacency matrices isomorphic?

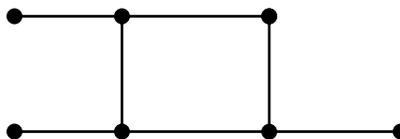
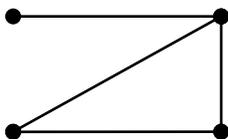
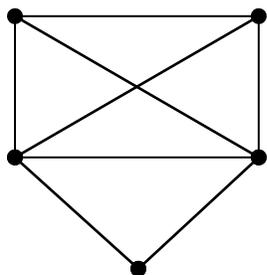
$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

15. If G is a simple graph with n vertices with minimum degree $\delta(G) \geq n/2$ show that G is connected.
16. Show that a simple graph G with n vertices is connected if it has more than $\frac{(n-1)(n-2)}{2}$ Edges.
17. Prove that a graph G is connected if and only if for any partition of V into subsets V_1 and V_2 , there exists an edge joining a vertex of V_1 to a vertex of V_2 .
18. Prove that the maximum number of edges in a simple disconnected graph G with n vertices and k components is $\frac{(n-k)(n-k+1)}{2}$.
19. Define Eulerian graph and Hamiltonian graph. Give an example of graph which is
- Eulerian but not Hamiltonian
 - Hamiltonian and Eulerian
 - Hamiltonian but not Eulerian
 - Neither Hamiltonian nor Eulerian

20. Prove that a connected graph G is Euler graph if and only if every vertex of G is even degree.

21. If G is a Connected simple graph with n vertices and the degree of vertex is atleast $n/2$, then show that G is Hamiltonian.
22. Show that K_n has a Hamiltonian cycle for $n \geq 3$. What is the maximum number of edge - joint Hamilton cycles possible in K_n . Obtain all the edge -disjoint Hamilton cycles in K_7 .
23. Let G be a simple undirected graph with n vertices. Let u and v be two non adjacent vertices in G such that $\deg(u) + \deg(v) \geq n$ in G . Show that G is Hamiltonian if and only if $G+uv$ is Hamiltonian.

24. Which of the following simple graphs have a Hamilton circuit or, if not, a Hamilton path?



25. Describe a discrete structure based on a graph that can be used to model airline Routes and

- their flight times.
26. In a undirected graph, the number of odd degree vertices are even.
27. If all the vertices of an undirected graph are each of degree K , show that the number of edges of the graph is a multiple of K .
28. In a simple digraph $G=(V,E)$, every node of the digraph lies in exactly one strong component.
29. Show that a connected multi-graph has an Euler circuit if and only if each of its vertices has an even degree.
30. Prove that the following are equivalent: (i) G is Eulerian (ii) Every vertex of G has even degree (iii) The set of edges of G can be partitioned into cycles.

UNIT-IV
ALGEBRIC STRUCTURES
PART-A

1. If S denotes the set of positive integers ≤ 100 for $x, y \in S$ define $x * y = \min\{x, y\}$.

Verify whether $(S, *)$ is a monoid assuming that $*$ is associative.

Solution :

100 is the identity element in (s,x) .

since $100*x = \min\{x,100\}=x$ since $x \leq 100$ for all $x \in S$.

Therefore (s,x) is a monoid.

2. If H is a sub-group of a group G , among the right coset of H in G , Prove that there is only one subgroup H .

Solution :

Let Ha be a right coset of H and G where $a \in G$. If Ha is a subgroup of G , then $e \in Ha$ where e is the identity element in G .

3. Give an example of subsemigroup.

Solution :

For the semi-group $(N,+)$, the set E of all even non negative integers is a sub semi group $(E,+)$ of $(N,+)$.

4. Define Normal subgroup of a group.

A subgroup H of a group G is called a normal sub group if for every $a \in G$, $aH = Ha$

5. Find all the cosets of the subgroup $H = \{1, -1\}$ in $G = \{1, -1, i, -i\}$ with the operation Multiplication.

Solution :

Let us find the right cosets of H in G .

$$H(1) = \{1, -1\} = H$$

$$H(-1) = \{-1, 1\} = H$$

$$H(i) = \{i, -i\} \text{ and } H(-i) = \{-i, i\} = Hi$$

$$H1 = H = H-1 = \{1, -1\}$$

$$\text{and } Hi = H-i = \{i, -i\}$$

are the two right cosets of H in G. Similarly we can find the left coset of H in G.

6. A semigroup homomorphism preserves property of associativity.

Solution :

Let $a \in S$ be an idempotent element.

There fore $a*a = a$

$$\Rightarrow g(a*a) = g(a)$$

$$g(a) \circ g(a) = g(a)$$

This shows that $g(a)$ is an idempotent element in T.

There fore the property of idempotent is preserved under semi group homomorphism.

7. A semigroup homomorphism preserves commutativity.

Proof :

Let $a, b \in S$

Assume that $a*b = b*a$

$$g(a*b) = g(b*a)$$

$$g(a) \circ g(b) = g(b) \circ g(a)$$

This means that the operation \circ is commutative in T.

There fore the semi group homomorphism preserves commutativity.

8. Define abelian group and subgroup.

Definition : Abelian group

A group $(G, *)$ is said to be abelian if $a*b = b*a$ for all $a, b \in G$.

Definition : Subgroup

Let $(G, *)$ be a group and let H be a non-empty subset of G. Then H is said to be a sub group of G if H itself is a group with the operation *.

9. Define ring and give an example of a ring with zero-divisors.

Definition : An algebraic system $(S, +, *)$ is called a ring if the binary operations + and * on S satisfy the following 3 properties.

i) $(S, +)$ is an abelian group.

ii) $(S, *)$ is a semi group.

iii) The operation * is distributive over + that is for any $a, b, c \in S$

$$a.(b+c) = a.b + a.c \text{ and}$$

$$(b+c).a = b.a + c.a$$

Example : The ring $(Z_{10}, +_{10}, \times_{10})$ not an integral domain. Since $5 \times_{10} 2 = 0$, yet $5 \neq 0, 2 \neq 0$ in Z_{10} .

10. State Cayley's theorem on Permutation groups.

Every finite group of order 'n' is isomorphic to a permutation group of degree n.

11. Prove that the only idempotent element of a group is its identity element.

Idempotent element : An element $a \in G$ is said to be idempotent with respect to $*$ if $a*a=a$

Identity element : An element e of G is said to be identity element if

$$e*a = a*e = a, \text{ for every } a \in G$$

In a group the identity element e only satisfies the idempotent condition $e*e=e$.

12. What do you call a homomorphism of a semi group into itself?

Solution :

A homomorphism of a semi group into itself is called a semi group endomorphism.

13. Show that if every element in a group is its own inverse, then the group must be abelian.

Solution :

Let $(G, *)$ be a group.

Given : $a^{-1} = a, \forall a \in G$

To prove : $a*b=b*a$ (Abelian)

Let $a, b \in G$

$$\Rightarrow a^{-1} = a \text{ and } b^{-1} = b$$

$$a*b \in G \quad [\text{Since } (G, *) \text{ be a group}]$$

$$(a*b)^{-1} = a*b \quad [\text{Since if every element in a group is its own inverse}]$$

$$b^{-1} * a^{-1} = a*b \quad [\text{Since } (a*b)^{-1} = b^{-1} * a^{-1}]$$

$$b*a = a*b \quad [\text{by (1)}]$$

Since $(G, *)$ is an abelian group.

14. Give an example of a Monoid which is not a group.

$(\mathbb{Z}^+, *)$ is a monoid which is not a group. $\left[\text{Since } a \in G, \frac{1}{a} \notin G \right]$

15. Show that $(\mathbb{Z}_5, +_5)$ is a cyclic group.

$+_5$	[0]	[1]	[2]	[3]	[4]
[0]	0	1	2	3	4
[1]	1	2	3	4	0
[2]	2	3	4	0	1
[3]	3	4	0	1	2
[4]	4	0	1	2	3

16. Define Semi group and monoid. Give an example of a semi group which is not a monoid

Definition : Semi group

Let S be a non empty set and \circ be a binary operation on S . The algebraic system (S, \circ) is called a semigroup if the operation \circ is associative. In other words (S, \circ) is semi group if for any $x, y, z \in G$, $(x \circ y) \circ z = x \circ (y \circ z)$.

Definition : Monoid

An algebraic system (M, \circ) is a Monoid if

- i) \circ is binary
- ii) \circ is associative
- iii) the set M has an identity with respect to the operation \circ .

Example : (\mathbb{Z}, \circ) is a semi group.

The inverse property does not exist.

Therefore (\mathbb{Z}, \circ) is not a monoid.

17. A semigroup homomorphism preserves property of idempotency.

Let $f = (M, *) \rightarrow (H, \Delta)$ be semi group homomorphism.

x idempotent element in M .

Therefore $x * x = x$, $f(x * x) = f(x) \Delta f(x)$

Then $f(x) = f(x) \Delta f(x)$

Therefore idempotency is preserved.

18. Define a semigroup.

Solution:

If a non empty set S together with the binary operation ‘*’ satisfying the following two properties:

a) $a * b = b * a$, $a, b \in S$ (Closure Property)

b) $(a * b) * c = a * (b * c)$, $a, b, c \in S$ (Associative Property)

is called a semigroup. It is denoted by $(S, *)$.

19. If ‘ a ’ is the generator of the cyclic group G , then show that a^{-1} is also a generator of G .

Solution:

$$\text{Now } (a^{-1}) = \{(a^{-1})^n; n \in \mathbb{Z}\} = \{a^{-n}; n \in \mathbb{Z}\} = \{a^m; m \in \mathbb{Z}\} = a$$

UNIT- IV PART- B

1. Show that the mapping g from the algebraic system $(S, +)$ to the system (T, \times) defined by $g(a) = s^a$, where S is the set of all rational numbers under $+$ and T is the set of non-zero real numbers under multiplication operation \times , is a homomorphism but not an isomorphism.
2. The intersection of any two subgroups of a group G is again a subgroup of G - Prove.
3. State & prove Lagrange’s theorem for finite groups.
4. Find all the non-trivial subgroups of $(\mathbb{Z}_6, +_6)$.
5. If every element of a group is its own inverse, prove that G is abelian. Is the converse true?
6. Prove that the direct product of two (or) more groups is again a group.
7. Show that monoid homomorphism preserves invertibility and monoid epimorphism preserves zero element (if it exist).
8. State and prove Fundamental theory on Homomorphism of groups

UNIT-V LATTICES AND BOOLEAN ALGEBRA PART- A

1. Is the lattice of divisors of 32 a Boolean algebra.

No. the divisors of 32 is a chain and hence it is not complemented.

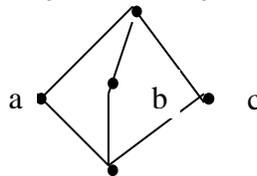
2. Give an example of a lattice which is modular but not distributive.

$M_5 \rightarrow$ Diamond lattice

In M_5 , $a \vee (b \wedge c) = a \vee 0 = a$ while $(a \vee b) \wedge (a \vee c) = 1 \wedge 1 = 1$

So M_5 is not distributive.

As N_5 is not a sublattice of M_5 , M_5 is modular.



3. Obtain the Hasse diagram of $\langle \rho(A_3), \subseteq \rangle$ where $A_3 = \{x, y, z\}$.

Solution :

Given $S = \{x, y, z\}$

$P(S) = \{\{x\}, \{y\}, \{z\}, \{x, y\}, \{y, z\}, \{z, x\}, \{x, y, z\}, \{\}\}$

We know that $\{P(S), \subseteq\}$ is a poset

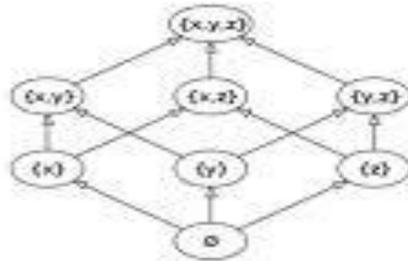
Since empty set is a subset of every set in $P(S)$, $\{\}$ is the least set of $P(S)$.

Similarly $S = \{x, y, z\}$ contains all elements of $P(S)$ i.e an element of $P(S)$ is a subset of $\{x, y, z\}$.

Therefore S is the greatest element in $P(S)$.

Hence $\{P(S), \subseteq\}$ is a lattice.

The Hasse diagram is



4. In a lattice (L, \leq) , prove that $a \wedge (a \vee b) = a$, for all $a, b \in L$.

Solution:

Since $a \wedge b$ is the GLB of $\{a, b\}$, we have

$$a \wedge b \leq a \dots\dots(1)$$

$$\text{Obviously } a \leq a \dots\dots(2)$$

$$\text{From (1) and (2), we have } a \vee (a \wedge b) \leq a \dots\dots(3)$$

$$\text{By definition of LUB, we have } a \leq a \vee (a \wedge b) \dots\dots(4)$$

By combining (3) and (4), we have $a \vee (a \wedge b) = a$. Similarly we can prove $a \wedge (a \vee b) = a$.

5. Give an example of a relation which is symmetric but not reflexive.
6. Give an example of a relation which is reflexive but not symmetric.
7. Give an example of a relation which is both reflexive and symmetric.
8. Give an example of a relation which is neither reflexive nor symmetric.
9. Give an example of a relation which is both symmetric and anti-symmetric.
9. Give an example of a relation which is neither reflexive nor irreflexive.
10. Give an example of a relation which is neither symmetric nor anti-symmetric.
11. Let $X = \{1, 2, 3, 4\}$ & $R = \{\langle 1, 1 \rangle, \langle 4, 1 \rangle, \langle 1, 4 \rangle, \langle 4, 4 \rangle, \langle 2, 2 \rangle, \langle 2, 3 \rangle, \langle 3, 2 \rangle, \langle 3, 3 \rangle\}$.

Write the matrix of R and sketch its graph.

Show that $A \subseteq A \cup B$ and $A \subseteq B \Leftrightarrow A \cup B = B$ $A \cap B \subseteq A$.

UNIT – V
PART- B

1. Let R denote a relation on the set of ordered pairs of positive integers such that $\langle x, y \rangle R \langle u, v \rangle$ iff $xv = yu$. Show that R is an equivalence relation.
2. Prove that the relation “Congruence modulo m ” given by $\equiv = \{ \langle x, y \rangle / x - y \text{ is divisible by } m \}$ over the set of positive integers is an equivalence relation. Show also that if $x_1 = y_1$ & $x_2 = y_2$ then $(x_1 + x_2) = (y_1 + y_2)$.
3. Prove that distinct equivalence classes are disjoint.
4. In a lattice show that $a \leq b$ & $c \leq d$ implies $a * c \leq b * d$.
5. In a distributive lattice prove that $a * b = a * c$ & $a \oplus b = a \oplus c$ implies $b = c$.
6. Let $P = \{ \{1,2\}, \{3,4\}, \{5\} \}$ be a partition of the set $S = \{1,2,3,4,5\}$. Construct an equivalence R on S so that the equivalent classes with respect to R are precisely the members of
7. Show that a chain with two or more elements is not complemented.
8. Establish De Morgan’s laws in a Boolean algebra.
9. In a distributive lattice, prove that the following are equivalent.
 - (a) $a \wedge b \leq x \leq a \vee b$
 - (b) $x = (a \wedge x) \vee (b \wedge x) \vee (a \wedge b)$
10. Prove that any chain ‘ a ’ is modular lattice.
11. In the Boolean algebra of all divisors of 70, find all sub algebras.
12. For any Boolean, prove
 1. $x \vee y = x \vee z, \bar{x} \vee y = \bar{x} \vee z \Rightarrow y = z.$
 2. $x \vee y = 0 \Leftrightarrow x = 0, y = 0.$
 3. $x \leq \bar{y} \Leftrightarrow x \wedge y = 0.$
 4. $x \wedge y = 1 \Leftrightarrow x = 1, y = 1.$
13. Prove that in a distributive lattice every element has unique complement.
14. In any lattice prove that $a \wedge (b \vee c) \geq (a \wedge b) \vee (a \wedge c)$.
15. Let R be a relation on a set A . Then define $R^{-1} = \{ \langle a, b \rangle \in A \times A / \langle b, a \rangle \in R \}$. Prove that if $\langle A, R \rangle$ is a poset then $\langle A, R^{-1} \rangle$ is also a poset.
16. Show that $\langle Z^+, / \rangle$ is distributive.
17. Show that every finite partial ordered set has a maximal and minimal element.
18. Show that if L is a distributive lattice then for all $a, b, c \in L$

$$(a * b) \oplus (b * c) \oplus (c * a) = (a \oplus b) * (b \oplus c) * (c \oplus a).$$
19. In a Boolean algebra, Prove that $(a \wedge b)' = a' \vee b'$ for all $a, b \in L$.

V.S.B. ENGINEERING COLLEGE, KARUR
Department of Computer Science and Engineering
Academic Year: 2018-2019 (ODD Semester)
Assignment Questions

Class: III Year / V Semester B.E. Computer Science and Engineering 'B' Section
 Subject Code/ Subject Name : MA6566/ Discrete Mathematics
 Name of the faculty : T.Parameshwari

Sl. No.	Reg. No.	Questions
1.	922516104061	Show that $\sim(P \wedge Q) \rightarrow (\sim P \vee (\sim P \vee Q)) \equiv \sim P \vee Q$.
2.	922516104062	Show that the following argument is valid: "In a triangle XYZ , there is no pair of angles of equal measure". If a triangle has two sides of equal measure, it is isosceles. If a triangle is isosceles, then it has two angles of equal measure.
3.	922516104063	Find the sequence whose generating function is $\frac{6-29x}{30x^2-11x+1}$ using partial fraction.
4.	922516104064	Establish the validity of the following argument : "All integers are rational numbers. Some integers are powers of 2. Therefore some rational numbers are powers of 2."

		of 2.”
5.	922516104065	Prove that the sum of an irrational number and a rational number is irrational.
6.	922516104066	Show that the following is valid : Babu is in this class watches whale watching. Everyone who watches whale watching cares about ocean pollution. Therefore, someone in the class cares about ocean pollution.
7.	922516104068	Identify the sequence having the expression having the expression $\frac{5+2x}{1-4x^2}$ as a generating function.
8.	922516104069	Find the sequence whose generating function is $\frac{6-29x}{30x^2-11x+1}$ using partial fraction.
9.	922516104070	Solve the recurrence relation, $S(n) = S(n-1) + 2S(n-2)$, with $S(0) = 3, S(1) = 1$, by generating functions.
10.	922516104071	There are three file of identical red, blue and green balls, where each file contains at least 10 balls. In how many ways can 10 balls be selected (i) If there is no restriction (ii) If atleast one red ball must be selected (iii) If atleast one red ball, atleast two blue balls are selected.
11.	922516104072	Prove that $(P \rightarrow Q) \Rightarrow P \rightarrow (P \wedge Q)$.
12.	922516104073	How many solutions does $x_1 + x_2 + x_3 = 11$ have, where x_1, x_2, x_3 are nonnegative integers and $x_1 \leq 3, x_2 \leq 4, x_3 \leq 6$.
13.	922516104074	How many prime numbers not exceeding 100 are there?
14.	922516104075	How many positive integers n can be formed using the digits 3,4,4,5,5,6,7 if n has to exceed 50000?
15.	922516104077	Use Mathematics induction to prove the inequality $n < 2^n$ for all positive integer n.
16.	922516104078	Prove that $n^3 + (n+1)^3 + (n+2)^3$ is divisible by 9, for $n \geq 1$.
17.	922516104079	A team of 11 players is to be chosen from 15 members. In how many ways can it be chosen if (i) one particular player is always included? (ii) Two such players have always to be included?
18.	922516104080	Solve the recurrence relation $a_n - 7a_{n-1} + 10a_{n-2} = 0$ for $n \geq 2$, given that $a_0 = 10, a_1 = 41$ by generating function.
19.	922516104081	Use induction to prove that, If $n \geq 1$, then $1.1! + 2.2! + \dots + n.n! = (n+1)! - 1$.
20.	922516104082	Show that the following two statements are logically equivalent: “It is not true that all comedians are funny” and “There are some comedians who are not funny” .
21.	922516104083	Test the truth value of the statement “Every positive integer is the sum of the squares of two integers”.
22.	922516104084	Prove the following implication $(P \rightarrow Q) \wedge \neg Q \Rightarrow \neg P$.
23.	922516104085	Find the number of positive integers less than or equal to 1000 that are divisible by 10.
24.	922516104086	How many solutions does $x_1 + x_2 + x_3 = 11$ have, where x_1, x_2, x_3 are nonnegative integers and $x_1 \leq 3, x_2 \leq 4, x_3 \leq 6$.
25.	922516104089	Solve the recurrence relation, $S(n) = S(n-1) + 2S(n-2)$, with $S(0) = 3, S(1) = 1$, by generating functions.
26.	922516104090	Solve the recurrence relation $a_n - 7a_{n-1} + 10a_{n-2} = 0$ for $n \geq 2$ with $a_0 = 10, a_1 = 41$ by generating function.
27.	922516104091	Solve the recurrence relation $a_n - 7a_{n-1} + 10a_{n-2} = 0$ for $n \geq 2$, given that $a_0 = 10, a_1 = 41$ by generating function.
28.	922516104092	Identify the sequence having the expression having the expression $\frac{5+2x}{1-4x^2}$ as a generating function.
29.	922516104093	Test the validity of the following argument: If an integer is divisible by 10, then it is divisible by 2, then it is divisible by 3. Therefore the integer divisible by 10 is also divisible by 3.
30.	922516104094	Establish the validity of the following argument : “All integers are rational numbers. Some integers are powers of 2. Therefore some rational numbers are powers of 2.”

31.	922516104096	Using generating function solve the recurrence relation corresponding to the Fibonacci sequence. $a_n = a_{n-1} + a_{n-2}$, $n \geq 2$ with $a_0 = a_1 = 1$.
32.	922516104097	Show that the following is valid : Raju is in this class watches whale watching. Everyone who watches whale watching cares about ocean pollution. Therefore, someone in the class cares about ocean pollution.
33.	922516104098	Use Mathematics induction to prove the inequality $n < 2^n$ for all positive integer n .
34.	922516104099	Prove that the sum of an irrational number and a rational number is irrational.
35.	922516104100	Prove by mathematical induction that $6^{n+2} + 7^{2n+1}$ is divisible by 43 for each positive integer n .
36.	922516104101	Show that the following argument is valid: "In a triangle XYZ , there is no pair of angles of equal measure". If a triangle has two angles of equal measure, then it is isosceles. If a triangle is isosceles, then it has two angles, of equal measure.
37.	922516104102	If n Pigeonholes are occupied by $(kn+1)$ pigeons, where k is positive integer, prove that at least one pigeonhole is occupied by $k+1$ or more pigeons. Hence, find the minimum number of integers to be selected from $S = \{1, 2, \dots, 9\}$ so that the sum of the m integers are even.
38.	922516104103	Use induction to prove that, If $n \geq 1$, then $1.1! + 2.2! + \dots + n.n! = (n+1)! - 1$.
39.	922516104104	Prove that $n^3 + (n+1)^3 + (n+2)^3$ is divisible by 9, for $n \geq 1$.
40.	922516104105	How many positive integers n can be formed using the digits 3,4,4,5,5,6,7 if n has to exceed 5000?
41.	922516104106	A team of 11 players is to be chosen from 15 members. In how many ways can it be chosen if (i) one particular player is always included? (ii) Two such players have always to be included?
42.	922516104107	Show that $\sim(P \wedge Q) \rightarrow (\sim P \vee (\sim P \vee Q)) \equiv \sim P \vee Q$.
43.	922516104108	A factory makes custom sports cars at an increasing rate. In the first month only one car is made, in the second month two cars are made, and so on, with n cars made in the n^{th} month. (i) Set up recurrence relation for the number of cars produced in the first n months of the factory. (ii) How many cars are produced in the first year?
44.	922516104110	Prove that $(P \rightarrow Q) \Rightarrow P \rightarrow (P \wedge Q)$.
45.	922516104111	Find the number of positive integers less than or equal to 1000 that are divisible by 3 or 5.
46.	922516104112	How many of these 2500 students have taken a course in any of these three courses C, Pascal and Networking? (ii) How many of these 2500 students have not taken a course in any of these three courses C, Pascal and Networking?
47.	922516104113	Test the truth value of the statement "Every positive integer is the sum of the squares of two integers".
48.	922516104114	Prove the following implication $(P \rightarrow Q) \wedge \neg Q \Rightarrow \neg P$.
49.	922516104115	Show that the following two statements are logically equivalent: "It is not true that all comedians are funny" and "There are some comedians who are not funny".
50.	922516104116	Solve the recurrence relation $a_n - 7a_{n-1} + 10a_{n-2} = 0$ for $n \geq 2$, given that $a_0 = 1, a_1 = 2$ using generating function.
51.	922516104117	How many prime numbers not exceeding 100 are there?
52.	922516104118	There are 2500 students in a college, of these 1700 have taken a course in C, 1000 in Pascal and 550 have taken a course in Networking. Further 750 have taken courses in both C and Pascal, 400 have taken courses in both C and Networking, and 275 have taken courses in both Pascal and Networking. If 200 of these students have taken courses in C, Pascal and Networking.
53.	922516104119	Test the validity of the following argument: If an integer is divisible by 10, then it is divisible by 2, then it is divisible by 3. Therefore the integer divisible by 10 is also divisible by 3.
54.	922516104120	Use Mathematics induction to prove the inequality $n < 2^n$ for all positive integer n .
55.	922516104301	Show that $(\exists x)P(x) \rightarrow \forall xQ(x) \Rightarrow (\exists x)(P(x) \rightarrow Q(x))$
56.	922516104701	Verify the validity of the following argument. Every living thing is a plant or an animal.

		alive and it is not a plant. All animals have hearts. Therefore John's gold fish has a heart.
--	--	---

Signature of the faculty member
HOD

CS6501 - Internet programming

Unit- I Part - A	
1	Define Java. Java is a programming language expressly designed for use in the distributed environment of the Internet. It was designed to have the "look and feel" of the C++ language, but it is simpler to use than C++ and enforces an object-oriented programming model.
2.	What is a Class? Class is a template for a set of objects that share a common structure and a common behaviour.
3.	What is an Object? Object is an instance of a class. It has state, behaviour and identity. It is also called as an instance of a class.
4.	What is an Instance? An instance has state, behaviour and identity. The structure and behaviour of similar classes are defined in their common class. An instance is also called as an object.
5.	What are different types of access modifiers (Access specifiers)? Access specifiers are keywords that determine the type of access to the member of a class. These keywords are for allowing privileges to parts of a program such as functions and variables. These are: <i>public</i> : Anything declared as public can be accessed from anywhere. <i>private</i> : Anything declared as private can't be seen outside of its class. <i>protected</i> : Anything declared as protected can be accessed by classes in the same package and subclasses in the there packages. <i>default modifier</i> : Can be accessed only to classes in the same package.
6.	What is method overloading and method overriding? Method overloading: When a method in a class having the same method name with different arguments is said to be method overloading. Method overriding: When a method in a class having the same method name with same arguments is said to be method overriding.
7.	List the access specifier used in JAVA? Java provides a number of access modifiers to set access levels for classes, variables, methods and constructors. The four access levels are: <ul style="list-style-type: none"> • Visible to the package. the default. No modifiers are needed. • Visible to the class only (private). • Visible to the world (public). • Visible to the package and all subclasses (protected).
8.	What is the difference between Array and vector? Array is a set of related data type and static whereas vector is a growable array of objects and dynamic
9.	What is a package? A package is a collection of classes and interfaces that provides a high-level layer of access protection and name space management.
10.	What is meant by Inheritance? Inheritance is a relationship among classes, wherein one class shares the structure or behaviour defined in another class. This is called Single Inheritance. If a class shares the structure or behaviour from multiple classes, then it is called Multiple Inheritance. Inheritance defines "is-a" hierarchy among classes in which one subclass inherits from one or more generalised superclasses.
11.	What is an Abstract Class? Abstract class is a class that has no instances. An abstract class is written with the expectation that its concrete subclasses will add to its structure and behaviour, typically by implementing its abstract operations.
12.	What are inner class and anonymous class? Inner class : classes defined in other classes, including those defined in methods are called inner classes. An inner

	<p>class can have any accessibility including private.</p> <p>Anonymous class: Anonymous class is a class defined inside a method without a name and is instantiated and declared in the same place and cannot have explicit constructors.</p>
13.	<p>Define interface and write the syntax of the Interface.</p> <p>Interface is an outside view of a class or object which emphasizes its abstraction while hiding its structure and secrets of its behaviour.</p> <p>Syntax:</p> <pre>[visibility] interface InterfaceName [extends other interfaces] { constant declarations abstract method declarations }</pre>
14.	<p>What is the difference between abstract class and interface?</p> <p>a) All the methods declared inside an interface are abstract whereas abstract class must have at least one abstract method and others may be concrete or abstract.</p> <p>b) In abstract class, key word abstract must be used for the methods whereas interface we need not use that keyword for the methods.</p> <p>c) Abstract class must have subclasses whereas interface can't have subclasses.</p>
15.	<p>What is an exception?</p> <p>An exception is an event, which occurs during the execution of a program, that disrupts the normal flow of the program's instructions.</p>
16.	<p>What is meant by JAVA package?(Nov/Dec 2014)</p> <p>Package represents a collection of classes, methods and interface. The name of the package must be written as the first statement in the java source program. The syntax of specifying the package in the java program is: package name_of_package</p>
17.	<p>What are the types of Exceptions in Java?</p> <p>There are two types of exceptions in Java, unchecked exceptions and checked exceptions.</p> <p>Checked exceptions: A checked exception is some subclass of Exception (or Exception itself), excluding class RuntimeException and its subclasses. Each method must either handle all checked exceptions by supplying a catch clause or list each unhandled checked exception as a thrown exception.</p> <p>Unchecked exceptions: All Exceptions that extend the RuntimeException class are unchecked exceptions. Class Error and its subclasses also are unchecked.</p>
18.	<p>What are the different ways to handle exceptions?</p> <p>There are two ways to handle exceptions:</p> <ul style="list-style-type: none"> Wrapping the desired code in a try block followed by a catch block to catch the exceptions. List the desired exceptions in the throws clause of the method and let the caller of the method handle those exceptions.
19.	<p>How to create custom exceptions?</p> <p>the Exception class or one of its subclasses.</p> <p>By Extending</p> <pre>class MyException extends Exception { public MyException() { super(); } public MyException(String s) { super(s); } }</pre>
20.	<p>Write the properties of Threads.(Nov/Dec 2014).</p> <ul style="list-style-type: none"> Thread Priority Deamon Thread Thread group
21.	<p>What is multi-threaded programming?(Nov/Dec 2014)</p>

	<p>Multithreading is the ability of a program or an operating system process to manage its use by more than one user at a time and to even manage multiple requests by the same user without having to have multiple copies of the programming running in the computer.</p>
22.	<p>Write the life cycle of thread. A thread goes through various stages in its life cycle. For example, a thread is born, started, runs, and then dies. Following diagram shows complete life cycle of a thread.</p> <pre> graph TD new((new)) -- "program starts thread using start()" --> runnable([runnable]) runnable -- "unlock, signal, singalAll" --> waiting([waiting]) waiting -- "await, lock" --> runnable runnable -- "await, sleep" --> timed_waiting([timed waiting]) timed_waiting -- "interval expires" --> runnable runnable -- "thread completes" --> terminated([terminated]) </pre>
23.	<p>What is daemon thread and which method is used to create the daemon thread? A daemon thread is a thread, that does not prevent the JVM from exiting when the program finishes but the thread is still running. An example for a daemon thread is the garbage collection. You can use the <code>setDaemon()</code> method to change the Thread daemon properties</p>
24.	<p>What is the purpose of toString() method in java ? The <code>toString()</code> method returns the string representation of any object. If you print any object, java compiler internally invokes the <code>toString()</code> method on the object. So overriding the <code>toString()</code> method, returns the desired output, it can be the state of an object etc. depends on your implementation.</p>
25.	<p>What is immutable string in java? In java, string objects are immutable. Immutable simply means unmodifiable or unchangeable. Once string object is created its data or state can't be changed but a new string object is created. Eg:</p> <pre> class Testimmutablestring{ public static void main(String args[]){ String s="Sachin"; s.concat(" Tendulkar");//concat() method appends the string at the end System.out.println(s);//will print Sachin because strings are immutable objects } </pre>
26.	<p>Define assert . Java assertion feature allows developer to put "assert" statements in Java source code to help unit testing and debugging. An "assert" statement has the following format: <i>assert boolean_expression : string_expression;</i> When this statement is executed: If <code>boolean_expression</code> evaluates to true, the statement will pass normally. If <code>boolean_expression</code> evaluates to false, the statement will fail with an "AssertionError" exception.</p>

27.	<p>Define Applet. An applet is a small Internet-based program written in Java, a programming language for the Web, which can be downloaded by any computer. The applet is also able to run in HTML. The applet is usually embedded in an HTML page on a Web site and can be executed from within a browser.</p>								
28.	<p>Define transient and volatile Modifiers. Java defines two interesting type modifiers: transient and volatile. These modifiers are used to handle somewhat specialized situations. When an instance variable is declared as transient, then its value need not persist when an object is stored. For example: <pre>class T { transient int a; // will not persist int b; // will persist }</pre> Here, if an object of type T is written to a persistent storage area, the contents of a would not be saved, but the contents of b would.</p>								
29.	<p>What is use of the run-time operator instanceof. The instanceof operator has this general form: <pre>objref instanceof type</pre> Here, objref is a reference to an instance of a class, and type is a class type. If objref is of the specified type or can be cast into the specified type, then the instanceof operator evaluates to true. Otherwise, its result is false. Thus, instanceof is the means by which your program can obtain run-time type information about an object</p>								
30.	<p>How to Enabling and Disabling Assertion Options? When executing code, you can disable assertions by using the -da option. You can enable or disable a specific package by specifying its name after the -ea or -da option. For example, to enable assertions in a package called MyPack, use -ea:MyPack To disable assertions in MyPack, use -da:MyPack</p>								
31.	<p>Define String Constructors. The String class supports several constructors. To create an empty String, you call the default constructor. For example, String s = new String(); will create an instance of String with no characters in it. Frequently, you will want to create strings that have initial values. The String class provides a variety of constructors to handle this. To create a String initialized by an array of characters, use the constructor shown here: String(char chars[]) Here is an example: <pre>char chars[] = { 'a', 'b', 'c' }; String s = new String(chars);</pre> This constructor initializes s with the string “abc”.</p>								
32.	<p>What are the String Comparison? The String class includes several methods that compare strings or substrings within strings.</p> <table border="1" data-bbox="196 1581 1523 1890"> <tr> <td data-bbox="196 1581 618 1650">equals() and equalsIgnoreCase()</td> <td data-bbox="618 1581 1523 1650">To compare two strings for equality, use equals(). To perform a comparison that ignores case differences, call equalsIgnoreCase().</td> </tr> <tr> <td data-bbox="196 1650 618 1719">regionMatches()</td> <td data-bbox="618 1650 1523 1719">The regionMatches() method compares a specific region inside a string with another specific region in another string.</td> </tr> <tr> <td data-bbox="196 1719 618 1854">startsWith() and endsWith()</td> <td data-bbox="618 1719 1523 1854">The startsWith() method determines whether a given String begins with a specified string. Conversely, endsWith() determines whether the String in question ends with a specified string.</td> </tr> <tr> <td data-bbox="196 1854 618 1890">equals() Versus ==</td> <td data-bbox="618 1854 1523 1890">The equals() method compares the characters inside a String object. The ==</td> </tr> </table>	equals() and equalsIgnoreCase()	To compare two strings for equality, use equals(). To perform a comparison that ignores case differences, call equalsIgnoreCase().	regionMatches()	The regionMatches() method compares a specific region inside a string with another specific region in another string.	startsWith() and endsWith()	The startsWith() method determines whether a given String begins with a specified string. Conversely, endsWith() determines whether the String in question ends with a specified string.	equals() Versus ==	The equals() method compares the characters inside a String object. The ==
equals() and equalsIgnoreCase()	To compare two strings for equality, use equals(). To perform a comparison that ignores case differences, call equalsIgnoreCase().								
regionMatches()	The regionMatches() method compares a specific region inside a string with another specific region in another string.								
startsWith() and endsWith()	The startsWith() method determines whether a given String begins with a specified string. Conversely, endsWith() determines whether the String in question ends with a specified string.								
equals() Versus ==	The equals() method compares the characters inside a String object. The ==								

		= operator compares two object references to see whether they refer to the same instance.
	compareTo()	to simply know whether two strings are identical

33. List the name of methods for modifying string.

- substring()
- concat()
- replace()
- trim()

PART – B

1. (i) Describe the concepts of OOP.(5)
 Object Oriented Programming is a paradigm that provides many concepts such as **inheritance, data binding, polymorphism** etc.
Object-Oriented Programming is a methodology or paradigm to design a program using classes and objects. It simplifies the software development and maintenance by providing some concepts:

- Object
- Class
- Inheritance
- Polymorphism
- Abstraction
- Encapsulation

Object
 Any entity that has state and behavior is known as an object. For example: chair, pen, table, keyboard, bike etc. It can be physical and logical.

Class
 Collection of objects is called class. It is a logical entity.

Inheritance
 When one object acquires all the properties and behaviours of parent object i.e. known as inheritance. It provides code reusability. It is used to achieve runtime polymorphism.

Polymorphism
 When one task is performed by different ways i.e. known as polymorphism. For example: to converse the customer differently, to draw something e.g. shape or rectangle etc. In java, we use method overloading and method overriding to achieve polymorphism. Another example can be to speak something e.g. cat speaks meaw, dog barks woof etc.

Abstraction
 Hiding internal details and showing functionality is known as abstraction. For example: phone call, we don't know the internal processing. In java, we use abstract class and interface to achieve abstraction.

Encapsulation
 Binding (or wrapping) code and data together into a single unit is known as encapsulation. For example: capsule, it is wrapped with different medicines.
 A java class is the example of encapsulation. Java bean is the fully encapsulated class because all the data members are private here.

(ii) What is meant by overriding method? Give example.(5)
 If subclass (child class) has the same method as declared in the parent class, it is known as **method overriding in java**.
 In other words, If subclass provides the specific implementation of the method that has been provided by one of its parent class, it is known as method overriding.

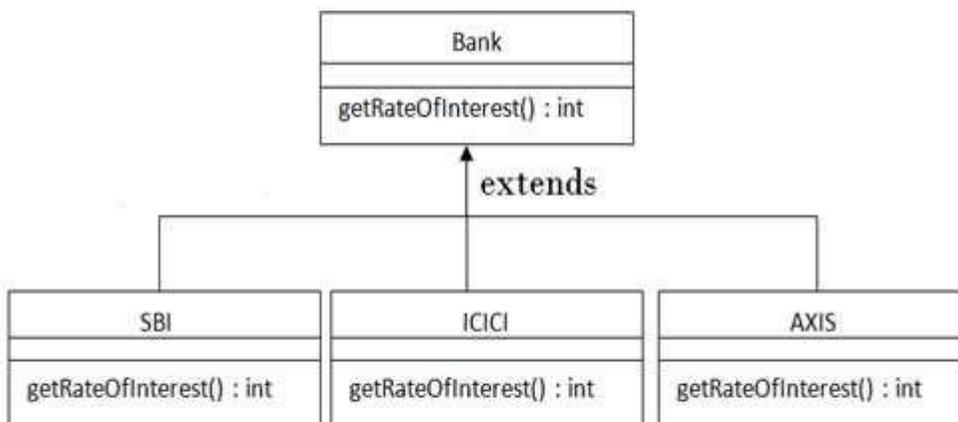
Usage of Java Method Overriding

- Method overriding is used to provide specific implementation of a method that is already provided by its super class.
- Method overriding is used for runtime polymorphism

Rules for Java Method Overriding

1. method must have same name as in the parent class
2. method must have same parameter as in the parent class.
3. must be IS-A relationship (inheritance).

Consider a scenario, Bank is a class that provides functionality to get rate of interest. But, rate of interest varies according to banks. For example, SBI, ICICI and AXIS banks could provide 8%, 7% and 9% rate of interest.



```
class Bank{
int getRateOfInterest(){return 0;}
}
```

```
class SBI extends Bank{
int getRateOfInterest(){return 8;}
}
```

```
class ICICI extends Bank{
int getRateOfInterest(){return 7;}
}
```

```
class AXIS extends Bank{
int getRateOfInterest(){return 9;}
}
```

```
class Test2{
public static void main(String args[]){
SBI s=new SBI();
ICICI i=new ICICI();
AXIS a=new AXIS();
}
```

```
System.out.println("SBI Rate of Interest: "+s.getRateOfInterest());
System.out.println("ICICI Rate of Interest: "+i.getRateOfInterest());
System.out.println("AXIS Rate of Interest: "+a.getRateOfInterest());
}
}
```

Output:

SBI Rate of Interest: 8

ICICI Rate of Interest: 7

AXIS Rate of Interest: 9

(iii) Write a JAVA program to reverse the given number.(6)

```
import java.util.*;
```

```
public class RevNumString
```

```
{
```

```
    public static void main(String[] args)
```

```
    {
```

```
        Scanner scanner = new Scanner(System.in);
```

```
        System.out.println("Please enter a number: ");
```

```
        int num = scanner.nextInt();
```

```
        System.out.println("Please enter a string: ");
```

```
        String str = scanner.next();
```

```
        RevNumString rns = new RevNumString();
```

```
        int revNum = rns.reverse(num);
```

```
        String revStr = rns.reverse(str);
```

```
        System.out.printf("\n The reverse of number %d is %d ", num, revNum);
```

```
        System.out.printf("\n The reverse of string '%s' is '%s' ", str, revStr);
```

```
    }
```

```
    // Method to return the reverse of a number
```

```
    public int reverse(int num) {
```

```
        int revNum = 0;
```

```
        while (num > 0) {
```

```
            int rem = num % 10;
```

```
            revNum = (revNum * 10) + rem;
```

```
            num = num / 10;
```

```
        }
```

```
        return revNum;
```

```
    }
```

```
    // Method to return the reverse of a string
```

```
    public String reverse(String str) {
```

```
        StringBuilder revStr = new StringBuilder();
```

```
        for (int i = str.length()-1; i >= 0; i--) {
```

```
            revStr.append(str.charAt(i));
```

```
        }
```

```
        return revStr.toString();
```

```
    }
```

```
}
```

	<p>Program Output: Please enter a number: 1234 Please enter a string: Java</p> <p>The reverse of number 1234 is 4321 The reverse of string 'Java' is 'avaJ'</p>
2.	<p>(i) What is meant by package? How it is created and implemented in JAVA.(8) Packages are used in Java in order to prevent naming conflicts, to control access, to make searching/locating and usage of classes, interfaces, enumerations and annotations easier, etc.</p> <p>A Package can be defined as a grouping of related types(classes, interfaces, numerations and annotations) providing access protection and name space management. Some of the existing packages in Java are::</p> <ul style="list-style-type: none"> • java.lang - bundles the fundamental classes • java.io - classes for input , output functions are bundled in this package <p>Programmers can define their own packages to bundle group of classes/interfaces, etc. It is a good practice to group related classes implemented by you so that a programmer can easily determine that the classes, interfaces, enumerations, annotations are related. Since the package creates a new namespace there won't be any name conflicts with names in other packages. Using packages, it is easier to provide access control and it is also easier to locate the related classes.</p> <p>Creating a package: When creating a package, you should choose a name for the package and put a package statement with that name at the top of every source file that contains the classes, interfaces, enumerations, and annotation types that you want to include in the package.</p> <p>The package statement should be the first line in the source file. There can be only one package statement in each source file, and it applies to all types in the file.</p> <p>If a package statement is not used then the class, interfaces, enumerations, and annotation types will be put into an unnamed package.</p> <p>Example: Let us look at an example that creates a package called animals. It is common practice to use lowercased names of packages to avoid any conflicts with the names of classes, interfaces. Put an interface in the package <i>animals</i>: /* File name : Animal.java */ package animals;</p> <pre>interface Animal { public void eat(); public void travel();</pre>

```
}  
Now, put an implementation in the same package animals:  
package animals;
```

```
/* File name : MammalInt.java */  
public class MammalInt implements Animal{  
  
    public void eat(){  
        System.out.println("Mammal eats");  
    }  
  
    public void travel(){  
        System.out.println("Mammal travels");  
    }  
  
    public int noOfLegs(){  
        return 0;  
    }  
  
    public static void main(String args[]){  
        MammalInt m = new MammalInt();  
        m.eat();  
        m.travel();  
    }  
}
```

Now, you compile these two files and put them in a sub-directory called *animals* and try to run as follows:

```
$ mkdir animals  
$ cp Animal.class MammalInt.class animals  
$ java animals/MammalInt  
Mammal eats  
Mammal travels
```

The import Keyword:

If a class wants to use another class in the same package, the package name does not need to be used. Classes in the same package find each other without any special syntax.

Example:

Here, a class named *Boss* is added to the payroll package that already contains *Employee*. The *Boss* can then refer to the *Employee* class without using the payroll prefix, as demonstrated by the following *Boss* class.

```
    package payroll;  
    public class Boss  
    {  
        public void payEmployee(Employee e)  
        {
```

```
        e.mailCheck();
    }
}
```

What happens if Boss is not in the payroll package? The Boss class must then use one of the following techniques for referring to a class in a different package.

- The fully qualified name of the class can be used. For example:
payroll.Employee
- The package can be imported using the import keyword and the wild card (*). For example:
import payroll.;*
- The class itself can be imported using the import keyword. For example:
import payroll.Employee;

Note: A class file can contain any number of import statements. The import statements must appear after the package statement and before the class declaration.

The Directory Structure of Packages:

Two major results occur when a class is placed in a package:

- The name of the package becomes a part of the name of the class, as we just discussed in the previous section.
- The name of the package must match the directory structure where the corresponding bytecode resides.

Here is simple way of managing your files in Java:

Put the source code for a class, interface, enumeration, or annotation type in a text file whose name is the simple name of the type and whose extension is .java. For example:

```
// File Name : Car.java
package vehicle;
public class Car {
    // Class implementation.
}
```

Now, put the source file in a directory whose name reflects the name of the package to which the class belongs:

```
....\vehicle\Car.java
```

Now, the qualified class name and pathname would be as below:

- Class name -> vehicle.Car
- Path name -> vehicle\Car.java (in windows)

In general, a company uses its reversed Internet domain name for its package names. Example: A company's Internet domain name is apple.com, then all its package names would start with com.apple. Each component of the package name corresponds to a subdirectory.

Example: The company had a com.apple.computers package that contained a Dell.java source file, it would be contained in a series of subdirectories like this:

```
....\com\apple\computers\Dell.java
```

At the time of compilation, the compiler creates a different output file for each class, interface and enumeration defined in it. The base name of the output file is the name of the type, and its extension is .class

For example:

```
// File Name: Dell.java
package com.apple.computers;
public class Dell{
}
class Ups{

}
```

Now, compile this file as follows using -d option:

```
$javac -d . Dell.java
```

(ii) Write a JAVA program to find the smallest number in the given list. (8)

```
import java.util.Scanner;
class group{
public static void main(String arng[]){
int value[]= new int[5];
int temp,i;
Scanner data = new Scanner(System.in);
System.out.println("Enter 5 element of array" );
// Enhanced for loop
for(i=0; i < 5; i++ )
value[i] = data.nextInt();
// finding smallest number
temp = value[0];
for(i=0; i < 5; i++ )
{
if(temp < value[i])
continue;
else
temp = value[i];
}
System.out.println("Smallest number in array is "+temp);
}
}
```

Output:-

```
Enter 5 element of array
56
84
95
12
32
Smallest number in array is 12
```

Output:-

```
Enter 5 element of array
48
124
```

	<p>20 54 14 Smallest number in array is 14</p>
3.	<p>(i) What is meant by interface? How it is declared and implemented in JAVA. Give example.(12) An interface is a collection of abstract methods. A class implements an interface, thereby inheriting the abstract methods of the interface.</p> <p>An interface is not a class. Writing an interface is similar to writing a class, but they are two different concepts. A class describes the attributes and behaviors of an object. An interface contains behaviors that a class implements.</p> <p>Unless the class that implements the interface is abstract, all the methods of the interface need to be defined in the class.</p> <p>An interface is similar to a class in the following ways:</p> <ul style="list-style-type: none"> (ii) An interface can contain any number of methods. (iii) An interface is written in a file with a .java extension, with the name of the interface matching the name of the file. (iv) The bytecode of an interface appears in a .class file. (v) Interfaces appear in packages, and their corresponding bytecode file must be in a directory structure that matches the package name. <p>However, an interface is different from a class in several ways, including:</p> <ul style="list-style-type: none"> (vi) You cannot instantiate an interface. (vii) An interface does not contain any constructors. (viii) All of the methods in an interface are abstract. (ix) An interface cannot contain instance fields. The only fields that can appear in an interface must be declared both static and final. (x) An interface is not extended by a class; it is implemented by a class. (xi) An interface can extend multiple interfaces. <p>Declaring Interfaces:</p> <p>The interface keyword is used to declare an interface. Here is a simple example to declare an interface: Example:</p> <p>Let us look at an example that depicts encapsulation:</p> <pre> /* File name : NameOfInterface.java */ import java.lang.*; //Any number of import statements public interface NameOfInterface </pre>

```
{  
  //Any number of final, static fields  
  //Any number of abstract method declarations\  
}
```

Interfaces have the following properties:

- (xii) An interface is implicitly abstract. You do not need to use the **abstract** keyword when declaring an interface.
- (xiii) Each method in an interface is also implicitly abstract, so the abstract keyword is not needed.

Methods in an interface are implicitly public.

Example:

```
/* File name : Animal.java */  
interface Animal {  
  
  public void eat();  
  public void travel();  
}
```

Implementing Interfaces:

When a class implements an interface, you can think of the class as signing a contract, agreeing to perform the specific behaviors of the interface. If a class does not perform all the behaviors of the interface, the class must declare itself as abstract.

A class uses the **implements** keyword to implement an interface. The implements keyword appears in the class declaration following the extends portion of the declaration.

```
/* File name : MammalInt.java */  
public class MammalInt implements Animal{  
  
  public void eat(){  
    System.out.println("Mammal eats");  
  }  
  
  public void travel(){  
    System.out.println("Mammal travels");  
  }  
  
  public int noOfLegs(){  
    return 0;  
  }  
}
```

```

public static void main(String args[]){
    MammalInt m = new MammalInt();
    m.eat();
    m.travel();
}
}

```

This would produce the following result:

```

Mammal eats
Mammal travels

```

When overriding methods defined in interfaces there are several rules to be followed:

Checked exceptions should not be declared on implementation methods other than the ones declared by the interface method or subclasses of those declared by the interface method.

The signature of the interface method and the same return type or subtype should be maintained when overriding the methods.

An implementation class itself can be abstract and if so interface methods need not be implemented.

When implementing interfaces there are several rules:

- (xiv) A class can implement more than one interface at a time.
- (xv) A class can extend only one class, but implement many interfaces.
- (xvi) An interface can extend another interface, similarly to the way that a class can extend another class.

Extending Interfaces:

An interface can extend another interface, similarly to the way that a class can extend another class. The extends keyword is used to extend an interface, and the child interface inherits the methods of the parent interface.

The following Sports interface is extended by Hockey and Football interfaces.

```

//Filename: Sports.java
public interface Sports
{
    public void setHomeTeam(String name);
    public void setVisitingTeam(String name);
}

```

```
//Filename: Football.java
public interface Football extends Sports
{
    public void homeTeamScored(int points);
    public void visitingTeamScored(int points);
    public void endOfQuarter(int quarter);
}
```

```
//Filename: Hockey.java
public interface Hockey extends Sports
{
    public void homeGoalScored();
    public void visitingGoalScored();
    public void endOfPeriod(int period);
    public void overtimePeriod(int ot);
}
```

The Hockey interface has four methods, but it inherits two from Sports; thus, a class that implements Hockey needs to implement all six methods. Similarly, a class that implements Football needs to define the three methods from Football and the two methods from Sports.

Extending Multiple Interfaces:

A Java class can only extend one parent class. Multiple inheritance is not allowed. Interfaces are not classes, however, and an interface can extend more than one parent interface.

The extends keyword is used once, and the parent interfaces are declared in a comma-separated list.

For example, if the Hockey interface extended both Sports and Event, it would be declared as:

```
public interface Hockey extends Sports, Event
```

iii) Write note on final keyword.(4)

The **final keyword** in java is used to restrict the user. The java final keyword can be used in many context. Final can be:

- i) variable
- ii) method
- iii) class

The final keyword can be applied with the variables, a final variable that have no value it is called blank final variable or uninitialized final variable. It can be initialized in the constructor only. The blank final variable can be static also which will be initialized in the static block only. We will have detailed learning of these

Example of final variable

```
class Bike9{
    final int speedlimit=90;//final variable
    void run(){
```

```

speedlimit=400;
}
public static void main(String args[]){
Bike9 obj=new Bike9();
obj.run();
}
} //end of class

```

Example of final method

```

class Bike{
final void run(){System.out.println("running");}
}

class Honda extends Bike{
void run(){System.out.println("running safely with 100kmph");}

public static void main(String args[]){
Honda honda= new Honda();
honda.run();
}
}

```

Example of final class

```

final class Bike{ }

class Honda1 extends Bike{
void run(){System.out.println("running safely with 100kmph");}

public static void main(String args[]){
Honda1 honda= new Honda();
honda.run();
}
}

```

4. (i) Explain in details the concepts of inner classes.

Java inner class or nested class is a class i.e. declared inside the class or interface.

We use inner classes to logically group classes and interfaces in one place so that it can be more readable and maintainable.

Additionally, it can access all the members of outer class including private data members and methods.

Syntax of Inner class

```

class Java_Outer_class{
//code
class Java_Inner_class{
//code
}
}

```

Advantage of java inner classes

There are basically three advantages of inner classes in java. They are as follows:

- 1) Nested classes represent a special type of relationship that is **it can access all the members (data members and methods) of outer class** including private.
- 2) Nested classes are used **to develop more readable and maintainable code** because it logically group classes and interfaces in one place only.
- 3) **Code Optimization:** It requires less code to write.

Difference between nested class and inner class in Java

Inner class is a part of nested class. Non-static nested classes are known as inner classes.

Types of Nested classes

There are two types of nested classes non-static and static nested classes. The non-static nested classes are also known as inner classes.

1. Non-static nested class(inner class)
 - o a)Member inner class
 - o b)Annomynous inner class
 - o c)Local inner class
2. Static nested class

Type	Description
Member Inner Class	A class created within class and outside method.
Anonymous Inner Class	A class created for implementing interface or extending class. Its name is decided by the java compiler.
Local Inner Class	A class created within method.
Static Nested Class	A static class created within class.
Nested Interface	An interface created within class or interface.

(ii) Explain in details the concepts of applets.

Applet is a special type of program that is embedded in the webpage to generate the dynamic content. It runs inside the browser and works at client side.

Advantage of Applet

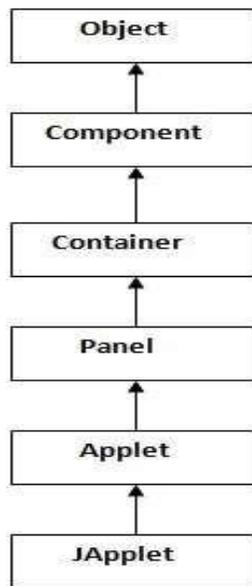
There are many advantages of applet. They are as follows:

- It works at client side so less response time.
- Secured
- It can be executed by browsers running under many plateforms, including Linux, Windows, Mac Os etc.

Drawback of Applet

- Plugin is required at client browser to execute applet.

Hierarchy of Applet



As displayed in the above diagram, Applet class extends Panel. Panel class extends Container which is the subclass of Component.

Lifecycle of Java Applet

1. Applet is initialized.
2. Applet is started.
3. Applet is painted.
4. Applet is stopped.
5. Applet is destroyed.
- 6.

Lifecycle methods for Applet:

The `java.applet.Applet` class provides 4 life cycle methods and `java.awt.Component` class provides 1 life cycle method for an applet.

`java.applet.Applet` class

For creating any applet `java.applet.Applet` class must be inherited. It provides 4 life cycle methods of applet.

1. **`public void init():`** is used to initialize the Applet. It is invoked only once.
2. **`public void start():`** is invoked after the `init()` method or browser is maximized. It is used to start the Applet.
3. **`public void stop():`** is used to stop the Applet. It is invoked when Applet is stopped or browser is minimized.
4. **`public void destroy():`** is used to destroy the Applet. It is invoked only once.

`java.awt.Component` class

The `Component` class provides 1 life cycle method of applet.

1. **`public void paint(Graphics g):`** is used to paint the Applet. It provides `Graphics` class object that

can be used for drawing oval, rectangle, arc etc.

How to run an Applet?

There are two ways to run an applet

1. By html file.
2. By appletViewer tool (for testing purpose).

Simple example of Applet by html file:

To execute the applet by html file, create an applet and compile it. After that create an html file and place the applet code in html file. Now click the html file.

```
//First.java
import java.applet.Applet;
import java.awt.Graphics;
public class First extends Applet{
public void paint(Graphics g){
g.drawString("welcome",150,150);
}
}
```

myapplet.html

```
<html>
<body>
<applet code="First.class" width="300" height="300">
</applet>
</body>
</html>
```

Simple example of Applet by appletviewer tool:

To execute the applet by appletviewer tool, create an applet that contains applet tag in comment and compile it. After that run it by: appletviewer First.java. Now Html file is not required but it is for testing purpose only.

```
//First.java
import java.applet.Applet;
import java.awt.Graphics;
public class First extends Applet{

public void paint(Graphics g){
g.drawString("welcome to applet",150,150);
}

}
/*
<applet code="First.class" width="300" height="300">
</applet>
*/
```

To execute the applet by appletviewer tool, write in command prompt:

```
c:\>javac First.java
```

	c:\>appletviewer First.java
5.	<p>What is Exception handling in java? Why it is used? Write a java code to simulate the way a stack mechanisms works with exception handling, throwing and dealing with exceptions such as stack is full(if you want to add more elements into the stack)or Stack is empty(you want to pop elements from the stack).</p> <p>An exception is a problem that arises during the execution of a program. An exception can occur for many different reasons, including the following:</p> <ul style="list-style-type: none"> • A user has entered invalid data. • A file that needs to be opened cannot be found. • A network connection has been lost in the middle of communications or the JVM has run out of memory. <p>Some of these exceptions are caused by user error, others by programmer error, and others by physical resources that have failed in some manner.</p> <p>To understand how exception handling works in Java, you need to understand the three categories of exceptions:</p> <ul style="list-style-type: none"> • Checked exceptions: A checked exception is an exception that is typically a user error or a problem that cannot be foreseen by the programmer. For example, if a file is to be opened, but the file cannot be found, an exception occurs. These exceptions cannot simply be ignored at the time of compilation. • Runtime exceptions: A runtime exception is an exception that occurs that probably could have been avoided by the programmer. As opposed to checked exceptions, runtime exceptions are ignored at the time of compilation. • Errors: These are not exceptions at all, but problems that arise beyond the control of the user or the programmer. Errors are typically ignored in your code because you can rarely do anything about an error. For example, if a stack overflow occurs, an error will arise. They are also ignored at the time of compilation. <pre> class SimpleStackDemo { public static void main(String arg[]) { FixedLengthStack stack=new FixedLengthStack(5); char ch; int i; try{ // overrun the stack for(i=0;i<6;i++) System.out.println("Attempting to push:"+(char) ('A'+i)); stack.push((char) ('A'+i)); System.out.println("=ok"); } catch(StackFullException exc) { </pre>

```

System.out.println(exc);
}
System.out.println();
try{
//over-empty the stack
for(i=0;i<6;i++){

System.out.println("Popping next char:");
ch=stack.pop();
System.out.println(ch);
}
}
Catch(stackEmptyException exc) {
System.out.println(exc);
}
}
}
}

```

6. Discuss the concept of synchronization in thread and develop a JAVA code for reader/writer problem.

When we start two or more threads within a program, there may be a situation when multiple threads try to access the same resource and finally they can produce unforeseen result due to concurrency issue. For example if multiple threads try to write within a same file then they may corrupt the data because one of the threads can overwrite data or while one thread is opening the same file at the same time another thread might be closing the same file.

So there is a need to synchronize the action of multiple threads and make sure that only one thread can access the resource at a given point in time. This is implemented using a concept called **monitors**. Each object in Java is associated with a monitor, which a thread can lock or unlock. Only one thread at a time may hold a lock on a monitor.

Java programming language provides a very handy way of creating threads and synchronizing their task by using **synchronized** blocks. You keep shared resources within this block. Following is the general form of the synchronized statement:

```

synchronized(objectidentifier) {
// Access shared variables and other shared resources
}

```

Here, the **objectidentifier** is a reference to an object whose lock associates with the monitor that the synchronized statement represents. Now we are going to see two examples where we will print a counter using two different threads. When threads are not synchronized, they print counter value which is not in sequence, but when we print counter by putting inside synchronized() block, then it prints counter very much in sequence for both the threads.

Multithreading example with Synchronization:

Here is the same example which prints counter value in sequence and every time we run it, it produces same result.

```
class PrintDemo {
    public void printCount(){
        try {
            for(int i = 5; i > 0; i--) {
                System.out.println("Counter --- " + i);
            }
        } catch (Exception e) {
            System.out.println("Thread interrupted.");
        }
    }
}

class ThreadDemo extends Thread {
    private Thread t;
    private String threadName;
    PrintDemo PD;

    ThreadDemo( String name, PrintDemo pd){
        threadName = name;
        PD = pd;
    }
    public void run() {
        synchronized(PD) {
            PD.printCount();
        }
        System.out.println("Thread " + threadName + " exiting.");
    }

    public void start ()
    {
        System.out.println("Starting " + threadName );
        if (t == null)
        {
            t = new Thread (this, threadName);
            t.start ();
        }
    }
}
```

```

public class TestThread {
    public static void main(String args[]) {

        PrintDemo PD = new PrintDemo();

        ThreadDemo T1 = new ThreadDemo( "Thread - 1 ", PD );
        ThreadDemo T2 = new ThreadDemo( "Thread - 2 ", PD );

        T1.start();
        T2.start();

        // wait for threads to end
        try {
            T1.join();
            T2.join();
        } catch( Exception e) {
            System.out.println("Interrupted");
        }
    }
}

```

This produces same result every time you run this program:

```

Starting Thread - 1
Starting Thread - 2
Counter --- 5
Counter --- 4
Counter --- 3
Counter --- 2
Counter --- 1
Thread Thread - 1 exiting.
Counter --- 5
Counter --- 4
Counter --- 3
Counter --- 2
Counter --- 1
Thread Thread - 2 exiting.

```

```

/**
 * Reader.java
 *
 * A reader to the database.
 */

```

```

class Reader implements Runnable
{

private RWLock database;
private int readerNum;

public Reader(int readerNum, RWLock database) {
    this.readerNum = readerNum;
    this.database = database;
}

public void run() {
    while (true) {
        SleepUtilities.nap();

        System.out.println("reader " + readerNum + " wants to read.");
        database.acquireReadLock(readerNum);

        // you have access to read from the database
        // let's read for awhile .....
        SleepUtilities.nap();

        database.releaseReadLock(readerNum);
    }
}
;
}

/**
*****
/**
 * Writer.java
 *
 * A writer to the database.
 */
class Writer implements Runnable
{
private RWLock database;
private int writerNum;

public Writer(int w, RWLock d) {
    writerNum = w;
    database = d;
}
}

```

```

public void run() {
    while (true){
        SleepUtilities.nap();

        System.out.println("writer " + writerNum + " wants to write.");
        database.acquireWriteLock(writerNum);

        // you have access to write to the database
        // write for awhile ...
        SleepUtilities.nap();

        database.releaseWriteLock(writerNum);
    }
}
}

```

7. (i) **Describe the concept of I/O with example.(8)**
Java I/O (Input and Output) is used to process the input and produce the output based on the input. Java uses the concept of stream to make I/O operation fast. The java.io package contains all the classes required for input and output operations. We can perform **file handling in java** by java IO API.

Stream

A stream is a sequence of data. In Java a stream is composed of bytes. It's called a stream because it's like a stream of water that continues to flow.

In java, 3 streams are created for us automatically. All these streams are attached with console.

1) **System.out:** standard output stream

2) **System.in:** standard input stream

3) **System.err:** standard error stream

Let's see the code to print **output and error** message to the console.

1. System.out.println("simple message");
2. System.err.println("error message");

Let's see the code to get **input** from console.

1. int i=System.in.read();//returns ASCII code of 1st character
2. System.out.println((char)i);//will print the character
- 3.

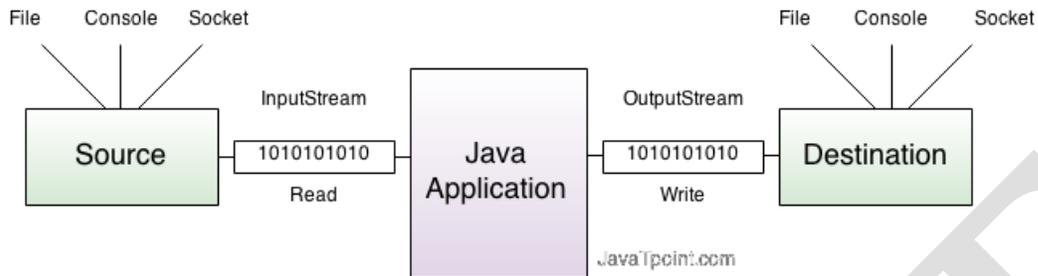
OutputStream

Java application uses an output stream to write data to a destination, it may be a file, an array, peripheral device or socket.

InputStream

Java application uses an input stream to read data from a source, it may be a file, an array, peripheral device or socket.

Let's understand working of Java OutputStream and InputStream by the figure given below.

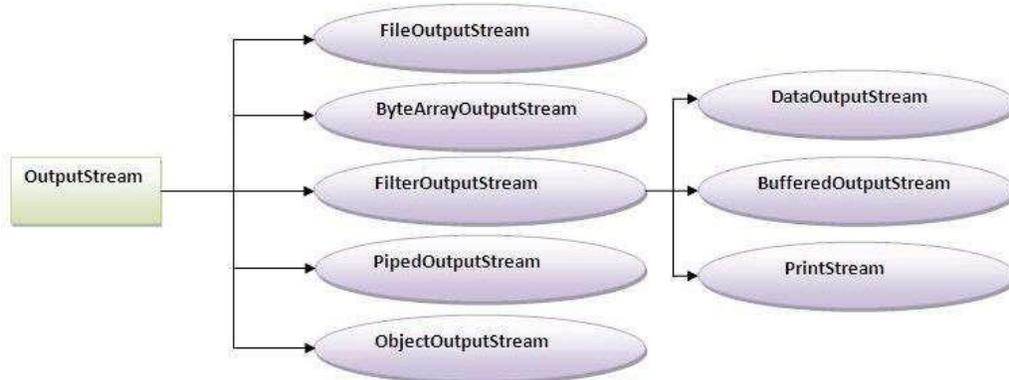


OutputStream class

OutputStream class is an abstract class. It is the superclass of all classes representing an output stream of bytes. An output stream accepts output bytes and sends them to some sink.

Commonly used methods of OutputStream class

Method	Description
1) public void write(int) throws IOException:	is used to write a byte to the current output stream.
2) public void write(byte[]) throws IOException:	is used to write an array of byte to the current output stream.
3) public void flush() throws IOException:	flushes the current output stream.
4) public void close() throws IOException:	is used to close the current output stream.

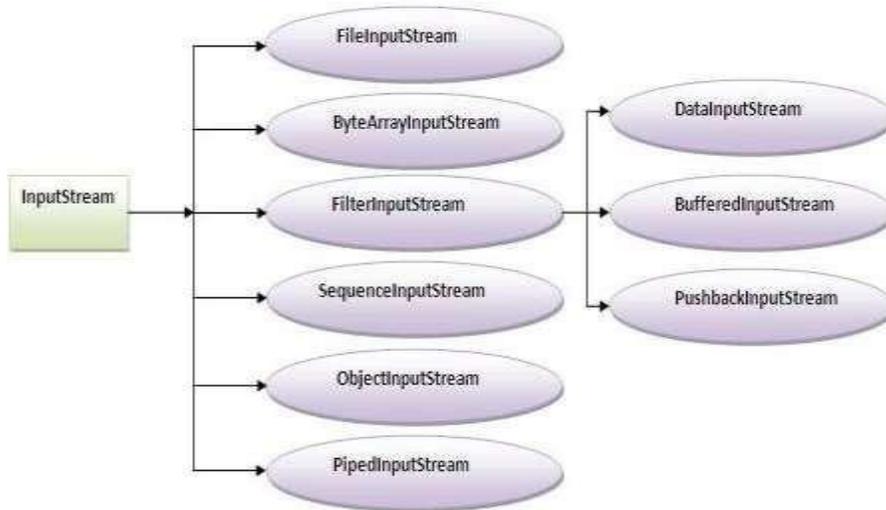


InputStream class

InputStream class is an abstract class. It is the superclass of all classes representing an input stream of bytes.

Commonly used methods of InputStream class

Method	Description
1) public abstract int read() throws IOException:	reads the next byte of data from the input stream. It returns -1 at the end of file.
2) public int available() throws IOException:	returns an estimate of the number of bytes that can be read from the current input stream.
3) public void close() throws IOException:	is used to close the current input stream.



(ii) Explain the detail about string handling.(8).

Java String provides a lot of concepts that can be performed on a string such as compare, concat, equals, split, length, replace, compareTo, intern, substring etc.

In java, string is basically an object that represents sequence of char values.

An array of characters works same as java string. For example:

1. `char[] ch={'j','a','v','a','t','p','o','i','n','t'};`
2. `String s=new String(ch);`

is same as:

1. `String s="javatpoint";`

The `java.lang.String` class implements *Serializable*, *Comparable* and *CharSequence* interfaces.

The java String is immutable i.e. it cannot be changed but a new instance is created. For mutable class, you can use `StringBuffer` and `StringBuilder` class.

Generally, string is a sequence of characters. But in java, string is an object that represents a sequence of characters. String class is used to create string object.

There are two ways to create String object:

1. By string literal
2. By new keyword

1) String Literal

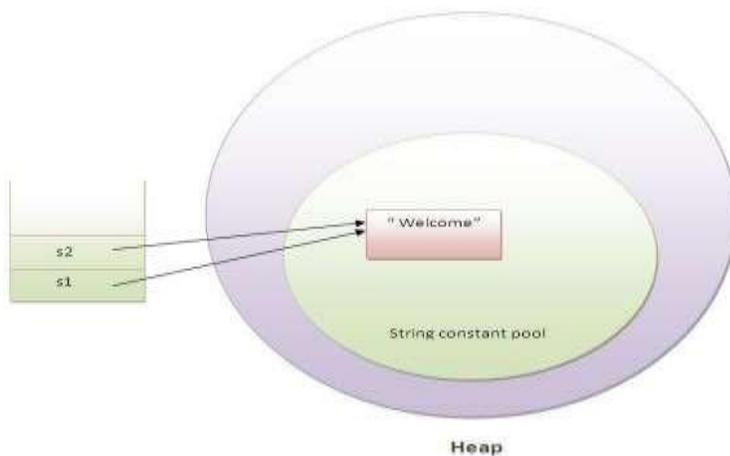
Java String literal is created by using double quotes. For Example:

1. `String s="welcome";`

Each time you create a string literal, the JVM checks the string constant pool first. If the string already exists in the pool, a reference to the pooled instance is returned. If string doesn't exist in the pool, a new string instance is created and placed in the pool. For example:

1. `String s1="Welcome";`

2. `String s2="Welcome";`//will not create new instance



2) By new keyword

1. `String s=new String("Welcome");`//creates two objects and one reference variable

Java String Example

```
public class StringExample{
public static void main(String args[]){
String s1="java";//creating string by java string literal

char ch[]={ 's','t','r','i','n','g','s' };
String s2=new String(ch);//converting char array to string

String s3=new String("example");//creating java string by new keyword

System.out.println(s1);
System.out.println(s2);
System.out.println(s3);
}}
Output:
```

java
strings
example

8. **What is meant by constructors? Describe the type of constructors supported by java with example.**
Constructor in java is a *special type of method* that is used to initialize the object. Java constructor is *invoked at the time of object creation*. It constructs the values i.e. provides data for the object that is why it is known as constructor.

Rules for creating java constructor

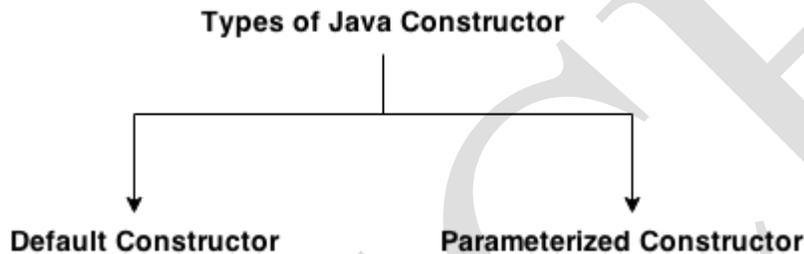
There are basically two rules defined for the constructor.

1. Constructor name must be same as its class name
2. Constructor must have no explicit return type

Types of java constructors

There are two types of constructors:

1. Default constructor (no-arg constructor)
2. Parameterized constructor



Java Default Constructor

A constructor that have no parameter is known as default constructor.

Syntax of default constructor:

```
<class_name>(){ }
```

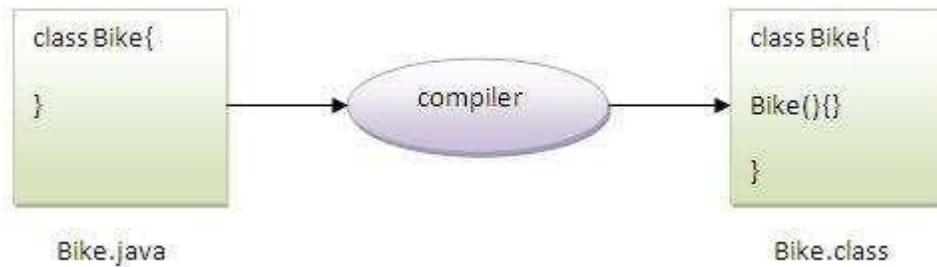
Example of default constructor

In this example, we are creating the no-arg constructor in the Bike class. It will be invoked at the time of object creation.

```
class Bike1{
    Bike1(){System.out.println("Bike is created");}
    public static void main(String args[]){
        Bike1 b=new Bike1();
    }
}
```

Output:

Bike is created



Java parameterized constructor

A constructor that has parameters is known as a parameterized constructor.

A parameterized constructor is used to provide different values to distinct objects.

In this example, we have created the constructor of the Student class that has two parameters. We can have any number of parameters in the constructor.

```

class Student4{
  int id;
  String name;
  Student4(int i,String n){
    id = i;
    name = n;
  }
  void display(){System.out.println(id+" "+name);}
  public static void main(String args[]){
    Student4 s1 = new Student4(111,"Karan");
    Student4 s2 = new Student4(222,"Aryan");
    s1.display();
    s2.display();
  }
}
    
```

Output:

111 Karan

222 Aryan

Constructor Overloading in Java

Constructor overloading is a technique in Java in which a class can have any number of constructors that differ in parameter lists. The compiler differentiates these constructors by taking into account the number of parameters in the list and their type.

```

class Student5{
  int id;
  String name;
  int age;
  Student5(int i,String n){
    id = i;
    name = n;
  }
}
    
```

```

}
Student5(int i,String n,int a){
id = i;
name = n;
age=a;
}
void display(){System.out.println(id+" "+name+" "+age);}

public static void main(String args[]){
Student5 s1 = new Student5(111,"Karan");
Student5 s2 = new Student5(222,"Aryan",25);
s1.display();
s2.display();
}
}

```

Output:

```

111 Karan 0
222 Aryan 25

```

Java Copy Constructor

There is no copy constructor in java. But, we can copy the values of one object to another like copy constructor in C++.

There are many ways to copy the values of one object into another in java. They are:

- By constructor
- By assigning the values of one object into another
- By clone() method of Object class

In this example, we are going to copy the values of one object into another using java constructor.

```

class Student6{
int id;
String name;
Student6(int i,String n){
id = i;
name = n;
}

Student6(Student6 s){
id = s.id;
name =s.name;
}
void display(){System.out.println(id+" "+name);}
}

```

```

public static void main(String args[]){
    Student6 s1 = new Student6(111,"Karan");
    Student6 s2 = new Student6(s1);
    s1.display();
    s2.display();
}
}

```

Test it Now

Output:

```

111 Karan
111 Karan

```

9. Define inheritances. Explain in details types of inheritances supported by JAVA with example program.

Inheritance in java is a mechanism in which one object acquires all the properties and behaviors of parent object.

The idea behind inheritance in java is that you can create new classes that are built upon existing classes. When you inherit from an existing class, you can reuse methods and fields of parent class, and you can add new methods and fields also.

Inheritance represents the **IS-A relationship**, also known as *parent-child* relationship.

use inheritance in java

- For Method Overriding (so runtime polymorphism can be achieved).
- For Code Reusability.

Syntax of Java Inheritance

```

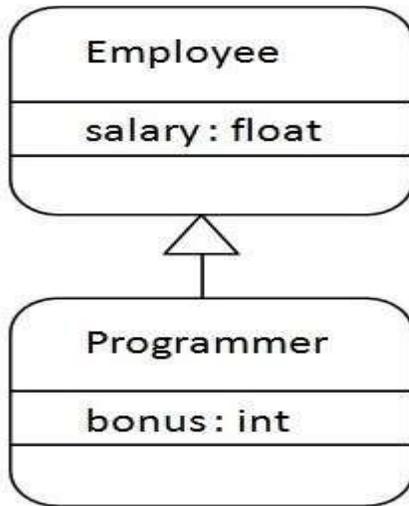
class Subclass-name extends Superclass-name
{
    //methods and fields
}

```

The **extends keyword** indicates that you are making a new class that derives from an existing class.

In the terminology of Java, a class that is inherited is called a super class. The new class is called a subclass.

Understanding the simple example of inheritance



As displayed in the above figure, Programmer is the subclass and Employee is the superclass. Relationship between two classes is **Programmer IS-A Employee**. It means that Programmer is a type of Employee.

```

class Employee{
float salary=40000;
}
class Programmer extends Employee{
int bonus=10000;
public static void main(String args[]){ Programmer
p=new Programmer(); System.out.println("Programmer
salary is:"+p.salary); System.out.println("Bonus of
Programmer is:"+p.bonus);
}
}
  
```

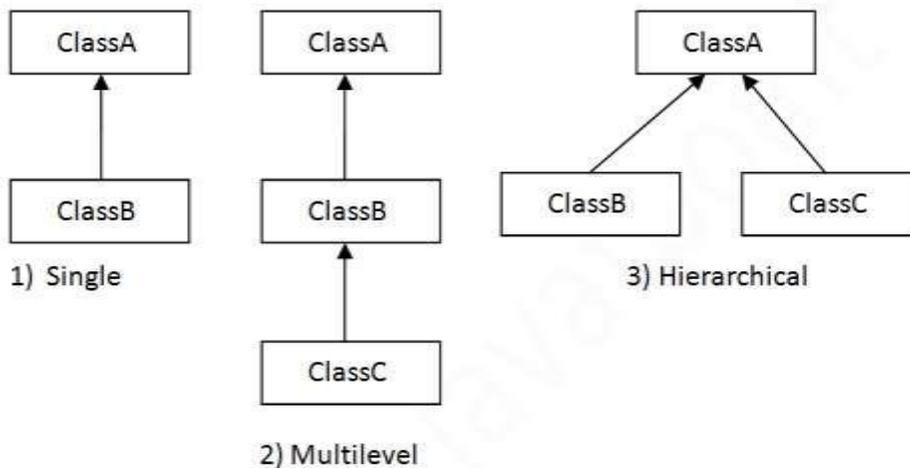
```

Programmer salary is:40000.0
Bonus of programmer is:10000
  
```

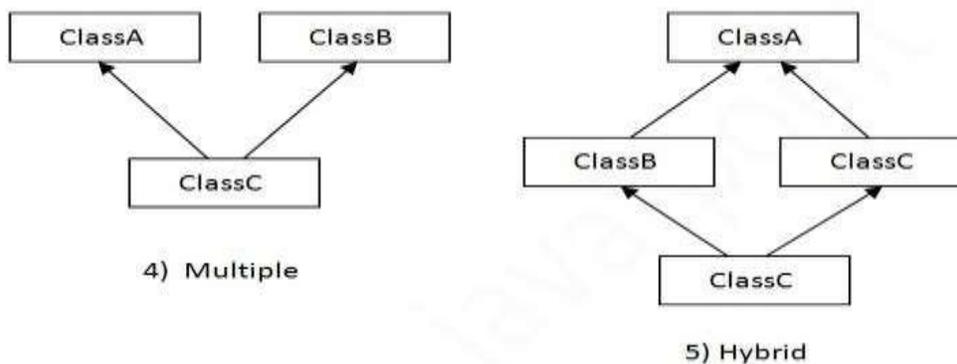
Types of inheritance in java

On the basis of class, there can be three types of inheritance in java: single, multilevel and hierarchical.

In java programming, multiple and hybrid inheritance is supported through interface only. We will learn about interfaces later.



When a class extends multiple classes i.e. known as multiple inheritance. For Example:



Why multiple inheritance is not supported in java?

To reduce the complexity and simplify the language, multiple inheritance is not supported in java.

Consider a scenario where A, B and C are three classes. The C class inherits A and B classes. If A and B classes have same method and you call it from child class object, there will be ambiguity to call method of A or B class.

Since compile time errors are better than runtime errors, java renders compile time error if you inherit 2 classes. So whether you have same method or different, there will be compile time error now.

```

class A{
void msg(){System.out.println("Hello");}
}
class B{
void msg(){System.out.println("Welcome");}
}
  
```

```
class C extends A,B{//suppose if it were
```

```
Public Static void main(String args[]){
  C obj=new C();
  obj.msg();//Now which msg() method would be invoked?
}
}
```

Compile Time Error

10. Explain in details stream classes with suitable example.

The java.io package contains nearly every class you might ever need to perform input and output (I/O) in Java. All these streams represent an input source and an output destination. The stream in the java.io package supports many data such as primitives, Object, localized characters, etc.

A stream can be defined as a sequence of data. The InputStream is used to read data from a source and the OutputStream is used for writing data to a destination.

Java provides strong but flexible support for I/O related to Files and networks but this tutorial covers very basic functionality related to streams and I/O. We would see most commonly used example one by one:

Byte Streams

Java byte streams are used to perform input and output of 8-bit bytes. Though there are many classes related to byte streams but the most frequently used classes are , **FileInputStream** and **FileOutputStream**. Following is an example which makes use of these two classes to copy an input file into an output file:

```
import java.io.*;

public class CopyFile {
  public static void main(String args[]) throws IOException
  {
    FileInputStream in = null;
    FileOutputStream out = null;

    try {
      in = new FileInputStream("input.txt");
      out = new FileOutputStream("output.txt");

      int c;
      while ((c = in.read()) != -1) {
        out.write(c);
      }
    } finally {
      if (in != null) {
        in.close();
      }
    }
  }
}
```

```

    }
    if (out != null) {
        out.close();
    }
}
}
}
}
}

```

Now let's have a file **input.txt** with the following content:

This is test for copy file.

As a next step, compile above program and execute it, which will result in creating output.txt file with the same content as we have in input.txt. So let's put above code in CopyFile.java file and do the following:

```

$javac CopyFile.java
$java CopyFile

```

Character Streams

Java **Byte** streams are used to perform input and output of 8-bit bytes, where as Java **Character** streams are used to perform input and output for 16-bit unicode. Though there are many classes related to character streams but the most frequently used classes are , **FileReader** and **FileWriter**.. Though internally FileReader uses FileInputStream and FileWriter uses FileOutputStream but here major difference is that FileReader reads two bytes at a time and FileWriter writes two bytes at a time.

We can re-write above example which makes use of these two classes to copy an input file (having unicode characters) into an output file:

```

import java.io.*;

public class CopyFile {
    public static void main(String args[]) throws IOException
    {
        FileReader in = null;
        FileWriter out = null;

        try {
            in = new FileReader("input.txt");
            out = new FileWriter("output.txt");

            int c;
            while ((c = in.read()) != -1) {
                out.write(c);
            }
        } finally {

```

```

    if (in != null) {
        in.close();
    }
    if (out != null) {
        out.close();
    }
}
}
}
}
}

```

Now let's have a file **input.txt** with the following content:

This is test for copy file.

As a next step, compile above program and execute it, which will result in creating output.txt file with the same content as we have in input.txt. So let's put above code in CopyFile.java file and do the following:

```

$javac CopyFile.java
$java CopyFile

```

Standard Streams

All the programming languages provide support for standard I/O where user's program can take input from a keyboard and then produce output on the computer screen. If you are aware of C or C++ programming languages, then you must be aware of three standard devices STDIN, STDOUT and STDERR. Similar way Java provides following three standard streams

- **Standard Input:** This is used to feed the data to user's program and usually a keyboard is used as standard input stream and represented as **System.in**.
- **Standard Output:** This is used to output the data produced by the user's program and usually a computer screen is used to standard output stream and represented as **System.out**.
- **Standard Error:** This is used to output the error data produced by the user's program and usually a computer screen is used to standard error stream and represented as **System.err**.

Following is a simple program which creates **InputStreamReader** to read standard input stream until the user types a "q":

```

import java.io.*;

public class ReadConsole {
    public static void main(String args[]) throws IOException
    {
        InputStreamReader cin = null;

        try {
            cin = new InputStreamReader(System.in);

```

```

System.out.println("Enter characters, 'q' to quit.");
char c;
do {
    c = (char) cin.read();
    System.out.print(c);
} while(c != 'q');
}finally {
    if (cin != null) {
        cin.close();
    }
}
}
}

```

Let's keep above code in ReadConsole.java file and try to compile and execute it as below. This program continues reading and outputting same character until we press 'q':

```

$javac ReadConsole.java
$java ReadConsole
Enter characters, 'q' to quit.
l
l
e
e
q
q

```

Unit-II Part-A

1. **What is web 2.0?**
A Web 2.0 site may allow users to interact and collaborate with each other in a social media dialogue as creators of user-generated content in a virtual community, in contrast to Web sites where people are limited to the passive viewing of content. Examples of Web 2.0 include social networking sites, blogs, wikis, folksonomies, video sharing sites, hosted services, Web applications, and mashups.
2. **Define RIA.**
A rich Internet application (RIA) is a Web application designed to deliver the same features and functions normally associated with desktop applications. RIAs generally split the processing across the Internet/network divide by locating the user interface and related activity and capability on the client side, and the data manipulation and operation on the application server side.
3. **Define collaboration.**
Collaboration is a process defined by the recursive interaction of knowledge and mutual learning between two or more people who are working together, in an intellectual endeavour, toward a common goal which is typically creative in nature.
4. **List the Collaborations tools.**
AnswerGaeden, Thinkature, DotVoting, ePals, Gaggle, Glass, Tricider.
5. **What are the collaborative processes.**
 - Team Creation

	<ul style="list-style-type: none"> • Idea Generation • Decision-Making • Work or Production • Evaluation or Recap
6.	<p>Define Web services. A <i>Web service</i> is a method of communication between two electronic devices over a network. It is a software function provided at a network address over the Web with the service <i>always on</i> as in the concept of utility computing.</p>
7.	<p>Write short notes on Software as service(Soas). SOAs : Software as a service (SaaS), sometimes referred to as "software on demand," is software that is deployed over the internet and/or is deployed to run behind a firewall on a local area network or personal computer. With SaaS, a provider licenses an application to customers either as a service on demand, through a subscription, in a "pay-as-you-go" model, or (increasingly) at no charge.</p>
8.	<p>Write short notes on Social networking. A social network is a social structure made up of a set of social actors (such as individuals or organizations) and a set of the dyadic ties between these actors. The social network perspective provides a set of methods for analyzing the structure of whole social entities as well as a variety of theories explaining the patterns observed in these structures.</p>
9.	<p>Define Website. A website is hosted on at least one web server, accessible via a network such as the Internet or a private local area network through an Internet address known as a uniform resource locator (URL). All publicly accessible websites collectively constitute the World Wide Web</p>
10.	<p>Differences between web sites and web server. Website: A website is a set of linked documents associated with a particular person, organization or topic that is held on a computer system and can be accessed as part of the world wide web. (Not to be confused with: Web page, a document on the world wide web written in HTML and displayed in a web browser.) Web server: The web server on the other side is a computer program, which delivers content, such as websites or web pages, for example, over the world wide web from a web server to your computer.</p>
11.	<p>Define internet. The Internet is a global system of interconnected computer networks that use the standard Internet protocol suite (TCP/IP) to link several billion devices worldwide. It is a network of networks that consists of millions of private, public, academic, business, and government networks of local to global scope, linked by a broad array of electronic, wireless, and optical networking technologies.</p>
12.	<p>Define intranet. An intranet is a computer network that uses Internet Protocol technology to share information, operational systems, or computing services within an organization. This term is used in contrast to extranet, a network between organizations, and instead refers to a network within an organization. Sometimes, the term refers only to the organization's internal website, but may be a more extensive part of the organization's information technology infrastructure, and may be composed of multiple local area networks. The objective is to organize each individual's desktop with minimal cost, time and effort to be more productive, cost efficient, timely, and competitive.</p>
13.	<p>Differentiate between internet and intranet.</p> <ul style="list-style-type: none"> • Internet is general to PCs all over the world whereas Intranet is specific to few PCs. • Internet has wider access and provides a better access to websites to large population whereas Intranet is restricted. • Internet is not as safe as Intranet as Intranet can be safely privatized as per the need.
14.	<p>Define HTML. HTML is a simple web page description language, which enables document creation for the web. HTML is the set</p>

	of mark-up symbols or codes placed in a file intended for display on the web browser page. These mark-up symbol and codes identify structural elements such as paragraphs, heading, and lists. HTML can be used to place media (such as graphics, video, and audio) on the Web page and describe fill-in-forms. A method is an implementation of an objects behavior.									
15.	Explain about HTTP Connection. It is a communication channel between web browser and web server. It begins on the client side with the browser sending a request to the web server for a document. Request Header Fields are 1. From 2. Reference 3. If_modified_since 4. Pragma 5. User Agent									
16.	Define cascading. Cascading refers to a certain set of rules that browsers use, in cascading order, to determine how to use the style information. Such a set of rules is useful in the event of conflicting style information because the rules would give the browser a way to determine which style is given precedence.									
17.	State the use of web server logs and list the contents of a message log. (APR/MAY 2011) A server log is a log file (or several files) automatically created and maintained by a server of activity performed by it. A typical example is a web server log which maintains a history of page requests. The W3C maintains a standard format (the Common Log Format) for web server log files, but other proprietary formats exist. The message log is used by a number of processes to provide debugging and troubleshooting information. You can view the message log from the process monitor after clicking on the details hyperlink for a process and the by clicking on the message log hyperlink in the actions area.									
18.	How will you create a password field in a HTML form? (NOV/DEC 2011) <code><input type="password" name="pwd" size="15"></code>									
19.	List any four common browsers. (NOV/DEC 2011) <ul style="list-style-type: none"> • Google Chrome • Netscape Navigator • Microsoft Internet Explorer • Mozilla 									
20.	State the uses of internet protocol. (APR/MAY 2012) <ul style="list-style-type: none"> • IP function: transfer data from source device to destination device • IP source software creates a packet representing the data • Header: source and destination IP addresses, length of data, etc. • Data: Data itself 									
21.	Define Tags. What are the two different types of tags? Tags signal the browser to inform about the formatting details.ie how the content should be displayed in the browser screen. Tags are enclosed between "<" and ">" Standalone tag only start tag is present and no end tag. Example and container tag have start and end tag will be present .Example <html>.... </html>									
22.	What are the rules to define a tag? Attributes should be placed inside start tag, appears as Name-value pairs separated by blank spaces, Attributes should have only one value, values should be enclosed within either single(') or double (") quotes.									
23.	Differentiate between standalone and container tag. <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 5%;">S.no</th> <th style="width: 50%;">Standalone</th> <th style="width: 45%;">Container</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Only start tag is present and no end tag.</td> <td>Both start and end tag will be present</td> </tr> <tr> <td>2</td> <td>Can have only attributes and no parameters</td> <td>Can have both attributes and parameters.</td> </tr> </tbody> </table>	S.no	Standalone	Container	1	Only start tag is present and no end tag.	Both start and end tag will be present	2	Can have only attributes and no parameters	Can have both attributes and parameters.
S.no	Standalone	Container								
1	Only start tag is present and no end tag.	Both start and end tag will be present								
2	Can have only attributes and no parameters	Can have both attributes and parameters.								

	3	Example: 	Example:<html>.....</html>
24.	What is the use of <pre> tag in HTML? The pre tag can be used to preserve the white spaces and lines in the text.		
25.	What is cellpadding and cell spacing attributes? The cellpadding allows to have some space between the contents of each cell and its borders. The distance between each cell is called cell spacing.		
26.	What is the need of using form in HTML? Form is a typical layout on the web page by which user can interact with the web page. The components that can be placed on the form are text box, check box, radio buttons, and push buttons and so on. Thus form is typically used to create an interactive Graphical User Interface.		
27.	What is the purpose of using frames in HTML? The HTML frames allows the web designer to present the web document in multiple views. Using multiple views one can keep the formation visible and at the same time other views can be scrolled or replaced.		
28.	What is the need for special character in HTML? There are some symbols that cannot be used directly in HTML document. For example <(less than) because this symbol is also used along with the tag. Hence this is called a special symbol and can be denoted with the help of entity reference.		
29.	State how an unrecognized element or attribute treated by the HTML document? If any unrecognized element or attribute is written then the HTML document simply displays the contents. For example <title>testing</title> will display the string “testing” on the web page. It will not display it as a title of the web page.		
30.	What is the use of hyperlink tag in HTML? The hyperlink tag is used to link logically with other page. Using this tag a web link can be specified. The <a> tag is used to specify the hyperlink in HTML.		
31.	What are the uses of hyperlink in HTML? To logically link one page with another, use of link to enhance readability of the web document, the navigation from one page to another is possible.		
32.	What is BODY in HTML document? The effects which we want in the window are mentioned with the help of tags in the body. It is the place where the actual data is written in html. All the changes can be viewed by changing the tags content in the body whereas the head part is the introduction part and the body is the actual content part.<BODY>data content</BODY>		
33.	What is an image map? An image map allows you to link to several web pages through one image. Simply define shapes within images and link these to the pages you want. Here’s a video to help you learn more about images and links in HTML.		
34.	What are style sheets? The style sheets are the collection of styles that can be either embedded within the HTML documents or can be externally applied. The Cascading style sheet is a markup language used to apply the styles to HTML elements.		
35.	What is selector string? Specify any three forms of selectors. The rule set in CSS consists of selector string which is basically an HTML element. These selectors can be defined with the help of properties and values.		
36.	What is the use of Universal Selector? Using the universal selector the values can be defined for all the elements in the document. It is denoted by *.		
37.	What is generic class selector? The generic class applied to any tag in the HTML document. And thus the values defined within that generic selector can be applied to the corresponding tag. The class selector must be preceded by the dot operator.		
38.	What are the advantages of External style sheet? When we use external style sheet then the style is defined in one file and actual contents of the web are defined in another file. Hence if we want to change the style of presentation of web page then we can simply modify the file in which the style is defined.		

39.	<p>What is the difference the external style sheet and embedded style sheet?</p> <p>The external style sheet is a kind of style sheet in which the styles are defined in a separate.css file and this file is mentioned at the beginning of the HTML document. When we need to apply the particular style to more than one web documents then the external style sheet is used. The embedded style sheet is a method in which the style is specified within the HTML document itself. It is not defined in separate file. Due to embedded style sheet unique style can be applied to all the elements.</p>
40.	<p>What do you mean by the term inline element?</p> <p>The inline elements are those elements that do not form new blocks of content. The content is distributed in lines.</p>
41.	<p>What are the various style sheets?</p> <p>Inline, external, imported and embedded are the different types of style sheets.</p>
42.	<p>Explain inline, embedded and external style sheets.</p> <p>Inline If only a small piece of code has to be styled then inline style sheets can be used.</p> <p>Embedded Embedded style sheets are put between the <head> </head> tags.</p> <p>External If you want to apply a style to all the pages within your website by changing just one <i>style sheet, then external style sheets can be used.</i></p>
43.	<p>Give example for inline style sheet. (APR/MAY 2013)</p> <pre><h2>InLINE CSS</h2> <p style="color:sienna;margin-left:20px"></pre> <p>The style ATTRIBUTE we are able to modify the appearance of HTML elements </p></p>
44.	<p>How will you embed the external style sheet? (May 2014)</p> <p>In external style sheet is ideal when the style is applied to many pages. With an external style sheet, you can change the look of an entire Web site by changing just one file.Each page must include a link to the style sheet with the <link> tag. The <link> tag goes inside the head section:</p> <pre><head> <link rel="stylesheet" type="text/css" href="mystyle.css"> </head></pre> <p>An external style sheet can be written in any text editor. The file should not contain any html tags. The style sheet file must be saved with a .css extension. An example of a style sheet file is shown below:</p> <p>"myStyle.css":</p> <pre>body { background-color: lightblue;} h1 { color: navy; margin-left: 20px;}</pre>
45.	<p>How will you include CSS in a web site? (MAY/JUNE 2014)</p> <p>Inline</p> <p>Inline styles are when you just insert the type of style you want inside another tag, using the style attribute. This is usually the least useful way to use CSS.</p> <pre><p style="width:100%; color:#660099; text-align:right; background-color:#ffcc00;" ></pre> <p>Embedded</p> <p>Styles can also be placed in the document using the <style> tag. The <style> tag is usually placed in the head section of the document, where it will apply to the whole document.</p> <pre><style> <!-- p { color:#009900; font-family:"comic sans ms",sans-serif; } h1 { color:#660000; font-size:12pt; } </style></pre>

	<p>External styles Styles can also be set in an external style sheet which is linked to the page with a <link> tag. For example the style sheet for this site is included like this: <code><link rel="stylesheet" type="text/css" href="class.css" /></code></p>							
<p>46.</p>	<p>What is the purpose of CSS Box Model and mention its parts also. The CSS box model is essentially a box that wraps around HTML elements, and it consists of: margins, borders, padding, and the actual content. The different parts are:</p> <ul style="list-style-type: none"> • Margin • Border • Padding • Content 							
<p>Part-B</p>								
<p>1.</p>	<p>Explain WWW and HTTP Protocol. WWW The term WWW refers to the World Wide Web or simply the Web. The World Wide Web consists of all the public Web sites connected to the Internet worldwide, including the client devices (such as computers and cell phones) that access Web content. The WWW is just one of many applications of the Internet and computer networks. The World Web is based on these technologies:</p> <ol style="list-style-type: none"> 1. HTML - Hypertext Markup Language 2. HTTP - Hypertext Transfer Protocol 3. Web servers and Web browsers <p>HTTP HTTP (Hypertext Transfer Protocol) is the set of rules for transferring files (text, graphic images, sound, video, and other multimedia files) on the World Wide Web. HTTP is an application protocol that runs on top of the TCP/IP suite of protocols. HTTP concepts include (as the Hypertext part of the name implies) the idea that files can contain references to other files whose selection will elicit additional transfer requests. Any Web server machine contains, in addition to the Web page files it can serve, an HTTP daemon, a program that is designed to wait for HTTP requests and handle them when they arrive. Your Web browser is an HTTP client, sending requests to server machines. When the browser user enters file requests by either "opening" a Web file (typing in a Uniform Resource Locator or URL) or clicking on a hypertext link, the browser builds an HTTP request and sends it to the Internet Protocol address (IP address) indicated by the URL. The HTTP daemon in the destination server machine receives the request and sends back the requested file or files associated with the request. (A Web page often consists of more than one file.)</p>							
<p>2.</p>	<p>Discuss the structure of the HTTP request message. (NOV/DEC 2012) Structure of the request:</p> <table border="1" style="margin-left: 40px;"> <tr><td>start line</td></tr> <tr><td>header field(s)</td></tr> <tr><td>blank line</td></tr> <tr><td>optional body</td></tr> </table> <p>1. Start line Example: GET / HTTP/1.1 Three space-separated parts:</p> <table border="1" style="margin-left: 40px;"> <tr><td>HTTP request method</td></tr> <tr><td>Request-URI</td></tr> <tr><td>HTTP version</td></tr> </table> <p><i>Request URI</i> Uniform Resource Identifier (URI) Syntax: <i>scheme : scheme-depend-part</i></p>	start line	header field(s)	blank line	optional body	HTTP request method	Request-URI	HTTP version
start line								
header field(s)								
blank line								
optional body								
HTTP request method								
Request-URI								
HTTP version								

Ex: In <http://www.example.com/> the scheme is http
 Request-URI is the portion of the requested URI that follows the host name (which is supplied by the required Host header field)

Ex: / is Request-URI portion of <http://www.example.com/>

Request methods:

GET

- Used if link is clicked or address typed in browser
- No body in request with GET method

POST

- Used when submit button is clicked on a form
- Form information contained in body of request

HEAD

- Requests that only header fields (no body) be returned in the response
- PUT
- DELETE
- TRACE
- OPTIONS

2. Header field structure:

– *field name : field value*

Syntax

- Field name is not case sensitive
- Field value may continue on multiple lines by starting continuation lines with white space
- Field values may contain MIME types, quality values, and wildcard characters (*'s)

3. MIME

- Convention for specifying content type of a message
- In HTTP, typically used to specify content type of the body of the response
- MIME content type syntax:
- *top-level type / subtype*
- Examples: text/html, image/jpeg
- Example header field with quality values:
 accept:
 text/xml;text/html;q=0.9,
 text/plain;q=0.8, image/jpeg,
 image/gif;q=0.2,*/*;q=0.1
- Quality value applies to all preceding items
- Higher the value, higher the preference
- Note use of wildcards to specify quality 0.1 for any MIME type not specified earlier

4. Common header fields:

- Host: host name from URL (required)
- User-Agent: type of browser sending request
- Accept: MIME types of acceptable documents
- Connection: value close tells server to close connection after single request/response
- Content-Type: MIME type of (POST) body, normally application/x-www-form-urlencoded
- Content-Length: bytes in body
- Referer: URL of document containing link that supplied URI for this HTTP request

3. Discuss the structure of the HTTP response message.[8] (NOV/DEC 2012)

- *Structure of the response*

status line
header field(s)
blank line

	optional body												
<ul style="list-style-type: none"> • <i>Status line</i> <ul style="list-style-type: none"> – Example: HTTP/1.1 200 OK – Three space-separated parts: <table border="1"> <tr> <td>HTTP version</td> </tr> <tr> <td>status code</td> </tr> <tr> <td>reason phrase (intended for human use)</td> </tr> </table> • <i>Status code</i> <ul style="list-style-type: none"> – Three-digit number – First digit is class of the status code: <table border="1"> <tr> <td>1</td> <td>Informational</td> </tr> <tr> <td>2</td> <td>Success</td> </tr> <tr> <td>3</td> <td>Redirection (alternate URL is supplied)</td> </tr> <tr> <td>4</td> <td>Client Error</td> </tr> <tr> <td>5</td> <td>Server Error</td> </tr> </table> – Other two digits provide additional information • <i>Header Fields</i> <ul style="list-style-type: none"> – Connection, Content-Type, Content-Length – Date: date and time at which response was generated (required) – Location: alternate URI if status is redirection – Last-Modified: date and time the requested resource was last modified on the server – Expires: date and time after which the client's copy of the resource will be out-of-date – ETag: a unique identifier for this version of the requested resource (changes if resource changes) • <i>Client Caching</i> <ul style="list-style-type: none"> – A cache is a local copy of information obtained from some other source – Most web browsers use cache to store requested resources so that subsequent requests to the same resource will not necessarily require an HTTP request/response • Ex: icon appearing multiple times in a Web page • <i>Cache advantages</i> <ul style="list-style-type: none"> – (Much) faster than HTTP request/response – Less network traffic – Less load on server • <i>Cache disadvantage</i> <ul style="list-style-type: none"> – Cached copy of resource may be invalid (inconsistent with remote version) • <i>Validating cached resource:</i> <ul style="list-style-type: none"> – Send HTTP HEAD request and check Last-Modified or ETag header in response – Compare current date/time with Expires header sent in response containing resource – If no Expires header was sent, use heuristic algorithm to estimate value for Expires – Ex: Expires = 0.01 * (Date – Last-Modified) + Date • <i>Character sets</i> <ul style="list-style-type: none"> – Every document is represented by a string of integer values (code points) – The mapping from code points to characters is defined by a character set – Some header fields have character set values: <ul style="list-style-type: none"> – Accept-Charset: request header listing character sets that the client can recognize • Ex: accept-charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7 <ul style="list-style-type: none"> – Content-Type: can include character set used to represent the body of the HTTP message • Ex: Content-Type: text/html; charset=UTF-8 <ul style="list-style-type: none"> – Technically, many “character sets” are actually character encodings 	HTTP version	status code	reason phrase (intended for human use)	1	Informational	2	Success	3	Redirection (alternate URL is supplied)	4	Client Error	5	Server Error
HTTP version													
status code													
reason phrase (intended for human use)													
1	Informational												
2	Success												
3	Redirection (alternate URL is supplied)												
4	Client Error												
5	Server Error												

	<ul style="list-style-type: none"> - An encoding represents code points using variable-length byte strings - Most common examples are Unicode-based encodings UTF-8 and UTF-16 <p>IANA maintains complete list of Internet-recognized character sets/encodings</p>
4.	<p>Explain HTML elements in detail also State the types of lists supported by HTML and explain them in detail. (APR/MAY 2011)</p> <p><u>HTML element</u></p> <p>An HTML element is an individual component of an HTML document or web page, once this has been parsed into the Document Object Model. HTML is composed of a tree of HTML elements and other nodes, such as text nodes. Each element can have HTML attributes specified. Elements can also have content, including other elements and text. HTML elements represent semantics, or meaning. For example, the title element represents the title of the document.</p> <p>Heading Tags</p> <p>Any document starts with a heading. You can use different sizes for your headings. HTML also has six levels of headings, which use the elements <h1>, <h2>, <h3>, <h4>, <h5>, and <h6>. While displaying any heading, browser adds one line before and one line after that heading.</p> <p><i>Example</i></p> <pre><h1>This is heading 1</h1> <h2>This is heading 2</h2> <h3>This is heading 3</h3> <h4>This is heading 4</h4> <h5>This is heading 5</h5> <h6>This is heading 6</h6></pre> <p>Paragraph Tag</p> <p>The <p> tag offers a way to structure your text into different paragraphs. Each paragraph of text should go in between an opening <p> and a closing </p> tag as shown below in the example:</p> <p><i>Example</i></p> <pre><p>Here is a first paragraph of text.</p></pre> <p>Line Break Tag</p> <p>Whenever you use the
 element, anything following it starts from the next line. This tag is an example of an empty element, where you do not need opening and closing tags, as there is nothing to go in between them.</p> <p><i>Example</i></p> <pre><p>Hello
 You delivered your assignment ontime.
 Thanks
</pre> <p>Centering Content</p> <p>You can use <center> tag to put any content in the center of the page or any table cell.</p> <p><i>Example</i></p> <pre><center> <p>This text is in the center.</p> </center></pre> <p>Horizontal Lines</p> <p>Horizontal lines are used to visually break up sections of a document. The <hr> tag creates a line from the current position in the document to the right margin and breaks the line accordingly.</p> <p><i>Example</i></p> <pre><hr /></pre>

Nonbreaking Spaces

you should use a nonbreaking space entity ` ` instead of a normal space. For example, when coding the "12 Angry Men" in a paragraph, you should use something similar to the following code:

Example

```
<p>An example of this technique appears in the movie "12 Angry Men."</p>
```

HTML Lists

HTML offers authors several mechanisms for specifying lists of information. All lists must contain one or more list elements.

Unordered HTML List

- The first item
- The second item
- The third item
- The fourth item
- Ordered HTML List

Example

```
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

Ordered information.

1. The first item
2. The second item
3. The third item
4. The fourth item
5. HTML Description List

Example

```
<ol>
  <li>Coffee</li>
  <li>Milk</li>
</ol>
```

Definitions.

- The first item
- Description of item
- The second item
- Description of item

Example

```
<dl>
  <dt>Coffee</dt>
  <dd>- black hot drink</dd>
  <dt>Milk</dt>
  <dd>- white cold drink</dd>
</dl>
```

5. Discuss the various features available in HTML to format the text with example.

Basic functionality

- Server calls on TCP software and waits for connection req to one or more ports
- When a connection request is received , the server dedicates a subtask(Single copy of server software handling a single client connection)

- Subtask establish connection and receives request
- Subtask examines the host header field to determine the host and invokes software for this host
- Virtual host software Map Request-URI to specific resource on the server.
- It maps Request-URI to specific resource associated with the virtual host
- File: Return file in HTTP response (MIME Type)
- Program: Run program and return output in HTTP response
- Log information about the request and response such as IP address and the status code in a plain-text file.
- If the TCP connection is kept alive , the server subtask continues to monitor the connection, the client send another request or initiates a connection close.

Few Definitions

- All modern servers concurrently process multiple requests
- Multiple copies of the server running simultaneously(Concurrency)
- Subtask → Single copy of server software handling a single client connection
- Virtual Host → HTTP request include a host header field
- Multiple host names mapped by a DNS to a single IP address
- Web server determine which virtual host is being requested by examining the host header field.

Server Configuration and Tuning

- Modern servers have large number of Configuration parameters
- Server Configuration broken into two areas:
 - External Communication
 - Internal Processing
- In Tomcat two separate Java Packages:
 - Coyote
 - Catalina
- Coyote → Provides HTTP 1.1 communication
 - Catalina → Actual Servlet Container

Coyote parameters affecting External Communication
- IP addresses and TCP ports
- Number of subtasks created when server initialized
- Max number of threads allowed to exist simultaneously
- Max no of TCP connection request that will be queued if server is running its max no of threads. If queue full the received connection request is refused.
- “Keep-alive” time for inactive TCP connections
- Settings of the parameter affect the performance of the server.
- Tuning the Server
 - Changing the values of these and similar parameters in order to optimize performance
- Tuning is done by trial and error
- Load generation or stress test tools used to simulate request to a web server helpful for experimenting with tuning parameters

Service has Five Components

- Connector, Host, Logger, Realm, and Valve
- Connector is a coyote component handles HTTP communication
- Clicking on the connector will produce the window containing the dropdown menus of possible action that can be performed for this component

Defining Virtual Hosts

	<p>Configuring Host Elements</p> <ul style="list-style-type: none"> The Host element represents a virtual host, which is an association of a network name for a server (such as www.mycompany.com) with the particular server on which Tomcat is running. <p>Host Attributes</p> <ul style="list-style-type: none"> The attributes shown in following table may be viewed, set, or modified for a Host. Web server logs record information about server activity Access log is a file that records information about every HTTP request processed by the server Message logs → variety of debugging and other information generated by web server Access logging is performed by adding a valve component <p>Logging</p> <ul style="list-style-type: none"> Web server logs record information about server activity Access log is a file that records information about every HTTP request processed by the server Message logs → variety of debugging and other information generated by web server Access logging is performed by adding a valve component <p>Access Control</p> <ul style="list-style-type: none"> Provide automatic password protection for resources Access control: <ul style="list-style-type: none"> Password protection (e.g., admin pages) Users and roles defined in conf/tomcat-users.xml <ul style="list-style-type: none"> Deny access to machines Useful for denying access to certain users by denying access from the machines they use <p>List of denied machines maintained in RemoteHostValve (deny by host name) or RemoteAddressValve (deny by IP address)</p>
6.	<p>i) Explain how tables can be inserted into HTML document with example.</p> <ul style="list-style-type: none"> The HTML table model allows authors to arrange data -- text, preformatted text, images, links, forms, form fields, other tables, etc. -- into rows and columns of cells. Each table may have an associated caption that provides a short description of the table's purpose. A longer description may also be for the benefit of people using speech or Braille-based user agents. Table rows may be grouped into a head, foot, and body sections, Row groups convey additional structural information and may be rendered by user agents in ways that emphasize this structure. User agents may exploit the head/body/foot division to support scrolling of body sections independently of the head and foot sections. When long tables are printed, the head and foot information may be repeated on each page that contains table data. Authors may also group columns to provide additional structural information that may be exploited by user agents. Furthermore, authors may declare column properties at the start of a table definition in a way that enables user agents to render the table incrementally rather than having to wait for all the table data to arrive before rendering. Table cells may either contain "header" information or "data. Cells may span multiple rows and columns.

Here's a simple table that illustrates some of the features of the HTML table model. The following table definition:

```
<TABLE border="1"
      summary="This table gives some statistics about fruit
              flies: average height and weight, and percentage
              with red eyes (for both males and females).">
<CAPTION><EM>A test table with merged cells</EM></CAPTION>
<TR><TH rowspan="2"><TH colspan="2">Average
      <TH rowspan="2">Red<BR>eyes
<TR><TH>height<TH>weight
<TR><TH>Males<TD>1.9<TD>0.003<TD>40%
<TR><TH>Females<TD>1.7<TD>0.002<TD>43%
</TABLE>
```

A test table with merged cells

	Average		Red eyes
	height	weight	
Males	1.9	0.003	40%
Females	1.7	0.002	43%

ii) What is the significance of using forms on the web page? Enlist various components used on form.

A **webform, or HTML form** on a web page allows a user to enter data that is sent to a server for processing. Forms can resemble paper or database forms because web users fill out the forms using checkboxes, radio buttons, or text fields. For example, forms can be used to enter shipping or credit card data to order a product, or can be used to retrieve search results from a search engine.

Forms are enclosed in the HTML form tag. This tag specifies the communication endpoint the data entered into the form should be submitted to, and the method of submitting the data, GET or POST.

Elements

Forms can be made up of standard graphical user interface elements:

- text input — a simple text box that allows input of a single line of text (an alternative, password, is used for security purposes, in which the character typed in are invisible or replaced by symbols such as *)
- radio — a radio button
- file — a file select control for uploading a file
- reset — a reset button that, when activated, tells the browser to restore the values to their initial values.
- submit — a button that tells the browser to take action on the form (typically to send it to a server)
- textarea — much like the text input field except a textarea allows for multiple rows of data to be shown and entered

- select — a drop-down list that displays a list of items a user can select from

uses of various elements:

- a text box asking for your name
- a pair of radio buttons asking you to pick your sex
- a select box giving you a list of eye colors to choose from
- a pair of check boxes to click on if they apply to you
- a text area to describe your athletic ability
- a submit button to send it to the server

These basic elements provide most common graphical user interface (GUI) elements, but not all. For example, there are no equivalents to a combo box, tree view, or grid view.

A grid view, however, can be mimicked by using a standard HTML table with each cell containing a text input element.

A tree view could also be mimicked through nested tables or, more semantically appropriately, nested lists.

In both cases, a server side process is responsible for processing the information, while JavaScript handles the user-interaction.

Implementations of these interface elements are available through JavaScript libraries such as jQuery.

HTML 4 introduced the label tag, which is intended to represent a caption in a user interface, and can be associated with a specific form control by specifying the id attribute of the control in the label tag's for attribute.

HTML 5 introduces a number of input tags that can be represented by other interface elements. Some are based upon text input fields and are intended to input and validate specific common data.

These include email to enter email addresses, tel for telephone numbers, number for numeric values.

There are additional attributes to specify required fields, fields that should have keyboard focus when the web page containing the form is loaded, and placeholder text that is displayed within the field but is not user input.

The date input type displays a calendar from which the user can select a date or date range.

And the color input type can be represented as an input text simply checking the value entered is a correct hexadecimal representation of a color, according to the specification, or a color picker widget.

7. Discuss how to create list and frame using HTML. Give Example.

HTML **frames** are used to divide your browser window into multiple sections where each section can load a separate HTML document. A collection of frames in the browser window is known as a frameset. The window is divided into frames in a similar way the tables are organized: into rows and columns.

Creating Frames

To use frames on a page we use <frameset> tag instead of <body> tag. The <frameset> tag defines how to divide the window into frames. The rows attribute of <frameset> tag defines horizontal frames and cols attribute defines vertical frames. Each frame is indicated by <frame> tag and it defines which HTML document shall open into the frame.

Example

Following is the example to create three horizontal frames:

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Frames</title>
</head>
<frameset rows="10%,80%,10%">
  <frame name="top" src="/html/top_frame.htm" />
  <frame name="main" src="/html/main_frame.htm" />
  <frame name="bottom" src="/html/bottom_frame.htm" />
</frameset>
</html>
```

This will produce following result:



The <frame> Tag Attributes

Following are important attributes of <frame> tag:

Attribute	Description
src	This attribute is used to give the file name that should be loaded in the frame. Its value can be any URL. For example, src="/html/top_frame.htm" will load an HTML file available in html directory.
name	This attribute allows you to give a name to a frame. It is used to indicate which frame a document should be loaded into. This is especially important when you want to create links in one frame that load pages into an another frame, in which case the second frame

	needs a name to identify itself as the target of the link.
frameborder	This attribute specifies whether or not the borders of that frame are shown; it overrides the value given in the frameborder attribute on the <frameset> tag if one is given, and this can take values either 1 (yes) or 0 (no).
marginwidth	This attribute allows you to specify the width of the space between the left and right of the frame's borders and the frame's content. The value is given in pixels. For example marginwidth="10".
marginheight	This attribute allows you to specify the height of the space between the top and bottom of the frame's borders and its contents. The value is given in pixels. For example marginheight="10".
noresize	By default you can resize any frame by clicking and dragging on the borders of a frame. The noresize attribute prevents a user from being able to resize the frame. For example noresize="noresize".
scrolling	This attribute controls the appearance of the scrollbars that appear on the frame. This takes values either "yes", "no" or "auto". For example scrolling="no" means it should not have scroll bars.
longdesc	This attribute allows you to provide a link to another page containing a long description of the contents of the frame. For example longdesc="framedescription.htm"

List :

HTML offers web authors three ways for specifying lists of information. All lists must contain one or more list elements. Lists may contain:

- - An unordered list. This will list items using plain bullets.
- - An ordered list. This will use different schemes of numbers to list your items.
- <dl> - A definition list. This arranges your items in the same way as they are arranged in a dictionary.

HTML Unordered Lists

An unordered list is a collection of related items that have no special order or sequence. This list is created by using HTML tag. Each item in the list is marked with a bullet.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Unordered List</title>
</head>
<body>
<ul>
<li>Beetroot</li>
<li>Ginger</li>
<li>Potato</li>
<li>Radish</li>
</ul>
</body>
</html>
```

This will produce following result:

- Beetroot
- Ginger
- Potato
- Radish

The type Attribute

You can use type attribute for tag to specify the type of bullet you like. By default it is a disc. Following are the possible options:

```
<ul type="square">
<ul type="disc">
<ul type="circle">
```

Example

Following is an example where we used <ul type="square">

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Unordered List</title>
</head>
<body>
<ul type="square">
<li>Beetroot</li>
<li>Ginger</li>
<li>Potato</li>
<li>Radish</li>
</ul>
</body>
</html>
```

This will produce following result:

- Beetroot
- Ginger
- Potato
- Radish

The start Attribute

You can use start attribute for tag to specify the starting point of numbering you need. Following are the possible options:

```
<ol type="1" start="4"> - Numerals starts with 4.
<ol type="I" start="4"> - Numerals starts with IV.
<ol type="i" start="4"> - Numerals starts with iv.
<ol type="a" start="4"> - Letters starts with d.
<ol type="A" start="4"> - Letters starts with D.
```

Example

Following is an example where we used <ol type="i" start="4" >

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Ordered List</title>
</head>
<body>
  <ol type="i" start="4">
    <li>Beetroot</li>
    <li>Ginger</li>
    <li>Potato</li>
    <li>Radish</li>
  </ol>
</body>
</html>
```

This will produce following result:

- iv. Beetroot
- v. Ginger
- vi. Potato
- vii. Radish

8. Explain the capabilities of Web Server (APR/MAY 2013)

Basic functionality

- Server calls on TCP software and waits for connection req to one or more ports
- When a connection request is received , the server dedicates a subtask(Single copy of server

software handling a single client connection)

- Subtask establish connection and receives request
- Subtask examines the host header field to determine the host and invokes software for this host
- Virtual host software Map Request-URI to specific resource on the server.
- It maps Request-URI to specific resource associated with the virtual host
 - File: Return file in HTTP response (MIME Type)
 - Program: Run program and return output in HTTP response
- Log information about the request and response such as IP address and the status code in a plain-text file.
- If the TCP connection is kept alive , the server subtask continues to monitor the connection, the client send another request or initiates a connection close.

Few Definitions

- All modern servers concurrently process multiple requests
- Multiple copies of the server running simultaneously(Concurrency)
- Subtask→ Single copy of server software handling a single client connection
- Virtual Host→ HTTP request include a host header field
- Multiple host names mapped by a DNS to a single IP address
- Web server determine which virtual host is being requested by examining the host header field.

Server Configuration and Tuning

- Modern servers have large number of Configuration parameters
- Server Configuration broken into two areas:
 - External Communication
 - Internal Processing
- In Tomcat two separate Java Packages:
 - Coyote
 - Catalina
- Coyote→ Provides HTTP 1.1 communication
 - Catalina→ Actual Servlet Container

Coyote parameters affecting External Communication

- IP addresses and TCP ports
- Number of subtasks created when server initialized
- Max number of threads allowed to exist simultaneously
- Max no of TCP connection request that will be queued if server is running its max no of threads. If queue full the received connection request is refused.
 - “Keep-alive” time for inactive TCP connections
- Settings of the parameter affect the performance of the server.
- Tuning the Server
 - Changing the values of these and similar parameters in order to optimize performance
- Tuning is done by trial and error
- Load generation or stress test tools used to simulate request to a web server helpful for experimenting with tuning parameters

Service has Five Components

- Connector, Host, Logger, Realm, and Valve
- Connector is a coyote component handles HTTP communication
- Clicking on the connector will produce the window containing the dropdown menus of possible

action that can be performed for this component

Defining Virtual Hosts

Configuring Host Elements

- The Host element represents a virtual host, which is an association of a network name for a server (such as www.mycompany.com) with the particular server on which Tomcat is running.

Host Attributes

- The attributes shown in following table may be viewed, set, or modified for a Host.
- Web server logs record information about server activity
- Access log is a file that records information about every HTTP request processed by the server
- Message logs → variety of debugging and other information generated by web server
- Access logging is performed by adding a valve component

Logging

- Web server logs record information about server activity
- Access log is a file that records information about every HTTP request processed by the server
- Message logs → variety of debugging and other information generated by web server
- Access logging is performed by adding a valve component

Access Control

- Provide automatic password protection for resources
- Access control:
 - Password protection (e.g., admin pages)
 - Users and roles defined in conf/tomcat-users.xml
 - Deny access to machines
 - Useful for denying access to certain users by denying access from the machines they use

List of denied machines maintained in RemoteHostValve (deny by host name) or RemoteAddressValve (deny by IP address)

9. Explain about the XHTML DTD with an Example.

DTD's

XHTML documents have three parts: the DOCTYPE (which contains the DTD declaration), the head and the body. To create web pages that properly conform to the XHTML 1.0 standard, each page must include a DTD declaration; either **strict**, **transitional**, or **frameset**.

1. Strict

You should use the strict DTD when your XHTML pages will be marked up cleanly, free of presentational clutter. You use the strict DTD together with cascading style sheets, because it doesn't allow attributes like "bgcolor" to be set for the <body> tag, etc.

The strict DTD looks like this:

```
<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

Example

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
```

```

<title> Strict DTD XHTML Example </title>
</head><body><p>
Please Choose a Day:
<br /><br />
<select name="day">
<option selected="selected">Monday</option>
<option>Tuesday</option>
<option>Wednesday</option>
</select>
</p></body>
</html>

```

2. Transitional

The transitional DTD should be used when you need to take advantage of the presentational features that are available through HTML. You should also use the transitional DTD when you want to support older browsers that don't have built-in support for cascading style sheets.

The transitional DTD looks like this:

```

<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

```

Example

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<title> Transitional DTD XHTML Example </title>
</head>
<body bgcolor="#FFFFFF" link="#000000" text="red">
<p>This is a transitional XHTML example</p>
</body>
</html>

```

3. Frameset

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">

```

contain frames. The frameset DTD looks like this:

Example

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
"DTD/xhtml1-frameset.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title> Frameset DTD XHTML Example </title> </head>
<frameset cols="100,*">
<frame src="toc.html" />

```

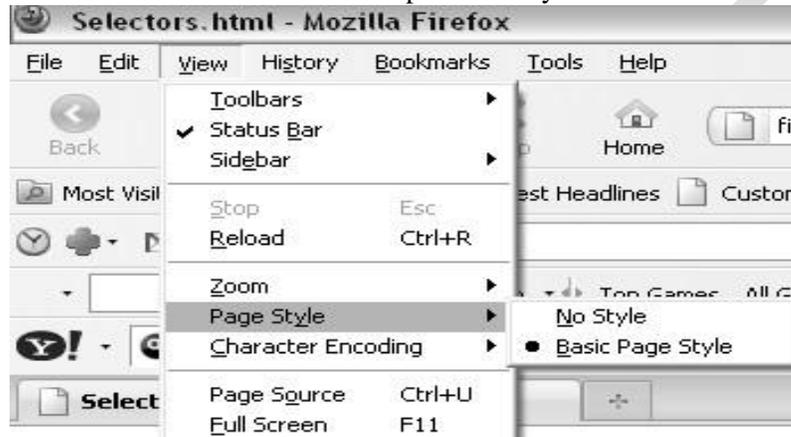
You should use the frameset DTD when your XHTML page will

	<pre><frame src="intro.html" name="content" /> </frameset> </html></pre>
10.	<p>Explain the significance of XHTML with the help of a real time application. Write necessary code snippets (MAY/JUNE 2014)</p> <p>Extensible Hypertext Markup Language (XHTML) is a family of XML markup languages that mirror or extend versions of the widely used Hypertext Markup Language (HTML), the language in which Web pages are formulated.</p> <p>Significance</p> <ul style="list-style-type: none"> • Sustainability. Web applications tend towards XML. Using XHTML now instead of HTML makes any future conversion of the website easier. • Extensibility. Owing to the extensibility of XML, XHTML documents can be supplemented with other forms of markup, MathML (Math Markup Language) SVG (Scalable Vector Graphics) or your own markup variants, thanks to the use of namespaces. • Compatibility. Because XHTML documents are written in compliance with the rules of XML, XML-processing programmes can effortlessly convert an XHTML file to another format (e.g. PDF, RSS or RTF). • Efficiency of processing applications. Once browsers support XHTML documents and the strict rules of XML, they will become quicker thanks to shorter error processing routines. At present, a great deal of the processing power of a browser is still spent on liberal error processing of documents containing malformed HTML markup. <p>Features</p> <ul style="list-style-type: none"> • XHTML requires strict adherence to coding rules. • XHTML encourages a more structured and conceptual way of thinking about content and, combined with the style sheet, a more creative way of displaying it. • XHTML makes it easier for people to dream up and add new elements. <p>Code snippets</p> <pre><!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"> <html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en"> <head> <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/> <title>Page Title</title> <link rel="stylesheet" href="style.css" type="text/css" media="screen" charset="utf-8"/> </head> <body> </body> </html></pre>
11.	<p>Explain about Style Rule Cascading and Inheritance</p> <p>Rule cascading:</p> <pre>*{ font-weight : bold } applies to every element of HTML #p1, #p3 { background-color : aqua } #p3{ font-weight : bold }</pre> <p>What if more than one style declaration applies to a property of an element?</p> <ul style="list-style-type: none"> • Multiple declaration • Browser applies rule cascading • A multistage sorting process that selects a single declaration that supply the property value • The CSS rule cascade determines which style rule's declaration applies

- Once declaration identified, associate **origin and weight** with every declaration
 - PERSON WHO WROTE THE DOCUMENT
 - PERSON WHO IS VIEWING THE DOCUMENT
 - PERSON WHO WROTE THE BROWSER SOFTWARE

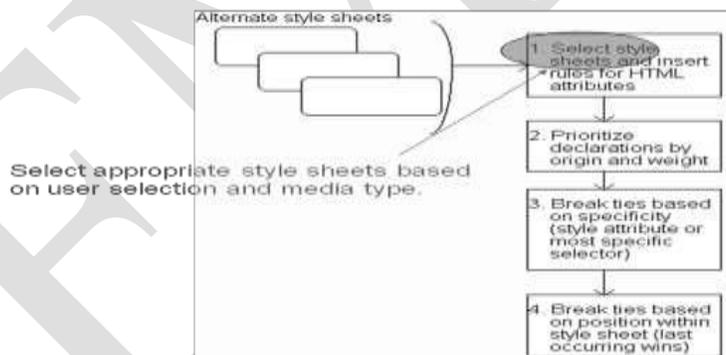
Origin of a declaration is one of the following:

- Author-> declaration is part of an external or embed style sheet or part of the value specified by style attribute
- User agent ->A browser define default style property values for HTML elements
- User-> Modern browsers allow users to provide a style sheet



- User style rules defined in two ways:**
 - Edit|Preferences|Appearance**, Fonts and colors panels allow a user to select various style options
 - User can explicitly create a style sheet file the browser will input when it is started
 - Features provided by IE6
 - Tools|Internet Options**
- Two weight values**
 - Normal
 - Important
- User/important highest priority in CSS to accommodate users with special needs

CSS Rule Cascade



- Rules made important by adding “!important”:

```
p { text-indent:3em; font-size:larger !important }
```

Cascading order

What style will be used when there is more than one style specified for an HTML element?

all the styles will "cascade" into a new "virtual" style sheet by the following rules, where number four has the highest priority:

1. **Browser default**
2. **External style sheet**
3. **Internal style sheet (in the head section)**
4. **Inline style (inside an HTML element)**

So, an inline style (inside an HTML element) has the highest priority, which means that it will override a style defined inside the <head> tag, or in an external style sheet, or in a browser (a default value).

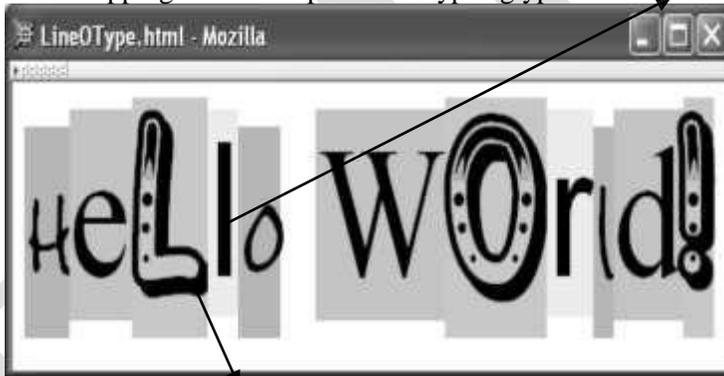
Note: If the link to the external style sheet is placed after the internal style sheet in HTML <head>, the external style sheet will override the internal style sheet!

Style Inheritance

- Cascading based on structure of style sheets
- Inheritance based on tree structure of document
- What if no style declaration applies to a property of an element?
- Generally, the property value is inherited from the nearest ancestor element that has a value for the property
- If no ancestor has a value (or the property does not inherit) then CSS defines an initial value that is used
- Property values:
 - Specified: value contained in declaration
 - Absolute: value can be determined without reference to context (*e.g.*, 2cm)
 - Relative: value depends on context (*e.g.*, larger)
 - Computed: browser performs calculation depends on particular relative value
 - absolute representation of relative value (*e.g.*, larger might be 1.2 x parent font size)
 - Actual: value actually used by browser (*e.g.*, computed value might be rounded)
- Most properties inherit computed value
 - Exception discussed later: line-height
- A little thought can usually tell you whether a property inherits or not
 - Example: height does not inherit

12. Explain any eight CSS text properties.

A font is a mapping from code points to Glyphs glyph

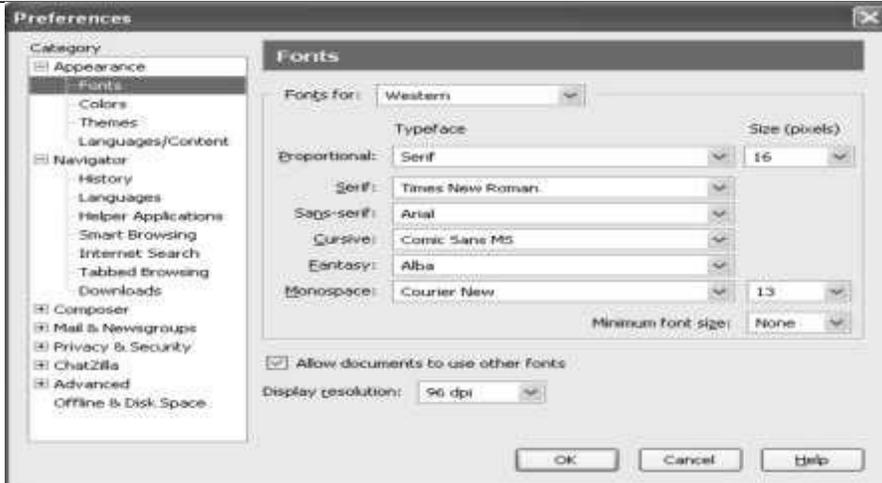


Character cell(content area)

A font family is a collection of related fonts(typically differ in size, weight, etc.)

```
<p style="font-family:'Jenkins v2.0'">
```

- font-family property can accept a list offamilies, including generic font families
- ```
font-family:"Edwardian Script ITC","French Script MT",cursive
```



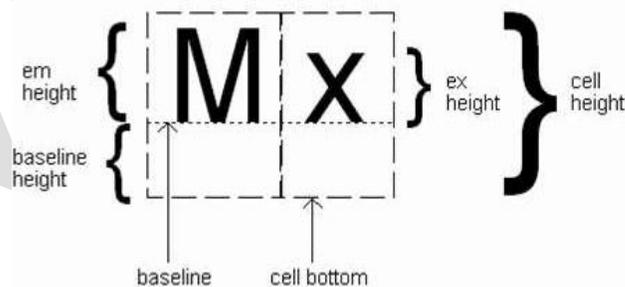
### **LENGTH SPECIFICATION IN CSS:**

Many properties, such as font-size, have a value that is a CSS length

- All CSS length values except 0 need units

TABLE 3.4: CSS length unit identifiers.

| Identifier | Meaning                                                                                    |
|------------|--------------------------------------------------------------------------------------------|
| in         | inches                                                                                     |
| cm         | centimeters                                                                                |
| mm         | millimeters                                                                                |
| pt         | points: 1/72-inch                                                                          |
| pc         | picas: 12 points                                                                           |
| px         | pixel: typically 1/96-inch (see text).                                                     |
| em         | 1em is roughly the height of a capital letter in the reference font (see text).            |
| ex         | 1ex is roughly the height of the lowercase 'x' character in the reference font (see text). |



### **Reference font defines em and ex units**

- Normally, reference font is the font of the element being styled
- Exception: Using em/ex to specify value for font-size

### **FONT PROPERTIES:**

Other ways to specify value for font-size:

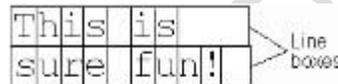
- Percentage (of parent font-size)
- Absolute size keyword: xx-small, x-small, small, medium (initial value), large, x-large, xx-large
- User agent specific; should differ by ~ 20%
- Relative size keyword: smaller, larger

Additional font style properties.

| Property     | Possible values                                                                                                                                                                       |
|--------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| font-style   | normal (initial value), italic (more cursive than normal), or oblique (more slanted than normal).                                                                                     |
| font-weight  | bold or normal (initial value) are standard values, although other values can be used with font families having multiple gradations of boldness (see CSS2 [W3C-CSS-2.0] for details). |
| font-variant | small-caps, which displays lowercase characters using uppercase glyphs (small uppercase glyphs if possible), or normal (initial value)                                                |

**LINE BOXES:**

Text is rendered using line boxes



- Height of line box given by line-height
  - Initial value: normal (*i.e.*, cell height;relationship with em height is font-specific)
  - Other values (following are equivalent):

```
line-height:1.5em
line-height:150%
line-height:1.5
```

**font shortcut property:**

```
{ font: italic bold 12pt "Helvetica",sans-serif } equals to
{ font-style: italic;
 font-variant: normal;
 font-weight: bold;
 font-size: 12pt;
 line-height: normal;
 font-family: "Helvetica",sans-serif }
```

**TEXT FORMATTING AND TEXT COLOR:**

Primary CSS text properties.

| Property        | Values                                                                                                                                                            |
|-----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| text-decoration | none (initial value), underline, overline, line-through, or space-separated list of values other than none.                                                       |
| letter-spacing  | normal (initial value) or a length representing additional space to be included between adjacent letters in words. Negative value indicates space to be removed.  |
| word-spacing    | normal (initial value) or a length representing additional space to be included between adjacent words. Negative value indicates space to be removed.             |
| text-transform  | none (initial value), capitalize (capitalizes first letter of each word), uppercase (converts all text to uppercase), lowercase (converts all text to lowercase). |
| text-indent     | length (initial value 0) or percentage of box width, possibly negative. Specify for block elements and table cells to indent text within first line box.          |
| text-align      | left (initial value for left-to-right contexts), right, center, or justified. Specify for block elements and table cells.                                         |
| white-space     | normal (initial value), pre. Use to indicate whether or not white space should be retained.                                                                       |

**Font color specified by color property****• Two primary ways of specifying colors:**

– Color name: black, gray, silver, white, red, lime, blue, yellow, aqua, fuchsia, maroon, green, navy, olive, teal, purple, full list at

<http://www.w3.org/TR/SVG11/types.html#ColorKeywords>

– red/green/blue (RGB) values

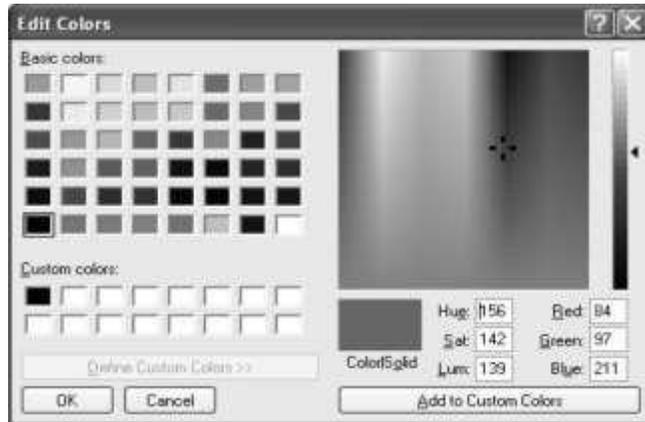


TABLE 3.7: Alternative formats for specifying numeric color values.

| Format                           | Example                         | Meaning                                                                                                                           |
|----------------------------------|---------------------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| Functional, integer arguments    | <code>rgb(255,170,0)</code>     | Use arguments as RGB values.                                                                                                      |
| Functional, percentage arguments | <code>rgb(100%,66.7%,0%)</code> | Multiply arguments by 255 and round to obtain RGB values (at most one decimal place allowed in arguments).                        |
| Hexadecimal                      | <code>#ffaa00</code>            | The first pair of hexadecimal digits represents the red intensity, second and third represent green and blue, respectively.       |
| Abbreviated hexadecimal          | <code>#fa0</code>               | Duplicate the first hexadecimal digit to obtain red intensity, duplicate second and third to obtain green and blue, respectively. |

**13. Explain about the various style sheets with examples. (Internal,External,Inline) (APR/MAY 2013)**

1. To create an inline style
  - a. Add the style attribute to the HTML tag.
  - b. The style declaration must be enclosed within double quotation marks.
2. To create an embedded style
  - a. Insert a `<style>` tag within the head section of HTML file.
  - b. Within the `<style>` tag, enclose the style declarations need to the entire Web page.
  - c. The style sheet language identifies the type of style used in the document.
  - d. The default and the most common language is “text/css” for use with CSS.
3. To create an External styles
  - a. Create a text file containing style declarations
  - b. Create a link to that file in each page of the Web site using a `<link>` tag.
  - c. Specify the link attributes, such as href, rel, and type.
  - d. Link a style sheet, the value of the href attribute should be the “URL” of the linked document, the value of the rel attribute should be “stylesheet” and the value of the type attribute should be

“text/css”.

**EXTERNAL.CSS:**

```
body{ background-color: gray;}
p { color: blue; }
h3{ color: white; }
```

**EXTERNAL.HTML:**

```
<html>
<head>
<link rel="stylesheet" type="text/css" href="EXTERNAL.css" /><!--Link tag for External CSS-->
</head>
<body>
<h3> A White Header </h3>
<p> This paragraph has a blue font.
The background color of this page is gray because
we changed it with CSS! </p>
</body>
</html>
```

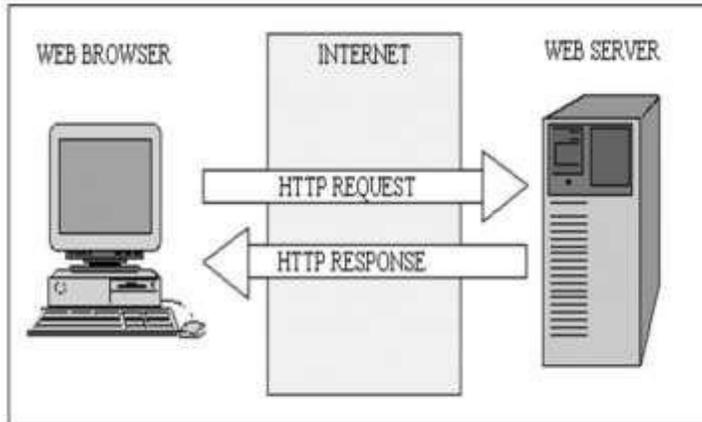
**INTERNAL.HTML:**

```
<html>
<head>
<style> <!--Style tag for Internal CSS-->
body { background-color: blue; }
p { color: white; }
</style>
</head>
<body>
<h2>Internal CSS</h2>
<p>This page uses internal CSS. Using the style tag we are able to modify
the appearance of HTML elements.</p>
</body>
</html>
```

**INLINE.HTML:**

```
<html>
<head>
</head>
<body>
<h2>InLINE CSS</h2>
<p style="color:sienna;margin-left:20px"><!--Style Attribute(INLINE)-->
This page uses INLINE CSS. Using the style ATTRIBUTE we are able to modify
the appearance of HTML elements.
</p>
</body>
</html>
```

14 **Difference between web browser and web server**



Web server and web browser are the terms which are commonly used for website. The basic purpose of both is to develop a platform for internet web directory. So that any users can anytime access any kind of website. Major difference between them is on their function and how they perform their functions. Check for the detail of both topics before understanding the differences between them.

### **Web Server:**

Web server is a computer system, which provides the web pages via HTTP (Hypertext Transfer Protocol). IP address and a domain name is essential for every web server. Whenever, you insert a URL or web address into your web browser, this sends request to the web address where domain name of your URL is already saved. Then this server collects the all information of your web page and sends to browser, which you see in form of web page on your browser. Making a web server is not a difficult job. You can convert your computer into a web server with the help of any server software and connecting the computer to the internet. Lot of web server software are available in the market in shape of NCSA, Apache, Microsoft and Netscape. Storing, processing and delivering web pages to clients is its main function. All the communication between client (web browser) and server takes place via HTTP.

### **Web Browser:**

Web browser is a client, program, software or tool through which we sent HTTP request to web server. The main purpose of web browser is to locate the content on the World Wide Web and display in the shape of web page, image, audio or video form. You can call it a client server because it contacts the web server for desired information. If the requested data is available in the web server data then it will send back the requested information again via web browser. Microsoft Internet Explorer, Mozilla Firefox, Safari, Opera and Google Chrome are examples of web browser and they are more advanced than earlier web browser because they are capable to understand the HTML, JavaScript, AJAX, etc. Now a days, web browser for mobiles are also available, which are called microbrowser.

### **Difference:**

Following are the differences between web server and web browser.

- Web server is essential to store all information and data of websites. While web browser are used

to access and locate these information and data.

- Web browser is used to search something on the internet via websites. While web server is used to make the links between websites and web browser.
- Web browser is a software or application which is used for collection and presentation of data in shape of websites while web server is a program server on computer or in cloud on internet that gives the data.

## 15 Difference between internet and intranet.

### Internet

It is a worldwide system which has the following characteristics:

- Internet is a world-wide / global system of interconnected computer networks.
- Internet uses the standard Internet Protocol (TCP/IP)
- Every computer in internet is identified by a unique IP address.
- IP Address is a unique set of numbers (such as 110.22.33.114) which identifies a computer's location.
- A special computer DNS (Domain Name Server) is used to give name to the IP Address so that user can locate a computer by a name.
- For example, a DNS server will resolve a name **http://www.tutorialspoint.com** to a particular IP address to uniquely identify the computer on which this website is hosted.
- Internet is accessible to every user all over the world.



### Intranet

- Intranet is system in which multiple PCs are connected to each other.

- PCs in intranet are not available to the world outside the intranet.
- Usually each company or organization has their own Intranet network and members/employees of that company can access the computers in their intranet.
- Each computer in Intranet is also identified by an IP Address which is unique among the computers in that Intranet.



#### Similarities in Internet and Intranet

- Intranet uses the internet protocols such as TCP/IP and FTP.
- Intranet sites are accessible via web browser in similar way as websites in internet. But only members of Intranet network can access intranet hosted sites.
- In Intranet, own instant messengers can be used as similar to yahoo messenger/ gtalk over the internet.

#### Differences in Internet and Intranet

- Internet is general to PCs all over the world whereas Intranet is specific to few PCs.
- Internet has wider access and provides a better access to websites to large population whereas Intranet is restricted.
- Internet is not as safe as Intranet as Intranet can be safely privatized as per the need.

### 16 Building Advanced Web 2.0 Applications.

#### Definition of Mash up applications

• A *mashup* is similar to a remix. You might have heard examples again from the music world where elements of Led Zeppelin are combined with Jayzee, for example, to form a weird rap/rock song. That's a mashup. The same can be done with data from the Internet.

#### Mashup Techniques - Mashing on the Web Server

• Every site sits on a Web server. It's the thing that serves up the page, typically Internet Information Server (IIS) in the Microsoft world.

## **Understanding the Architecture How it works**

This use case is definitely the most straightforward:

- The Web browser communicates with the server, requesting a page using straight HTTP or HTTPS.
- That page is constructed by the Web server, which reaches out to what I'll call the *source* or *partner sites* (for example, Amazon, Yahoo, or Google, and so on). The first request in this example is to Amazon using the Simple Object Access Protocol (SOAP) over HTTP.
- Amazon returns back a SOAP response.
- The second request in this example is to Yahoo using a Representational State Transfer style approach.
- Yahoo responds with Plain Old XML over HTTP.
- Lastly, the Web server aggregates the responses, combining and rationalizing the data in whatever manner makes sense.
- The resulting data is bound to the HTML and inserted into the response, which is sent back to the browser.

## **Pros and Cons**

- The benefits of this approach are that the browser is decoupled entirely from the partner sites supplying the data. The Web server acts as a proxy and aggregator for the responses.
- Disadvantages of this approach are that the browser requests an entire page, which typically is acceptable.
- Second, the Web server is doing all the work in terms of data manipulation. Though this is good in terms of maintenance, it's not so good in terms of scalability. When your mashup gains popularity and starts being viewed by thousands of users, the amount of work the server's doing increases, while the browser residing at the client is relatively idle.

## **Remote Data Communication**

- Remote data communication occurs at runtime.
- Flex applications support a variety of remote data communication techniques built on standards.
- There are three basic categories of Flex application remote data communication:

### *HTTP request/response-style communication*

- This category consists of several overlapping techniques. Utilizing the Flex framework HTTPService component or the Flash Player API URLLoader class, you can send and load uncompressed data such as text blocks, URL encoded data, and XML packets. Each technique achieves the similar goal of sending requests and receiving responses using HTTP or HTTPS.

### *Real-time communication*

- This category consists of persistent socket connections. Flash Player supports two types of socket connections: those that require a specific format for packets (XMLSocket) and those that allow raw socket connections (Socket)

### *File upload/download communication*

- This category consists of the FileReference API which is native to Flash Player and allows file upload and download directly within Flex applications.

## Understanding Strategies for Data Communication

- When you build Flex applications that utilize data communication, it's important to understand the strategies available for managing those communications and how to select the right strategy for an application. All Flex applications run in Flash Player. With the exception of some Flex applications created using Flex Data Services, almost all Flex applications are composed of precompiled *.swf* files that are loaded in Flash Player on the client.
- Because Flex applications are stateful and self-contained, they don't require new page requests and wholesale screen refreshes to make data requests and handle responses.
- The Flex framework provides components for working with data communication using standard HTTP requests as well as SOAP requests.

## Working with Request/Response Data Communication

- You can work with request/response data communication in three basic ways: via simple HTTP services, web services, and Flash Remoting.

### Simple HTTP Services

- The most basic type of HTTP request/response communication uses what we call *simple HTTP services*. These services include things such as text and XML resources, either in static documents or dynamically generated by something such as a ColdFusion page, a servlet, or an ASP.NET page.

### HTTPService

- HTTPService is a component that allows you to make requests to simple HTTP services such as text files, XML files, or scripts and pages that return dynamic data. You must always define a value for the `url` property of an HTTPService object.
- The following example uses MXML to create an HTTPService object that loads text from a file called *data.txt* saved in the same directory as the compiled *.swf* file:

```
<mx:HTTPService id="textService" url="data.txt" />
```

### Sending requests

- Creating an HTTPService object does not automatically make the request to load the data. In order to make the request, you must call the `send()` method. If you want to load the data when the user clicks a button, you can call the `send()` method in response to a click event:

```
textService.send();
```

### Handling results

- The `send()` method makes the request, but a response is not likely to be returned instantaneously. Instead, the application must wait for a result event. The following example displays an alert when the data loads:

```
<mx:HTTPService id="textService" url="data.txt" result="mx.controls.Alert.show('Data loaded')" />
```

### Sending parameters

- When you want to pass parameters to the service, you can use the `request` property of the HTTPService instance. The `request` property requires an Object value. By default, the name/value pairs of the object are converted to URL-encoded format and are sent to the service using HTTP GET.
- The default value is `object`, which yields the default behavior you've already seen. You can optionally

specify any of the following values:

**Text:**The data is not parsed at all, but is treated as raw text.

**Flashvars:**The data is assumed to be in URL-encoded format, and it will be parsed into an object with properties corresponding to the name/value pairs.

**Array:**The data is assumed to be in XML format, and it is parsed into objects much the same as with the object settings. However, in this case, the result is always an array. If the returned data does not automatically parse into an array, the parsed data is placed into an array.

**Xml:**The data is assumed to be in XML format, and it is interpreted as XML using the legacy XMLNode ActionScript class.

**e4x:**The data is assumed to be in XML format, and it is interpreted as XML using the ActionScript 3.0 XML class (E4X).

### Using HTTPService with ActionScript

- Although the simplest and quickest way to use an HTTPService object is to primarily use MXML, this technique is best-suited to nonenterprise applications in which the data communication scenarios are quite simple.
- Because HTTPService components provide significant data conversion advantages (such as automatic serialization of data), it is still frequently a good idea to use an HTTPService object within a remote proxy. However, it is generally necessary to then work with the HTTPService component entirely with ActionScript, including constructing the object and handling the responses. **URLLoader**
- HTTPService allows you to use requests and handle responses to and from simple HTTP services. You can optionally use the Flash Player class called flash.net.URLLoader to accomplish the same tasks entirely with ActionScript, but at a slightly lower level.
- The first step when working with a URLLoader object is always to construct the object using the constructor method, as follows:

```
var loader:URLLoader = new URLLoader();
```

- Once you've constructed the object, you can do the following:
  - Send requests.
  - Handle responses.
  - Send parameters.

#### **Sending requests**

- You can send requests using the load( ) method of a URLLoader object. The load( ) method requires that you pass it a flash.net.URLRequest object specifying at a minimum what URL to use when making the request. The following makes a request to a text file called *data.txt*:

```
loader.load(new URLRequest("data.txt"));
```

#### **Handling responses**

- URLLoader objects dispatch complete events when a response has been returned. Any return value is stored in the data property of the URLLoader object.

#### **Sending parameters**

- You can send parameters using URLLoader as well. In order to send parameters, you assign a value to the data property of the URLRequest object used to make the request. The URLRequest object can send binary data or string data.

#### **Web Services**

- Flash Player has no built-in support for SOAP web services. However, Flex provides a WebService component that uses built-in HTTP request/response support as well as XML support to enable you to

work with SOAP-based web services. There are two ways you can work with the WebService components: using MXML and using ActionScript. Using WebService Components with MXML

- You can create a WebService component instance using MXML. When you do, you should specify an id and a value for the wsdl property.\

Eg: `<mx:WebService id="statesService" wsdl="http://www.rightactionscript.com/states/webservice/StatesService.php?wsdl" />`

Web services define one or more methods or operations. You must define the WebService instance so that it knows about the operations using nested operation tags. The operation tag requires that you specify the name at a minimum.

#### Calling web service methods

- All operations that you define for a WebService component instance are accessible as properties of the instance. For example, in the preceding section, we created a WebService instance called statesService with an operation called getCountries. That means you can use ActionScript to reference the operation as statesService.getCountries.
- You can then call getCountries just as though it were a method of statesService:  
statesService.getCountries( );

#### Handling results

- When a web service operation returns a result, you can handle it in one of two ways: explicitly handle the result event or use data binding. Then, once a result is returned, you can retrieve the result value from the lastResult property of the operation.

#### Using WebService Components with ActionScript

- You can use a WebService component using ActionScript instead of MXML. This is useful in cases where you want to fully separate the view from the controller and the model, such as in the recommended remote proxy approach.
- The MXML version of the WebService component is an instance of mx.rpc.soap.mxml.WebService, which is a subclass of mx.rpc.soap.WebService. When you use the component directly from ActionScript you should instantiate mx.rpc.soap.WebService directly:  
// Assume the code already has an import statement for mx.rpc.soap.WebService.  
var exampleService:WebService = new WebService( );
- Next, you must call a method called loadWSDL( ). You must call the method prior to calling any of the web service operations. Assuming you set the wsdl property, you don't need to pass any parameters to loadWSDL( ):  
exampleService.loadWSDL( );

### Part – A

- 1. What is JavaScript?**  
JavaScript is a platform-independent, event-driven, interpreted client-side scripting language developed by Netscape Communications Corp. and Sun Microsystems.
- 2. What are the primitive data types in javascript?**  
JavaScript supports five primitive data types: number, string, Boolean, undefined, and null. These types are referred to as primitive types because they are the basic building blocks from which more complex types can be built. Of the five, only number, string, and Boolean are real data types in the sense of actually storing data. Undefined and null are types that arise under special circumstances.
- 3. What are the Escape Codes Supported in JavaScript?**

	The Escape codes supported in javascript are \b Backspace,\t Tab (horizontal), \n Linefeed (newline),\v Tab (vertical),\f Form feed,\r Carriage return,\" Double quote \' Single quote,\\ Backslash.			
4.	<b>What is JavaScript name spacing? How and where is it used?</b> Using global variables in JavaScript is evil and a bad practice. That being said, namespacing is used to bundle up all your functionality using a unique name. In JavaScript, a namespace is really just an object that you've attached all further methods, properties and objects. It promotes modularity and code reuse in the application.			
5.	<b>How many looping structures can you find in javascript?</b> If you are a programmer, you know the use of loops. It is used to run a piece of code multiple times according to some particular condition. Javascript being a popular scripting language supports the following loops for, while, do-while loop			
6.	<b>Mention the various Java Script Object Models.</b> Math Object, String Object, Date Object, Boolean and Number Object, Document Object Window Object.			
7.	<b>How Scripting Language Is Differs from HTML?</b> HTML is used for simple web page design, HTML with FORM is used for both form design and Reading input values from user, Scripting Language is used for Validating the given input values weather it is correct or not, if the input value is incorrect, the user can pass an error message to the user, Using form concept various controls like Text box, Radio Button, Command Button, Text Area control and List box can be created.			
8.	<b>What are the different types of objects in JavaScript?</b>			
	<b>Type</b>	<b>Example</b>	<b>Implementation Provided By</b>	<b>Governing Standard</b>
	User-defined	Programmer defined Customer or Circle	Programmer	None
	Built-in	Array, Math	The browser via engine its JavaScript	ECMA-262
	Browser	Window, Navigator	The browser	None (though some portions adhere to an adhoc standard)
	Document	Image, HTMLInputElement	The browser via its DOM engine	W3C DOM
9.	<b>Justify “JavaScript” is an event-driven programming”</b> Javascript supports event driven programming. when user clicks the mouse or hit the keys on the keyboard or if user submits the form then these events and response to them can be handled using javascript. Hence javascript is mainly used in web programming for validating the data provided by the user.			
10.	<b>What is the use of pop up boxes in java script?</b> There are three types of popup boxes used in javascript. Using these popup boxes the user can interact with the web application.			
11.	<b>What is DOM?</b> Document Object Model (DOM) is a set of platform independent and language neutral application interface (API) which describes how to access and manipulate the information stored in XML, XHTML and javascript documents.			
12.	<b>Enlist any four mouse events.</b> The MouseEvent are-mousedown, mouseup, mouseover, mousemove, mouseout.			
13.	<b>List ad various level of document object modeling.</b> Various levels of DOM are DOM0, Dom1, Dom2, and Dom3			
14.	<b>What are they validation properties and methods?</b> Validation properties and methods are checkvalidity (), validaionMessage, customerror, patternMismatch, rangeOverflow, rangeUnderflow, tooLong.			
15.	<b>Define event bubbling.</b> Suppose, there is an element present inside another element. Then during the event handling, if the event which is			

	present in the inner element is handled and then the event of the outer element is handled. This process of event handling is called event bubbling
<b>16.</b>	<p><b>How to create arrays in Javascript?</b></p> <p>We can declare an array like this <code>Var scripts = new Array();</code>  We can add elements to this array like this</p> <pre>scripts[0] = "PHP"; scripts[1] = "ASP"; scripts[2] = "JavaScript"; scripts[3] = "HTML";</pre> <p>Now our array scripts has 4 elements inside it and we can print or access them by using their index number. Note that index number starts from 0. To get the third element of the array we have to use the index number 2. Here is the way to get the third element of an array. <code>document.write (scripts[2]);</code> We also can create an array like this <code>var no_array = new Array(21, 22, 23, 24, 25);</code></p>
<b>17.</b>	<p><b>Write a simple program in JavaScript to validate the email-id.</b></p> <pre>&lt;!DOCTYPE html&gt; &lt;html&gt; &lt;head&gt; &lt;script&gt; function validateForm() {   var x = document.forms["myForm"]["email"].value;   var atpos = x.indexOf("@");   var dotpos = x.lastIndexOf(".");   if (atpos&lt;1    dotpos&lt;atpos+2    dotpos+2&gt;=x.length) {     alert("Not a valid e-mail address");     return false;}} &lt;/script&gt; &lt;/head&gt; &lt;body&gt; &lt;form name="myForm" action="demo_form.asp" onsubmit="return validateForm();" method="post"&gt; Email: &lt;input type="text" name="email"&gt; &lt;input type="submit" value="Submit"&gt; &lt;/form&gt; &lt;/body&gt; &lt;/html&gt;</pre>
<b>18.</b>	<p><b>Write short notes on JDBC.</b></p> <p>JDBC standard is intended for people developing industrial-strength database applications. JDBC makes java effective for developing enterprise information system. <code>java.sql</code> is the JDBC package that contains classes &amp; interfaces that enable a java program to interact with a database.</p>
<b>19.</b>	<p><b>Write short notes on JDBC drivers.</b></p> <p>A JDBC driver is basically an implementation of the function calls specified in the JDBC API for a particular vendor's RDBMS. Hence, a java program with JDBC function calls can access any RDBMS that has a JDBC driver available. A driver manager is used to keep track of all the installed drivers on the system. The operations of driver manager are <code>getDriver</code>, <code>registerDriver</code>, <code>deregisterDriver</code>.</p>
<b>20.</b>	<p><b>What are the advantages of servlet over CGI?</b></p> <ul style="list-style-type: none"> <li>➤ Performance is significantly better, servlet execute within the address space of a web server.</li> <li>➤ Servlets are platform independent</li> <li>➤ The java security manager on the server enforces a set of restrictions to protect the resources on a server machine.</li> <li>➤ The full functionality of java class libraries is available to a servlet.</li> </ul>
<b>21.</b>	<b>Write down the methods of servlet interface</b>

	<p>void destroy() –called when the servlet is unloaded.</p> <p>ServletConfig getServletConfig() –returns a ServletConfig object that contains any initialization parameters.</p> <p>String getServletInfo() – returns a string describing the servlet.</p> <p>void init(ServletConfig sc) throws ServletException –called when the servlet is initialized .Initialization parameters for servlet can be obtained from sc. An unavailable exception should be thrown if the servlet is not initialized.</p> <p>Void Service(ServletRequest req,ServletResponse res) throws ServletException, IOException- Called to process a request from a client. The request from the client can be read from req. response to the client can be written to res. An exception is generated if a servlet or IO problem occurs.</p>
22.	<p><b>What is the difference between CGI and servlets?</b></p> <ul style="list-style-type: none"> <li>➤ Performance is significantly better, servlet execute within the address space of a web server.</li> <li>➤ Servlets are platform independent</li> <li>➤ The java security manager on the server enforces a set of restrictions to protect the resources on a server machine.</li> <li>➤ The full functionality of java class libraries is available to a servlet.</li> </ul>
23.	<p><b>Define Servlet Life Cycle?</b></p> <ul style="list-style-type: none"> <li>➤ <b>init( )</b> method - invoked when the servlet is first loaded into memory</li> <li>➤ <b>service( )</b> - called for each HTTP request (for processing)</li> <li>➤ <b>destroy( )</b> - unloads the servlet from its memory.</li> </ul>
24.	<p><b>What is JSP?</b></p> <p>JavaServer Pages (JSP) is a technology for developing web pages that support dynamic content which helps developers insert java code in HTML pages by making use of special JSP tags, most of which start with &lt;% and end with %&gt;.</p>
25.	<p><b>What are advantages of using JSP?</b></p> <ul style="list-style-type: none"> <li>• Performance is significantly better because JSP allows embedding Dynamic Elements in HTML Pages itself.</li> <li>• JSP are always compiled before it's processed by the server unlike CGI/Perl which requires the server to load an interpreter and the target script each time the page is requested.</li> </ul>
26.	<p><b>Explain lifecycle of a JSP.</b></p> <ul style="list-style-type: none"> <li>• Compilation</li> <li>• Initialization</li> <li>• Execution</li> <li>• Cleanup</li> </ul>
27.	<p><b>What are the types of directive tags?</b></p> <p>The types directive tags are as follows:</p> <ul style="list-style-type: none"> <li>• &lt;%@ page ... %&gt; : Defines page-dependent attributes, such as scripting language, error page, and buffering requirements.</li> <li>• &lt;%@ include ... %&gt; : Includes a file during the translation phase.</li> <li>• &lt;%@ taglib ... %&gt; : Declares a tag library, containing custom actions, used in the page.</li> </ul>
28.	<p><b>What are JSP actions?</b></p> <p>JSP actions use constructs in XML syntax to control the behavior of the servlet engine. You can dynamically insert a file, reuse JavaBeans components, forward the user to another page, or generate HTML for the Java plugin.</p>
<b>Part - B</b>	
1.	<p><b>How to write function using Java Script? Give Example.</b></p> <p>A JavaScript function is a block of code designed to perform a particular task.A JavaScript function is executed when "something" invokes it (calls it).</p> <p>A JavaScript function is defined with the function keyword, followed by a name, followed by parentheses ().</p> <p>Function names can contain letters, digits, underscores, and dollar signs (same rules as variables).</p>

	<p>The parentheses may include parameter names separated by commas: (parameter1, parameter2, ...)</p> <p>The code to be executed, by the function, is placed inside curly brackets: {}</p> <p>Example:</p> <pre>&lt;!DOCTYPE html&gt; &lt;html&gt;  &lt;head&gt; &lt;script&gt; function myFunction() {   document.getElementById("demo").innerHTML = "Paragraph changed."; } &lt;/script&gt; &lt;/head&gt;  &lt;body&gt;  &lt;h1&gt;My Web Page&lt;/h1&gt;  &lt;p id="demo"&gt;A Paragraph&lt;/p&gt;  &lt;button type="button" onclick="myFunction()"&gt;Try it&lt;/button&gt;  &lt;/body&gt; &lt;/html&gt;</pre>
2.	<p><b>Explain sub classes and super classes in Javascript.</b></p> <p>The top-most class, the class from which all other classes are derived, is the Object class defined in java.lang . Object is the root of a hierarchy of classes, as illustrated in the following figure. The subclass inherits state and behavior in the form of variables and methods from its superclass.</p> <p>the Class.create() method. Until now the only feature of classes defined this way was that the constructor called a method called initialize automatically.</p> <pre>var Person = Class.create(); Person.prototype = { initialize: function(name) {   this.name = name; }, say: function(message) {   return this.name + ': ' + message; } }; var guy = new Person('Miro'); guy.say('hi'); // -&gt; "Miro: hi" var Pirate = Class.create(); // inherit from Person class: Pirate.prototype = Object.extend(new Person(), { // redefine the speak method say: function(message) {</pre>

```

 return this.name + ': ' + message + ', yarr!';
 }
});

```

```

 var john = new Pirate('Long John');
john.say('ahoy matey');
// -> "Long John: ahoy matey, yarr!"

```

Observe the direct interaction with class prototypes and the clumsy inheritance technique using `Object.extend`. Also, with `Pirate` redefining the `say()` method of `Person`, there is no way of calling the overridden method like you can do in programming languages that support class-based inheritance.

**Example:**

```

<script language="javascript" type="text/javascript">
<!--
function superClass() {
 this.supertest = superTest; //attach method superTest
}
function subClass() {
 this.inheritFrom = superClass;
 this.inheritFrom();
 this.subtest = subTest; //attach method subTest
}
function superTest() {
 return "superTest";
}
function subTest() {
 return "subTest";
}

var newClass = new subClass();
alert(newClass.subtest()); // yields "subTest"
alert(newClass.supertest()); // yields "superTest"

//-->
</script>

```

3. **Discuss Javascript objects in detail with suitable examples. (NOV/DEC 2012, MAY/JUNE 2014)**

- An object is a set of properties
- A property consists of a unique (within an object) name with an associated value
- The type of a property depends on the type of its value and can vary dynamically
- Object properties do not have data types  
Ex: Single property `prop` of an object `o`

```
o.prop = true; prop is Boolean
o.prop = "true"; prop is now String
o.prop = 1; prop is now Number
```

- There are no classes in JavaScript
- Object constructors defined to create objects and automatically define properties for the objects created
- Instead, properties and methods can be created and deleted dynamically

```
var o1 = new Object();
o1.testing = "This is a test";
delete o1.testing;
```

Create an object o1  
Create property testing  
Delete testing property

- Objects are created using new expression
- First line creates a variable named o1 initialize its value of type object by calling built-in constructor Object()
 

```
new Object() Constructor and argument list
```
- Second line adds a property named testing to the o1 object and assigns a string value to this property
- A constructor is a function
  - When called via new expression, a new empty Object is created and passed to the constructor along with the argument values
  - Constructor performs initialization on the object
    - Can add properties and methods to object
    - Can add object to an inheritance hierarchy from which it can inherit additional properties and methods
- The Object() built-in constructor
  - Does not add any properties or methods directly to the object
  - default toString() and valueOf() methods (used for conversions to String and Number, resp.)

```
o1.testing = "This is a test";
```
- Assign a value to an object property
- Property does not exist in the object
- Property with the given name is created in the object and assigned the specified value
- delete used to remove a property from an object
- Object initializer notation can be used to create an object (using Object() constructor) and one or more properties in a single statement

### Enumerating Properties

- To know which property an object has at any given time
- Special form of for statement used to iterate through all properties of an object:

```
var hash = new Object();

hash.kim = "85";
hash.sam = "92";
hash.lynn = "78";
for (var aName in hash) {
 window.alert(aName + " is a property of hash.");
}
```

### Array notation

- To print the values of those properties
- The JavaScript object dot notation is actually shorthand for a more general associative array notation in which Strings are array indices:
- Expressions can supply property names:
- Two different notations for accessing properties
- Dot notation
- Array reference syntax à index is viewed as string value hash
- Object can be viewed as a sort of array in which the elements are indexed by strings called associative

array

### Object reference

```
StringBuffer s1 = new StringBuffer("Hello");
StringBuffer s2 = s1;
```

- Single StringBuffer is created and both s1 and s2 will be references to it
- Copies the reference from s1 to s2
- If code followed by s2.append("World!");  
System.out.println(s1);

- JavaScript: Value of Object is reference to object;

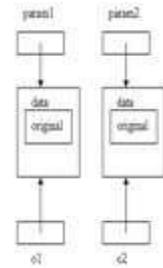
```
var o1 = new Object();
o1.data = "Hello";
var o2 = o1;
o2.data += " World!";
window.alert(o1.data);
```

o2 is another name for o1

o1 is changed

Output? → Hello World!

- Object argument values are references
- State of variables and parameters immediately before the first statement of the function objArgs() is executed

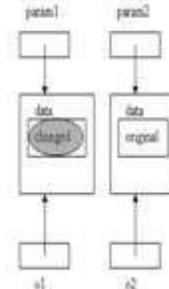


### Object Values

- immediately after the first statement of the function objArgs() is executed

```
function objArgs(param1, param2)
{
 // Change the data in param1 and its argument
 param1.data = "changed";
 // Change the object referenced by param2, but not its argument
 param2 = param1;

 window.alert("param1 is " + param1.data + "\n" +
 "param2 is " + param2.data);
 return;
}
```



	<p>• immediately after the second statement of the function objArgs() is executed</p> <p>• Object argument values are references</p> <pre> function objArgs(param1, param2) {   // Change the data in param1 and its argument   param1.data = "changed";   // Change the object referenced by param2, but not its argument   param2 = param1; }         </pre> <p><b>Methods</b></p> <ul style="list-style-type: none"> <li>• JavaScript functions are stored as values of type Object</li> <li>• A function declaration creates a function value and stores it in a variable (property of window) having the same name as the function</li> </ul> <p>A method is an object property for which the value is a function</p>
<p>4.</p>	<p><b>Discuss about Javascript debugging. Explain how local and global functions can be written using java script (MAY/JUNE 2012)</b></p> <p>A debugger is an application that places all aspects of script execution under the control of the programmer. Debuggers provide fine-grained control over the state of the script through an interface that allows you to examine and set values as well as control the flow of execution. Once a script has been loaded into a debugger, it can be run one line at a time or instructed to halt at certain breakpoints. Once execution is halted, the programmer can examine the state of the script and its variables in order to determine if something is amiss. You can also watch variables for changes in their values. The latest version of the Mozilla JavaScript Debugger for both Mozilla and Netscape browsers</p> <p><b>Local and global functions</b></p> <p>When a function is defined certain variables used for storing values are incorporated inside the function. These variables are found and used only inside these functions. Since functions are separate from the main code, it's advisable to use variables that are initialized only when a function is called and die when the execution comes out of the function. Variables that exist only inside a function are called Local variables. They have no presence outside the function. The values of such Local variables cannot be changed by the main code or other functions. This results in easy code maintenance and is especially helpful if many programmers are working together on the same project.</p> <p>Variables that exist throughout the script are called Global variables. Their values can be changed anytime in the code and even by other functions.</p> <p>In JavaScript, an inner (nested) function stores references to the local variables that are present in the same scope as the function itself, even after the function returns. This set of references is called a closure.</p> <pre> function myFunction() {   var a = 4;   return a * a; }         </pre> <p>The JavaScript global properties and functions can be used with all the built-in JavaScript objects.</p> <pre> var uri = "my test.asp?name=ståle&amp;car=saab"; var enc = encodeURI(uri); var dec = decodeURI(enc); var res = enc + "&lt;br&gt;" + dec;         </pre>
<p>5.</p>	<p><b>Explain the way in which java script handles arrays with example. (MAY/JUNE 2012)</b></p> <p><b>Array</b></p>

An array is a special variable, which can hold more than one value at a time. If you have a list of items (a list of car names, for example), storing the cars in single variables could look like this:

```
var car1 = "Saab";
var car2 = "Volvo";
var car3 = "BMW";
```

### Creating an Array

Using an array literal is the easiest way to create a JavaScript Array.

Syntax:

```
var array-name = [item1, item2, ...];
```

Example:

```
var cars = ["Saab", "Volvo", "BMW"];
```

### Using the JavaScript Keyword new

The following example also creates an Array, and assigns values to it:

```
Example: var cars = new Array("Saab", "Volvo", "BMW");
```

### Access the Elements of an Array

You refer to an array element by referring to the index number.

This statement access the value of the first element in myCars:

```
var name = cars[0];
```

This statement modifies the first element in cars:

```
cars[0] = "Opel";
```

Note [0] is the first element in an array. [1] is the second. Array indexes start with 0.

### Different Objects in One Array

JavaScript variables can be objects. Arrays are special kinds of objects.

Because of this, you can have variables of different types in the same Array.

You can have objects in an Array. You can have functions in an Array. You can have arrays in an Array:

```
myArray[0] = Date.now;
myArray[1] = myFunction;
myArray[2] = myCars;
```

### Arrays are Objects

Arrays are a special type of objects. The typeof operator in JavaScript returns "object" for arrays. But, JavaScript arrays are best described as arrays. Arrays use numbers to access its "elements". In this example, person[0] returns John:

```
Array: var person = ["John", "Doe", 46];
```

```
Object: var person = {firstName:"John", lastName:"Doe", age:46};
```

### Array Properties and Methods

The real strength of JavaScript arrays are the built-in array properties and methods:

Examples

```
var x = cars.length; // The length property returns the number of elements in cars
```

```
var y = cars.sort(); // The sort() method sort cars in alphabetical order
```

### The length Property

The length property of an array returns the length of an array (the number of array elements). Example

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
```

```
fruits.length; // the length of fruits is 4
```

### Adding Array Elements

The easiest way to add a new element to an array is to use the length property:

Example

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
```

```
fruits[fruits.length] = "Lemon"; // adds a new element (Lemon) to fruits
Adding elements with high indexes can create undefined "holes" in an array:
```

Example

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits[10] = "Lemon"; // adds a new element (Lemon) to fruits
```

### Looping Array Elements

The best way to loop through an array is using a standard for loop:

Example

```
var index; var fruits = ["Banana", "Orange", "Apple", "Mango"];
for (index = 0; index < fruits.length; index++) {
 text += fruits[index]; }
```

6. **i) Write a Java script to find the factorial of the given number.**

```
<html>
<head>
<script type="text/javascript">
function factorial(f,n)
{
l=1;
for(i=1;i<=n;i++)
l=l*i;
f.p.value=l;
}
</script>
</head>
<body>
<form>
number:<input type="text" name="t"></br>
<input type="button" value="submit" onClick="factorial(this.form,t.value)"></br>
result:<input type="text" name="p"></br>
</form>
</body>
</html>
```

**ii) Write a Java script to find the prime number between 1 and 100.**

```
<html>
<head>
<script language="javascript">
var n=prompt("Enter User Value")
var x=1;
if(n==0 || n==1) x=0;
for(i=2;i<n;i++)
{
if(n%i==0)
{
x=0;
break;
}
}
if(x==1)
```

```

 {
 alert(n + " "+" is prime");
 }
 else
 {
 alert(n + " "+" is not prime");
 }
 }

</script>
</head>
<body>
</html>

```

7. **Write a servlet program which displays the different content each time the user visits the page**

```

import java.io.*;
import java.sql.Date;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class PageHitCounter extends HttpServlet{

 private int hitCount;

 public void init()
 {
 hitCount = 0;
 }
 public void doGet(HttpServletRequest request,
 HttpServletResponse response)
 throws ServletException, IOException
 {
 // Set response content type
 response.setContentType("text/html");
 // This method executes whenever the servlet is hit
 // increment hitCount
 hitCount++;
 PrintWriter out = response.getWriter();
 String title = "Welcome User";
 String docType =
 "<!doctype html public "-//w3c//dtd html 4.0 " +
 "transitional//en">\n";
 out.println(docType +
 "<html>\n" +

```

```

"<head><title>" + title + "</title></head>\n" +
"<body bgcolor=#f0f0f0>\n" +
"<h1 align=center>" + title + "</h1>\n" +
"<h2 align=center>" + hitCount + "</h2>\n" +
"</body></html>");
}
public void destroy()
{
 // This is optional step but if you like you
 // can write hitCount value in your database.
}
}

```

Now let us compile above servlet and create following entries in web.xml

```

....
<servlet>
 <servlet-name>PageHitCounter</servlet-name>
 <servlet-class>PageHitCounter</servlet-class>
</servlet>

<servlet-mapping>
 <servlet-name>PageHitCounter</servlet-name>
 <url-pattern>/PageHitCounter</url-pattern>
</servlet-mapping>
....

```

8. **Write a Java script program to create Popup box, alert and confirm box.**
- ```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>Introduction to pop up box</title>
</head>
<body>
<p>Experiment with the popup boxes by clicking the buttons(OK and Cancel) on them</p>

<script type="text/javascript">
    if(confirm("do you agree?"))
        alert("You have agreed");
    else
        input_text=prompt("Enter some string here...", " ");
    /*the value entered in prompt box is returned
    and stored in the variable text */
    alert("Hi "+input_text);
</script>
</body>

```

| | |
|-----|--|
| | </html> |
| 9. | <p>Write a Java script program to print the numbers from 0 to 50.</p> <pre> <script type = "text/javascript"> var input; var i=0; input= 50; while (input > I) { document.write (i); i++ } </script> </pre> <p>b. Write a Java Script program to create table.</p> <pre> <html> function createTable(a) { document.getElementById("tbl").innerHTML = "<table border = '1'>" + " <tr>" + " <td>1</td>" + " <td>abc</td>" + " <td>123</td>" + " </tr>" + " <tr>" + " <td>2</td>" + " <td>def</td>" + " <td>456</td>" + " </tr>" + " <table>"; } </script> <div id="cnt"></div> <div id="tbl"></div> <input type="text" name="txtRow" style="width:200px;height:25px;" /><button name="cmdRow" style="width:125px;height:30px;" onclick="createTable()">Create Table</button> </html> </pre> |
| 10. | <p>Write a Java script program to create user registration form.</p> <pre> <!DOCTYPE html> <html lang="en"><head> <meta charset="utf-8"> <title>JavaScript Form Validation using a sample registration form</title> <meta name="keywords" content="example, JavaScript Form Validation, Sample registration form" /> <meta name="description" content="This document is an example of JavaScript Form Validation using a sample registration form. " /> </pre> |

```

<link rel='stylesheet' href='js-form-validation.css' type='text/css' />
<script src="sample-registration-form-validation.js"></script>
</head>
<body onload="document.registration.userid.focus();">
<h1>Registration Form</h1>
<p>Use tab keys to move from one input field to the next.</p>
<form name='registration' onSubmit="return formValidation();">
<ul>
<li><label for="userid">User id:</label></li>
<li><input type="text" name="userid" size="12" /></li>
<li><label for="passid">Password:</label></li>
<li><input type="password" name="passid" size="12" /></li>
<li><label for="username">Name:</label></li>
<li><input type="text" name="username" size="50" /></li>
<li><label for="address">Address:</label></li>
<li><input type="text" name="address" size="50" /></li>
<li><label for="country">Country:</label></li>
<li><select name="country">
<option selected="" value="Default">(Please select a country)</option>
<option value="AF">Australia</option>
<option value="AL">Canada</option>
<option value="DZ">India</option>
<option value="AS">Russia</option>
<option value="AD">USA</option>
</select></li>
<li><label for="zip">ZIP Code:</label></li>
<li><input type="text" name="zip" /></li>
<li><label for="email">Email:</label></li>
<li><input type="text" name="email" size="50" /></li>
<li><label id="gender">Sex:</label></li>
<li><input type="radio" name="msex" value="Male" /><span>Male</span></li>
<li><input type="radio" name="fsex" value="Female" /><span>Female</span></li>
<li><label>Language:</label></li>
<li><input type="checkbox" name="en" value="en" checked /><span>English</span></li>
<li><input type="checkbox" name="nonen" value="noen" /><span>Non English</span></li>
<li><label for="desc">About:</label></li>
<li><textarea name="desc" id="desc"></textarea></li>
<li><input type="submit" name="submit" value="Submit" /></li>
</ul>
</form>
</body>
</html>

```

11. i) **Explain any two validation function in java script.(4)**

JavaScript, provides a way to validate form's data on the client's computer before sending it to the web server. Form validation generally performs two functions.

- **Basic Validation** - First of all, the form must be checked to make sure data was entered into each form field that required it. This would need just loop through each field in the form and check for data.
- **Data Format Validation** - Secondly, the data that is entered must be checked for correct form and value. This would need to put more logic to test correctness of data.

```
<input id="id1" type="number" min="100" max="300">
```

```
<button onclick="myFunction()">OK</button>

<p id="demo"></p>

<script>
function myFunction() {
  var inpObj = document.getElementById("id1");
  if (inpObj.checkValidity() == false) {
    document.getElementById("demo").innerHTML = inpObj.validationMessage;
  }
}
</script>
```

ii) **Write a script to demonstrate the use of Date object.(6)**

JavaScript Date Formats

A JavaScript date can be written as a string:

Sat Jun 13 2015 10:24:39 GMT+0530 (India Standard Time)

or as a number:

1434171279721

Dates written as numbers, specifies the number of milliseconds since January 1, 1970, 00:00:00.

Displaying Dates

```
<p id="demo"></p>
<script>
document.getElementById("demo").innerHTML = Date();
</script>
```

Creating Date Objects

The Date object lets us work with dates.

A date consists of a year, a month, a day, an hour, a minute, a second, and milliseconds.

Date objects are created with the new Date () constructor.

There are 4 ways of initiating a date:

```
new Date()
new Date(milliseconds)
new Date(dateString)
new Date(year, month, day, hours, minutes, seconds, milliseconds)
```

Example:

```
<script>
var d = new Date();
document.getElementById("demo").innerHTML = d;
</script>
```

iii) **Write a java script program to generate Fibonacci series using do while loop.(6)**

```
<html>
<body>
<script type="text/javascript">
var a=0,b=1,c;
document.write("Fibonacci");
```

	<pre> while (b<=10) { document.write(c); document.write("
"); c=a+b; a=b; b=c; } </script> </body> </html> </pre>
12.	<p>i) Explain JavaScript & document object model (DOM) with example.(8)</p> <pre> import java.io.File; import java.util.Scanner; import javax.xml.parsers.DocumentBuilder; import javax.xml.parsers.DocumentBuilderFactory; import org.w3c.dom.Document; import org.w3c.dom.Element; import org.w3c.dom.Node; import org.w3c.dom.NodeList; public class NewClass { public static void main(String args[]) { try { File stocks = new File("C:\\Users\\Administrator\\Documents\\NetBeansProjects\\WebApplication3\\newXMLDocument.xml"); DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance(); DocumentBuilder dBuilder = dbFactory.newDocumentBuilder(); Document doc = dBuilder.parse(stocks); doc.getDocumentElement().normalize(); System.out.println("root of xml file" + doc.getDocumentElement().getNodeName()); NodeList nodes = doc.getElementsByTagName("stock"); System.out.println("Enter the user id"); Scanner scan=new Scanner(System.in); String s=scan.next(); System.out.println("====="); for (int i = 0; i < nodes.getLength(); i++) { Node node = nodes.item(i); if (node.getNodeType() == Node.ELEMENT_NODE) { Element element = (Element) node; if(s.equals(getValue("userid", element))) { System.out.println("Stock User Id: " + getValue("userid", element)); System.out.println("Stock Symbol: " + getValue("symbol", element)); } } } } </pre>

```

System.out.println("Stock Price: " + getValue("price", element));
System.out.println("Stock Quantity: " + getValue("quantity", element));
}
}
}
} catch (Exception ex) {
ex.printStackTrace();
}
}
private static String getValue(String tag, Element element) {
NodeList nodes = element.getElementsByTagName(tag).item(0).getChildNodes();
Node node = (Node) nodes.item(0);
return node.getNodeValue();
}
}

```

ii) Explain in details the JDBC CONNECTIVITY with example program.(8)

```

import java.io.*;
import java.util.*; import
javax.servlet.*; import
javax.servlet.http.*; import
java.sql.*;
public class DatabaseAccess extends HttpServlet{

public void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException
{
// JDBC driver name and database URL
static final String JDBC_DRIVER="com.mysql.jdbc.Driver";
static final String DB_URL="jdbc:mysql://localhost/TEST";

// Database credentials
static final String USER = "root";
static final String PASS = "password";

// Set response content type
response.setContentType("text/html");
PrintWriter out = response.getWriter();
String title = "Database Result";
String docType =
"<!doctype html public "-//w3c//dtd html 4.0 " +

```

```

"transitional//en\>\n";
out.println(docType +
"<html>\n" +
"<head><title>" + title + "</title></head>\n" +
"<body bgcolor=\"#f0f0f0\>\n" +
"<h1 align=\"center\>" + title + "</h1>\n");
try{
    // Register JDBC driver
    Class.forName("com.mysql.jdbc.Driver");

    // Open a connection
    conn = DriverManager.getConnection(DB_URL,USER,PASS);

    // Execute SQL query
    stmt = conn.createStatement();
    String sql;
    sql = "SELECT id, first, last, age FROM Employees";
    ResultSet rs = stmt.executeQuery(sql);

    // Extract data from result set
    while(rs.next()){
        //Retrieve by column name
        int id = rs.getInt("id");
        int age = rs.getInt("age");
        String first = rs.getString("first");
        String last = rs.getString("last");

        //Display values
        out.println("ID: " + id + "<br>");
        out.println(", Age: " + age + "<br>");
        out.println(", First: " + first + "<br>");
        out.println(", Last: " + last + "<br>");
    }
    out.println("</body></html>");

    // Clean-up environment
    rs.close();
    stmt.close();
    conn.close();
}catch(SQLException se){

```

```

//Handle errors for JDBC
se.printStackTrace();
}catch(Exception e){
//Handle errors for Class.forName
e.printStackTrace();
}finally{
//finally block used to close resources
try{
if(stmt!=null)
stmt.close();
}catch(SQLException se2){
} // nothing we can do
try{
if(conn!=null)
conn.close();
}catch(SQLException se){
se.printStackTrace();
} //end finally try
} //end try
}
}

```

13. **Explain the JDBC database access in detail. Write a java servlet to conduct online examination. (APR/MAY 2013)**

Index.html

```

<% @page contentType="text/html" pageEncoding="UTF-8"% >
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head><title>Database Test</title></head>
<body>
<center>
<h1>Online Examination</h1>
</center>
<form action="Servlet_tier1" method="POST">
<!--SERVLET NAME IN ACTION ATTRIBUTE -->
<div align="left"><br></div>
<b>Seat Number:</b> <input type="text" name="Seat_no">
<div align="Right">

```

```

<b>Name:</b> <input type="text" name="Name" size="50"><br>
</div>
<br><br>
<b>1. Every host implements transport layer.</b><br>
<input type="radio" name="group1" value="True">True
<input type="radio" name="group1" value="False">False<br>
<b>2. It is a network layer's responsibility to forward packets reliably from source to destination</b><br>
<input type="radio" name="group2" value="True">True
<input type="radio" name="group2" value="False">False<br>
<b>3. Packet switching is more useful in bursty traffic</b><br>
<input type="radio" name="group3" value="True">True
<input type="radio" name="group3" value="False">False<br>
<b>4. A phone network uses packet switching</b><br>
<input type="radio" name="group4" value="True">True
<input type="radio" name="group4" value="False">False<br>
<b>5. HTML1 is a Protocol for describing web contents</b><br>
<input type="radio" name="group5" value="True">True
<input type="radio" name="group5" value="False">False<br>
<br><br><br>
<center>
<input type="submit" value="Submit"><br><br>
</center>
</form>
</body>
</html>

```

SERVLET_TIER1.JAVA:

```

import java.io.*;
import java.sql.*;
import javax.servlet.*;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.*;

@WebServlet(name="Servlet_tier1", urlPatterns={"/Servlet_tier1"})
public class Servlet_tier1 extends HttpServlet

```

```

{
String message,Seat_no,Name,ans1,ans2,ans3,ans4,ans5;
int Total=0;
java.sql.Connection connect;
java.sql.Statement stmt=null;
java.sql.ResultSet rs=null;
public void doPost(HttpServletRequest request,HttpServletResponse response) throws
ServletException,IOException
{
try
{
String url="jdbc:odbc:NEO";// Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
connect=DriverManager.getConnection(url,"","");
message="Thank you for participating in online Exam";
}
catch(ClassNotFoundException cnfex){
cnfex.printStackTrace();
}
catch(SQLException sqlx){
sqlx.printStackTrace();
}
catch(Exception excp){
excp.printStackTrace();
}
Seat_no=request.getParameter("Seat_no");
Name=request.getParameter("Name");
ans1=request.getParameter("group1");
ans2=request.getParameter("group2");
ans3=request.getParameter("group3");
ans4=request.getParameter("group4");
ans5=request.getParameter("group5");
if(ans1.equals("True"))
Total+=2;
}

```

```

if(ans2.equals("False"))
Total+=2;
if(ans3.equals("True"))
Total+=2;
if(ans4.equals("False"))
Total+=2;
if(ans5.equals("False"))
Total+=2;
try
{
stmt=connect.createStatement();
String query="INSERT INTO
student("+ "Seat_no,Name,Total"+"")VALUES("+ "Seat_no+", "+ "Name+", "+ "Total+"");
stmt.executeUpdate(query);
stmt.close();
}catch(SQLException ex){
}
response.setContentType("text/html");
PrintWriter out=response.getWriter();
out.println("<html>");
out.println("<head>");
out.println("</head>");
out.println("<body bgcolor=cyan>");
out.println("<center>");
out.println("<h1>"+message+"</h1>\n");
out.println("<h3>Yours results stored in our database</h3>");
out.print("<br><br>");
out.println("<b>"+ "Participants and their Marks"+"</b>");
out.println("<table border=5>");
try
{
java.sql.Statement stmt2=connect.createStatement();
String query="SELECT * FROM student";

```

```
rs=stmt2.executeQuery(query);
out.println("<th>"+ "Seat_no"+ "</th>");
out.println("<th>"+ "Name"+ "</th>");
out.println("<th>"+ "Marks"+ "</th>");
while(rs.next())
{ out.println("<tr>");
out.print("<td>"+rs.getString(1)+"</td>");
out.print("<td>"+rs.getString(2)+"</td>");
out.print("<td>"+rs.getString(3)+"</td>");
out.println("</tr>");
}
out.println("</table>");
}
catch(SQLException ex){ }
finally
{
try
{ if(rs!=null)
rs.close();
if(stmt!=null)
stmt.close();
if(connect!=null)
connect.close();
} catch(SQLException e){ }
} out.println("</center>");
out.println("</body></html>");
Total=0;
} }
```

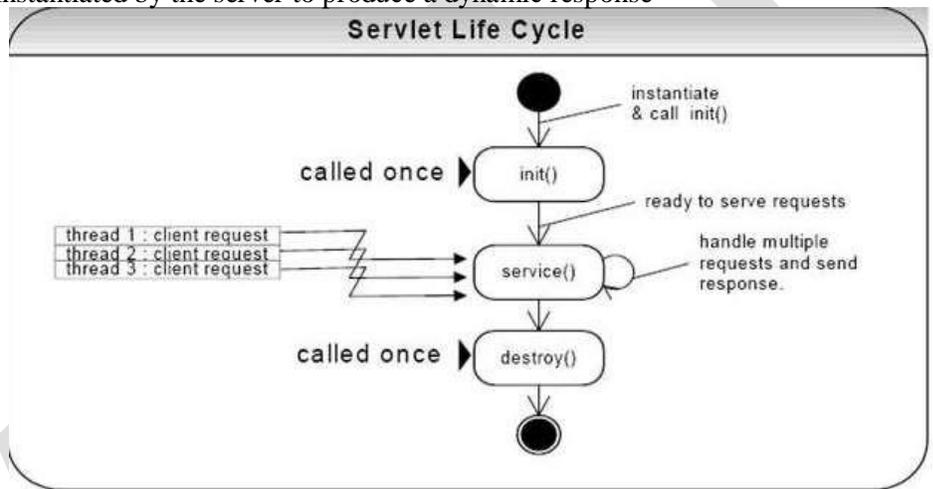
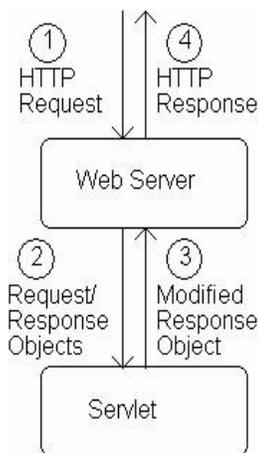
14. **What is a servlet? Explain briefly the Servlet life cycle and Servlet HTTP package?**

A servlet is a Java programming language class that is used to extend the capabilities of servers that host

applications accessed by means of a request-response programming model. Although servlets can respond to any type of request, they are commonly used to extend the applications hosted by web servers. For such applications, Java Servlet technology defines HTTP-specific servlet classes.

The `javax.servlet` and `javax.servlet.http` packages provide interfaces and classes for writing servlets. All servlets must implement the `Servlet` interface, which defines life-cycle methods. When implementing a generic service, you can use or extend the `GenericServlet` class provided with the Java Servlet API. The `HttpServlet` class provides methods, such as `doGet` and `doPost`, for handling HTTP-specific services.

- Static: HTML document is retrieved from the file system and returned to the client
- Dynamic: HTML document is generated by a program in response to an HTTP request
- Java servlets are one technology for producing dynamic server responses
 - Servlet is a class instantiated by the server to produce a dynamic response



What are all the Servlet API life cycle methods

Servlet API life cycle methods

- `init()`: called when servlet is instantiated; must return before any other methods will be called
- `service()`: method called directly by server when an HTTP request is received; default `service()` method calls `doGet()` (or related methods covered later)
- `destroy()`: called when server shuts down

PARAMETER DATA:

- The request object (which implements `HttpServletRequest`) provides information from the HTTP request to the servlet
- One type of information is parameter data, which is information from the query string portion of the HTTP request

query string with one parameter

```
http://www.example.com/servlet/PrintThis?arg=aString
```

parameter name: arg

parameter value: aString

GET vs. POST method for forms:

- GET:
 - Query string is part of URL
 - Length of query string may be limited

- Recommended when parameter data is not stored but used only to request information (e.g., search engine query)
- POST:
 - Query string is sent as body of HTTP request
 - Length of query string is unlimited
 - Recommended if parameter data is intended to cause the server to update stored data
 - Most browsers will warn you if they are about to resubmit POST data to avoid duplicate updates

15. **List out the classes and interfaces available in javax.servlet.http package?**

The javax.servlet.http package contains a number of classes and interfaces that describe and define the contracts between a servlet class running under the HTTP protocol and the runtime environment provided for an instance of such a class by a conforming servlet container.

Interface:

<u>HttpServletRequest</u>	Extends the <u>ServletRequest</u> interface to provide request information for HTTP servlets.
<u>HttpServletResponse</u>	Extends the <u>ServletResponse</u> interface to provide HTTP-specific functionality in sending a response.
<u>HttpSession</u>	Provides a way to identify a user across more than one page request or visit to a Web site and to store information about that user.
<u>HttpSessionActivationListener</u>	Objects that are bound to a session may listen to container events notifying them that sessions will be passivated and that session will be activated.
<u>HttpSessionAttributeListener</u>	This listener interface can be implemented in order to get notifications of changes to the attribute lists of sessions within this web application.
<u>HttpSessionBindingListener</u>	Causes an object to be notified when it is bound to or unbound from a session.
<u>HttpSessionContext</u>	Deprecated. As of Java(tm) Servlet API 2.1 for security reasons, with no replacement.
<u>HttpSessionListener</u>	Implementations of this interface are notified of changes to the list of active sessions in a web application.

Class:

<u>Cookie</u>	Creates a cookie, a small amount of information sent by a servlet to a Web browser, saved by the browser, and later sent back to the server.
<u>HttpServlet</u>	Provides an abstract class to be subclassed to create an HTTP servlet suitable for a Web site.
<u>HttpServletRequestWrapper</u>	Provides a convenient implementation of the <u>HttpServletRequest</u> interface that can be subclassed by developers wishing to adapt the request to a Servlet.
<u>HttpServletResponseWrapper</u>	Provides a convenient implementation of the <u>HttpServletResponse</u> interface that can be subclassed by developers wishing to adapt the response from a Servlet.
<u>HttpSessionBindingEvent</u>	Events of this type are either sent to an object that

		implements HttpSessionBindingListener when it is bound or unbound from a session, or to a HttpSessionAttributeListener that has been configured in the deployment descriptor when any attribute is bound, unbound or replaced in a session.
	HttpSessionEvent	This is the class representing event notifications for changes to sessions within a web application.
	HttpUtils	Deprecated. As of Java(tm) Servlet API 2.3.
16.	<p>Write short notes on the following servlet classes GenericServlet, ServletInputStream, ServletOutputStream and ServletException</p> <p>GenericServlet: GenericServlet class implements Servlet, ServletConfig and Serializable interfaces. It provides the implementation of all the methods of these interfaces except the service method. GenericServlet class can handle any type of request so it is protocol-independent. You may create a generic servlet by inheriting the GenericServlet class and providing the implementation of the service method. Methods of GenericServlet class There are many methods in GenericServlet class. They are as follows:</p> <ol style="list-style-type: none"> 1. public void init(ServletConfig config) is used to initialize the servlet. 2. public abstract void service(ServletRequest request, ServletResponse response) provides service for the incoming request. It is invoked at each time when user requests for a servlet. 3. public void destroy() is invoked only once throughout the life cycle and indicates that servlet is being destroyed. 4. public ServletConfig getServletConfig() returns the object of ServletConfig. 5. public String getServletInfo() returns information about servlet such as writer, copyright, version etc. 6. public void init() it is a convenient method for the servlet programmers, now there is no need to call super.init(config) 7. public ServletContext getServletContext() returns the object of ServletContext. 8. public String getInitParameter(String name) returns the parameter value for the given parameter name. 9. public Enumeration getInitParameterNames() returns all the parameters defined in the web.xml file. 10. public String getServletName() returns the name of the servlet object. 11. public void log(String msg) writes the given message in the servlet log file. 12. public void log(String msg, Throwable t) writes the explanatory message in the servlet log file and a stack trace. <p>ServletInputStream: ServletInputStream class provides stream to read binary data such as image etc. from the request object. It is an abstract class. The getInputStream() method of ServletRequest interface returns the instance of ServletInputStream class. So can be get as: ServletInputStream sin=request.getInputStream(); Method of ServletInputStream class There are only one method defined in the ServletInputStream class. int readLine(byte[] b, int off, int len) it reads the input stream.</p>	

ServletOutputStream:

ServletOutputStream class provides a stream to write binary data into the response. It is an abstract class.

The getOutputStream() method of ServletResponse interface returns the instance of ServletOutputStream class. It may be get as:

```
ServletOutputStream out=response.getOutputStream();
```

Methods of ServletOutputStream class

The ServletOutputStream class provides print() and println() methods that are overloaded.

1. void print(boolean b){ }
2. void print(char c){ }
3. void print(int i){ }
4. void print(long l){ }
5. void print(float f){ }
6. void print(double d){ }
7. void print(String s){ }
8. void println{ }
9. void println(boolean b){ }
10. void println(char c){ }
11. void println(int i){ }
12. void println(long l){ }
13. void println(float f){ }
14. void println(double d){ }
15. void println(String s){ }
16. public **ServletException**(java.lang.Throwable rootCause)

ServletException:

```
public ServletException(java.lang.String message,java.lang.Throwable rootCause)
```

Constructs a new servlet exception when the servlet needs to throw an exception and include a message about the "root cause" exception that interfered with its normal operation, including a description message.

Parameters:

message - a String containing the text of the exception message

rootCause - the Throwable exception that interfered with the servlet's normal operation, making this servlet exception necessary

Constructs a new servlet exception when the servlet needs to throw an exception and include a message about the "root cause" exception that interfered with its normal operation. The exception's message is based on the localized message of the underlying exception.

This method calls the getLocalizedMessage method on the Throwable exception to get a localized exception message. When subclassing ServletException, this method can be overridden to create an exception message designed for a specific locale.

Parameters:


```

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
public class InsertRecord extends HttpServlet {
    public void doPost(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException {

        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

        String name=request.getParameter("name");
        String en=request.getParameter("enrolmentNo");
        String program=request.getParameter("program");
        String gender=request.getParameter("gender");
        String address=request.getParameter("address");
        int id=0;
        int enrol=0;

        if(name.equals("") || en.equals("") ||
            program.equals("") || gender.equals("") ||
            address.equals(""))
        {
            out.println("Please insert valid data");
            RequestDispatcher rd =
                request.getRequestDispatcher("/index.html");
            rd.include(request, response);
        }
        else
        {
            enrol=Integer.parseInt(en);

            try{

```

```

Class.forName("oracle.jdbc.driver.OracleDriver");
Connection con=DriverManager.getConnection(
"jdbc:oracle:thin:@localhost:1521:xe","system","sunil");

PreparedStatement pst=con.prepareStatement(
"SELECT id FROM STUDENT");
ResultSet rs=pst.executeQuery();
while(rs.next())
{
    id=rs.getInt(1);
}

PreparedStatement ps=con.prepareStatement(
"insert into STUDENT values(?,?,?,?);");

ps.setInt(1,id+1);
ps.setString(2,name);
ps.setInt(3,enrol);
ps.setString(4,program);
ps.setString(5,gender);
ps.setString(6,address);

int i=ps.executeUpdate();
if(i>0)
out.print("Student record successfully inserted");
out.print("<BR>");
out.print("Insert another record ...");
RequestDispatcher
    rd = request.getRequestDispatcher("/index.html");
rd.include(request, response);
}
catch (Exception e) {
    System.out.println(e);
}
}
out.close();
}
}

```

Web.html

```

<?xml version="1.0" encoding="UTF-8"?>
<web-app id="WebApp_ID" version="2.5">

```

	<pre> <display-name>Student</display-name> <servlet> <description></description> <display-name>InsertRecord</display-name> <servlet-name>InsertRecord</servlet-name> <servlet-class>InsertRecord</servlet-class> </servlet> <servlet-mapping> <servlet-name>InsertRecord</servlet-name> <url-pattern>/InsertRecord</url-pattern> </servlet-mapping> <welcome-file-list> <welcome-file>index.html</welcome-file> </welcome-file-list> </web-app> </pre>
19.	<p>Explain in detail about JSP with an example of current date and simple message.</p> <pre> <% @page contentType="text/html" import="java.util.*" %> <html> <body> <p>&nbsp;</p> <div align="center"> <center> <table border="0" cellpadding="0" cellspacing="0" width="460" bgcolor="#EEFFCA"> <tr> <td width="100%">&nbsp;<Date Example</td> </tr> <tr> <td width="100%">&nbsp;<Current Date and time is:&nbsp; <%= new java.util.Date() %> </td> </tr> </table> </center> </div> </body> </html> </pre>
20.	<p>Discuss in detail about Action elements in JSP with an example of display current time and color.</p> <p>ACTION:</p> <ul style="list-style-type: none"> • Standard: provided by JSP itself • Custom: provided by a tag library such as JSTL. <ul style="list-style-type: none"> • JSTL is divided into several functional areas, each with its own namespace:

TABLE 8.6: JSTL functional areas.

Functional Area	Namespace Name Suffix
Core	core
XML Processing	xml
Functions	functions
Database	sql
Internationalization	fmt

Namespace prefix is

`http://java.sun.com/jsp/jstl/`

JSTL CORE ACTIONS:

`<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>`

TABLE 8.7: Some JSTL core actions.

Action	Purpose
set	Assign a value to a scoped variable, creating the variable if necessary
remove	Destroy a scoped variable
out	Write data to out implicit object, escaping XML special characters
url	Create a URL with query string
if	Conditional (if-then) processing
choose	Conditional (if-then-elseif) processing
forEach	Iterate over a collection of items

- Common variables:

- var

- Represents name of a scoped variable that is assigned to by the action
- Must be a string literal, not an EL expression

- scope

- Specifies scope of scoped variable as one of the literals page, request, session, or application

set action

- Setting (and creating) a scoped variable

out action

- Normally used to write a string to the out JSP implicit object
- Automatically escapes XML special characters

if action

- General form includes scoped variable to receive test value

```
<c:set var="age" value="20" scope="session"></c:set>
```

```
<c:if test="{age ge 18}" var="x">
<h3><c:out value="WELCOME"></c:out></h3>
</c:if>
```

```
<h3><c:out value="{x}"></c:out></h3>
```

```
<c:if test="{x}">
  <h3><c:out value="WELCOME"></c:out></h3>
</c:if>
```

Output:

WELCOME

true

WELCOME

remove action

- Only attributes are var and scope
- Removes reference to the specified scoped variable from the scope object

```
<c:set var="x" value="10" scope="session"></c:set>
<c:set var="y" value="20" scope="session"></c:set>
<h3>Product :<c:out value="{x*y}"></c:out></h3>
<c:remove var="x" scope="session"/>
<c:remove var="y" scope="session"/>
<h3>Product :<c:out value="{x*y}"></c:out></h3>
```

Output:

Product :200

Product :0

choose action:

```
<c:set var="salary" scope="session" value="5000"/>
  <p>Your salary is : <c:out value="{salary}"></p>
<c:choose>
  <c:when test="{salary > 1000}">
    Salary is very good.
  </c:when>
  <c:otherwise>
    Salary is very low
  </c:otherwise>
</c:choose>
```

Output:

Your salary is : 5000

Salary is very good.

forEach action:

Used to increment a variable (writes 2, 4, 6, 8 to the out object)

```
<c:forEach begin="1" end="10" step="4" var="x">
  Begin Index value :{x}<br>
</c:forEach>
```

Output:

Begin Index value :1

Begin Index value :5

Begin Index value :9

url action

- value attribute is a URL to be written to the out JSP implicit object
- URL's beginning with / are assumed relative to context path
- param elements can be used to define parameters that will be URL encoded

curl.jsp:

```
<c:url value="/URLTest.jsp" var="x" >
  <c:param name="d1" value="SCJP"/>
  <c:param name="d2" value="SCWCD"/>
</c:url>
<h1>The modified url : {x},</h1><br>
```

```
<a href="{x}">click Here </a>
```

Output:

The modified url : /JSTL/URLTest.jsp?d1=SCJP&d2=SCWCD,

URLTest.jsp

```
<h2> This is URLTest JSP </h2>
```

```
<h2>First Parameter : : ${param.d1} </h2>
```

```
<h2>Second Parameter : : ${param.d2} </h2>
```

Output:

URL:

This is URLTest JSP

First Parameter : : SCJP

Second Parameter : : SCWCD

21. **Explain about JSP object in detail.**

JSP Implicit Objects are the Java objects that the JSP Container makes available to developers in each page and developer can call them directly without being explicitly declared. JSP Implicit Objects are also called pre-defined variables.

JSP supports nine Implicit Objects which are listed below:

Object	Description
request	This is the HttpServletRequest object associated with the request.
response	This is the HttpServletResponse object associated with the response to the client.
out	This is the PrintWriter object used to send output to the client.
session	This is the HttpSession object associated with the request.
application	This is the ServletContext object associated with application context.
config	This is the ServletConfig object associated with the page.
pageContext	This encapsulates use of server-specific features like higher performance JspWriters.
page	This is simply a synonym for this, and is used to call the methods defined by the
Exception	The Exception object allows the exception data to be accessed by designated JSP.

The request Object:

The request object is an instance of a javax.servlet.http.HttpServletRequest object. Each time a client requests a page the JSP engine creates a new object to represent that request.

The request object provides methods to get HTTP header information including form data, cookies, HTTP methods etc.

We would see complete set of methods associated with request object in coming chapter: JSP - Client Request.

The response Object:

The response object is an instance of a `javax.servlet.http.HttpServletResponse` object. Just as the server creates the request object, it also creates an object to represent the response to the client.

The response object also defines the interfaces that deal with creating new HTTP headers. Through this object the JSP programmer can add new cookies or date stamps, HTTP status codes etc.

We would see complete set of methods associated with response object in coming chapter: [JSP - Server Response](#).

The out Object:

The out implicit object is an instance of a `javax.servlet.jsp.JspWriter` object and is used to send content in a response.

The initial `JspWriter` object is instantiated differently depending on whether the page is buffered or not. Buffering can be easily turned off by using the `buffered='false'` attribute of the page directive.

The `JspWriter` object contains most of the same methods as the `java.io.PrintWriter` class. However, `JspWriter` has some additional methods designed to deal with buffering. Unlike the `PrintWriter` object, `JspWriter` throws `IOException`s.

Following are the important methods which we would use to write boolean, char, int, double, object, String etc.

Method	Description
<code>out.print(dataType dt)</code>	Print a data type value
<code>out.println(dataType dt)</code>	Print a data type value then terminate the line with new line character.
<code>out.flush()</code>	Flush the stream.

The session Object:

The session object is an instance of `javax.servlet.http.HttpSession` and behaves exactly the same way that session objects behave under Java Servlets.

The session object is used to track client session between client requests. We would see complete usage of session object in coming chapter: [JSP - Session Tracking](#).

The application Object:

The application object is direct wrapper around the `ServletContext` object for the generated Servlet and in reality an instance of a `javax.servlet.ServletContext` object.

This object is a representation of the JSP page through its entire lifecycle. This object is created when the JSP page is initialized and will be removed when the JSP page is removed by the `jspDestroy()` method.

By adding an attribute to application, you can ensure that all JSP files that make up your web application have access to it.

You can check a simple use of Application Object in chapter: [JSP - Hits Counter](#)

The config Object:

The config object is an instantiation of `javax.servlet.ServletConfig` and is a direct wrapper around the `ServletConfig` object for the generated servlet.

This object allows the JSP programmer access to the Servlet or JSP engine initialization parameters such as the paths or file locations etc.

The following config method is the only one you might ever use, and its usage is trivial:

```
config.getServletName();
```

This returns the servlet name, which is the string contained in the `<servlet-name>` element defined in the `WEB-INF/web.xml` file

	<p>The pageContext Object: The pageContext object is an instance of a javax.servlet.jsp.PageContext object. The pageContext object is used to represent the entire JSP page. This object is intended as a means to access information about the page while avoiding most of the implementation details. This object stores references to the request and response objects for each request. The application, config, session, and out objects are derived by accessing attributes of this object. The pageContext object also contains information about the directives issued to the JSP page, including the buffering information, the errorPageURL, and page scope. The PageContext class defines several fields, including PAGE_SCOPE, REQUEST_SCOPE, SESSION_SCOPE, and APPLICATION_SCOPE, which identify the four scopes. It also supports more than 40 methods, about half of which are inherited from the javax.servlet.jsp. JspContext class. One of the important methods is removeAttribute, which accepts either one or two arguments. For example, pageContext.removeAttribute ("attrName") removes the attribute from all scopes, while the following code only removes it from the page scope:</p> <pre>pageContext.removeAttribute("attrName", PAGE_SCOPE);</pre> <p>You can check a very good usage of pageContext in coming chapter: JSP - File Uploading.</p> <p>The page Object: This object is an actual reference to the instance of the page. It can be thought of as an object that represents the entire JSP page. The page object is really a direct synonym for the this object.</p> <p>The exception Object: The exception object is a wrapper containing the exception thrown from the previous page. It is typically used to generate an appropriate response to the error condition.</p>																
Unit – IV																	
Part - A																	
1.	<p>What is PHP? PHP - Hypertext Preprocessor -one of the most popular server-side scripting languages for creating dynamic Web pages.</p> <ul style="list-style-type: none"> - an open-source technology - platform independent 																
2.	<p>List the data types used in PHP.</p> <table border="1" data-bbox="402 1312 1365 1591"> <thead> <tr> <th>Data types</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>Integer</td> <td>Whole numbers (i.e., numbers without a decimal point)</td> </tr> <tr> <td>Double</td> <td>Real numbers (i.e., numbers containing a decimal point)</td> </tr> <tr> <td>String</td> <td>Text enclosed in either single (") or double (") quotes.</td> </tr> <tr> <td>Boolean</td> <td>True or false</td> </tr> <tr> <td>Array</td> <td>Group of elements of the same type</td> </tr> <tr> <td>Object</td> <td>Group of associated data and methods</td> </tr> <tr> <td>Resource</td> <td>An external data source</td> </tr> </tbody> </table>	Data types	Description	Integer	Whole numbers (i.e., numbers without a decimal point)	Double	Real numbers (i.e., numbers containing a decimal point)	String	Text enclosed in either single (") or double (") quotes.	Boolean	True or false	Array	Group of elements of the same type	Object	Group of associated data and methods	Resource	An external data source
Data types	Description																
Integer	Whole numbers (i.e., numbers without a decimal point)																
Double	Real numbers (i.e., numbers containing a decimal point)																
String	Text enclosed in either single (") or double (") quotes.																
Boolean	True or false																
Array	Group of elements of the same type																
Object	Group of associated data and methods																
Resource	An external data source																
3.	<p>How type conversion is done in PHP? In PHP, data-type conversion can be performed by passing the data type as an argument to function settype. Function settype takes two arguments: The variable whose data type is to be changed and the variable 's new data type. E.g., settype(\$testString, "double");</p>																
4.	<p>Write the uses of text manipulation with regular expression in PHP.</p> <ul style="list-style-type: none"> • PHP processes text data easily and efficiently, enabling straightforward searching, substitution, extraction and concatenation of strings. 																

	<ul style="list-style-type: none"> Text manipulation in PHP is usually done with regular expressions — a series of characters that serve as pattern-matching templates (or search criteria) in strings, text files and databases. This feature allows complex searching and string processing to be performed using relatively simple expressions
5.	<p>List the important characteristics of PHP.</p> <p>The main characteristics of PHP are:</p> <ul style="list-style-type: none"> PHP is web-specific and open source Scripts are embedded into static HTML files Fast execution of scripts Fast access to the database tier of applications Supported by most web servers and operating systems Supports many standard network protocols libraries available for IMAP, NNTP, SMTP, POP3 Supports many database management systems libraries available for UNIX DBM, MySQL, Oracle, Dynamic Output any text, HTML XHTML and any other XML file. Also Dynamic Output images, PDF files and even Flash movies Text processing features, from the POSIX Extended or Perl regular expressions to parsing XML documents. A fully featured programming language suitable for complex systems development
6.	<p>How to Include PHP in a Web Page?</p> <p>There are 4 ways of including PHP in a web page</p> <ol style="list-style-type: none"> <code><?php echo("Hello world"); ?></code> <code><script language = "php"> echo("Hello world"); </script></code> <code><? echo("Hello world"); ?></code> <code><% echo("Hello world"); %></code> <p>we can also use print instead of echo</p> <ul style="list-style-type: none"> Method (1) is clear and unambiguous Method (2) is useful in environments supporting mixed scripting languages in the same HTML file Methods (3) and (4) depend on the server configuration
7.	<p>Write a simple PHP Script.</p> <p>Here is PHP script which is embedded in HTML using level one header with the PHP output text. The name of this file is called hello.php.</p> <pre><html> <head> <title>Hello world</title> </head> <body> <h1><?php echo("Hello world"); ?></h1> <h1><?php print("This prints the same thing!");?></h1> </body> </html></pre>
8.	<p>How do you include comments in PHP?</p> <p>PHP supports three types of comments:</p> <ol style="list-style-type: none"> Shell style comments - denoted <code>#THIS IS A COMMENT</code> C++ style comments - denoted <code>THIS IS A COMMENT—</code> C style comments - denoted <code>/* ALL THIS COMMENTED! */</code>
9.	<p>What are variables in PHP?</p> <p>Variables start with the \$ symbol.</p> <p>E.g.:</p> <pre>\$myInteger = 3; \$myString = "Hello world";</pre>

	<code>\$myFloat = 3.145;</code>
10.	<p>How do you declare a variable using PHP data types?</p> <p>Data types are not explicitly defined:</p> <ul style="list-style-type: none"> • Variable type is determined by assignment • Strings can be defined with single (') and double (") quotes. • PHP has a Boolean type: <ul style="list-style-type: none"> Defined as false <ul style="list-style-type: none"> – An integer or float value of 0 or – The keyword false – The empty string "" or the string "0" – An empty array or object – The NULL value Defined as true <ul style="list-style-type: none"> – Any non-zero integer or float value – The keyword true • Standard operators with standard syntax applied to variables
11.	<p>How do you declare and initialize an array in PHP?</p> <p>Two ways of declaring and initializing an array:</p> <ol style="list-style-type: none"> Individual element initialization in an array <pre>\$myArray[0]= "Apples"; \$myArray[1]= "Bananas";</pre> Arrays can be constructed using the array() keyword <pre>\$person = array("Dave", "Adam", "Ralph");</pre>
12.	<p>What are associative arrays in PHP?</p> <pre>\$myArray["Monday"]= "Apples"; \$myArray["Tuesday"]= "Bananas";</pre> <p>Associative Arrays can also be constructed using the array() keyword. <pre>\$food = array("Monday"=>"Apples","Tuesday"=> "Bananas");</pre> The symbol => delimits the hash name from the hash value.</p>
13.	<p>What is the scope of variables in PHP?</p> <p>Once PHP variables have been defined they are known for the rest of the Web page:</p> <ul style="list-style-type: none"> • Obeying standard scoping rules of course. • Variables can be local to functions etc, much like any languages.
14.	<p>List some built in functions in PHP.</p> <p>Mathematical functions:- abs, ceil, cos, log, min, rand, sqrt File handling:- fopen, flock, feof, fgets, fputs, fclose</p>
15.	<p>List the PHP standard Flow-controls statements</p> <p>if, if/else switch while for</p>
16.	<pre>\$a=3; Function what() { ++\$a; echo "a=\$a\n"; } what(); echo "a=\$a\n";</pre>

	<p>What is the output? 1 3</p>
17.	<p>List the functions to create a pattern. Preg_match, Preg_matchall, Preg_replace, Preg_split</p>
18.	<p>Write a PHP script to set the background colour to blue on Tuesday in a given date.</p> <pre><?php if(date("D") == "Tue") \$colour = "blue"; else \$colour = "red"; ?> <html> <head> <title>Welcome</title> </head> <body bgcolor = <?php echo(\$colour) ?>> <h1>Welcome</h1> </body> </html></pre>
19.	<p>What is cookie? Give example in PHP A cookie is a text string stored on the client machine by your script (to track users and manage transactions). Cookies are automatically returned (by the client), and can be accessed using a variable of the same name</p> <ul style="list-style-type: none"> • The following script reads and displays a cookie, and sets it with a new value (string) that was passed to the script as a parameter. • The cookie will expire after 20 minutes (1200 seconds) <pre><?php setCookie("CookieTest", \$val, time()+1200); ?> <html> <head><title>Welcome</title></head> <body> <?php echo("<h2>The cookie is: \$CookieTest</h1> </body> </html></pre>
20.	<p>What is XML ? Extensible markup language. It offer a standard, flexible and inherently extensible data format, XML significantly reduces the burden of deploying the many technologies needed to ensure the success of Web services.</p>
21.	<p>Define XML attributes</p> <ul style="list-style-type: none"> • XML elements can have attributes in the start tag, just like HTML. • Attributes are used to provide additional information about elements. • Attributes cannot contain multiple values (child elements can) • Attributes are not easily expandable (for future changes)
22.	<p>Write the main difference between XML and HTML. <i>Main Difference between XML and HTML</i> XML was designed to carry data. XML is not a replacement for HTML. XML and HTML were designed with different goals: XML was designed to describe data and to focus on what data is. HTML was designed to display data and to focus on how data looks. HTML is about displaying information, while XML is about describing information</p>
23.	<p>What is meant by a XML namespace? (APR/MAY 2011)</p>

	<p>XML Namespaces provide a method to avoid element name conflicts. When using prefixes in XML, a so-called namespace for the prefix must be defined. The namespace is defined by the xmlns attribute in the start tag of an element. The namespace declaration has the following syntax. xmlns:prefix="URI".</p> <pre><root> <h:table xmlns:h="http://www.w3.org/TR/html4/"> <h:tr> <h:td>Apples</h:td> <h:td>Bananas</h:td> </h:tr> </h:table> <f:table xmlns:f="http://www.w3schools.com/furniture"> <f:name>African Coffee Table</f:name> <f:width>80</f:width> <f:length>120</f:length> </f:table> </root></pre>								
24.	<p>What is XML namespace? (NOV/DEC 2012) XML allows document authors to create custom elements.</p> <ul style="list-style-type: none"> This extensibility can result in naming collisions (i.e. different elements that have the same name) among elements in an XML document. <p>An XML namespace is a collection of element and attribute names. Each namespace has a unique name that provides a means for document authors to unambiguously refer to elements with the same name (i.e. prevent collisions).</p>								
25.	<p>What is the purpose of namespace? (MAY/JUNE 2014) XML Namespaces provide a method to avoid element name conflicts. In XML, element names are defined by the developer. This often results in a conflict when trying to mix XML documents from different XML applications.</p>								
26.	<p>Compare DOM and SAX in XML processing. (MAY/JUNE 2013)</p> <table border="1"> <thead> <tr> <th>DOM</th> <th>SAX</th> </tr> </thead> <tbody> <tr> <td>DOM is an interface-oriented Application Programming Interface.</td> <td>SAX parser works incrementally and generates events that are passed to the application.</td> </tr> <tr> <td>It allows for navigation of the entire document.</td> <td>DOM parser reads the whole XML document and returns a DOM tree representation of xml document.</td> </tr> <tr> <td>DOM allows you to read and write.</td> <td>SAX is essentially an API for reading XML</td> </tr> </tbody> </table>	DOM	SAX	DOM is an interface-oriented Application Programming Interface.	SAX parser works incrementally and generates events that are passed to the application.	It allows for navigation of the entire document.	DOM parser reads the whole XML document and returns a DOM tree representation of xml document.	DOM allows you to read and write.	SAX is essentially an API for reading XML
DOM	SAX								
DOM is an interface-oriented Application Programming Interface.	SAX parser works incrementally and generates events that are passed to the application.								
It allows for navigation of the entire document.	DOM parser reads the whole XML document and returns a DOM tree representation of xml document.								
DOM allows you to read and write.	SAX is essentially an API for reading XML								
27.	<p>What are complex types? complex types are an important aspects of xml schema that allow application developers to define application-specific data types that can be checked by programs that check XML document for validity. XML schema divides complex types into two categories: those with <i>simple content</i> & those with <i>complex content</i>.</p>								
28.	<p>What are all the Transformation techniques?</p> <ul style="list-style-type: none"> XSLT - it is an XML- based languages used to transform XML documents into others format such as HTML for web display. XLINK - highlighting that element or taking the user directly to that point in the document. XPATH - xpath gets its name from its use of a payh notation to navigate through the hierarchical tree structure of an XML document XQUERY - it is W3C initiative to define a standard set of constructs for querying & searching XML document. 								
29.	<p>What is XSLT?</p> <ul style="list-style-type: none"> XSLT stands for XSL Transformations XSLT is the most important part of XSL XSLT transforms an XML document into another XML document XSLT uses XPath to navigate in XML documents <p>XSLT is a W3C Recommendation</p>								
30.	<p>Define the term DTD.</p>								

	A Document Type Definition (DTD) defines the legal building blocks of an XML document. It defines the document structure with a list of legal elements and attributes.
31.	<p>List two types of DTD declaration</p> <p>DTD stands for Document Type Definition which is used to structure the XML document. The types of DTD are as follows: i) Internal Declaration ii) External Declaration.</p>
32.	<p>How to declare DTD attributes?</p> <p>An attribute declaration has the following syntax:</p> <pre><!ATTLIST element-name attribute-name attribute-type default-value></pre> <p>DTD example: <pre><!ATTLIST payment type CDATA "check"></pre> <p>XML example: <pre><payment type="check" /></pre></p> </p>
33.	<p>What is XML schema?</p> <p>An XML schema is itself an XML document. It provides more detail about the kind of data that can appear as part of an XML document.</p>
34.	<p>What is the purpose of XML schema? (APR/MAY 2013)</p> <ul style="list-style-type: none"> • The schemas are more specific and provide the support for data types. • The schema is aware of namespace • The XML Schema is written in XML itself and has a large number of built-in and derived types. • The XML schema is the W3C recommendation. Hence it is supported by various XML validators and XML Processors.
35.	<p>What are the disadvantages of schema?</p> <ul style="list-style-type: none"> • The XML schema is complex to design and hard to learn • The XML document cannot be if the corresponding schema file is absent. • Maintaining the schema for large and complex operations sometimes slows down the processing of XML documents.
36.	<p>Explain DTD for XML Schemas.</p> <ul style="list-style-type: none"> • XML documents are processed by applications • Applications have assumptions about XML documents • DTDs allow to formalize some of these constraints • Part of the constraint checking must still be programmed
37.	<p>List some browsers that support XML and XSL</p> <p>Mozilla Firefox As of version 1.0.2, Firefox has support for XML and XSLT (and CSS). Mozilla: Mozilla includes Expat for XML parsing and has support to display XML + CSS. Mozilla also has some support for Namespaces. Mozilla is available with an XSLT implementation. Netscape: As of version 8, Netscape uses the Mozilla engine, and therefore it has the same XML / XSLT support as Mozilla. Opera: As of version 9, Opera has support for XML and XSLT (and CSS). Version 8 supports only XML + CSS. Internet Explorer: As of version 6, Internet Explorer supports XML, Namespaces, CSS, XSLT, and XPath. Version 5 is NOT compatible with the official W3C XSL Recommendation.</p>
38.	<p>What is XML presentation technique?</p> <p>XML presentation technologies provide a modular way to deliver and display content to a variety of devices. There are different presentation technologies used in XML to display the content. Eg: CSS</p>
39.	<p>List some of presentation technologies.</p> <p>Presentation technologies provide a modular way to deliver and display content to a variety of devices.</p> <p>i) CSS ii) XSL iii) XFORMS iv) XHTML</p>
40.	<p>Write about DOM.</p> <p>DOM is W3C supported standard application programming interface (API) that provides a platform and</p>

	language- neutral interface to allow developers to programmatically access and modify the content and structure documents.
41.	What is SAX? SAX is an example of a grass- roots development effort to provide a simple; Java based API for processing XML.
42.	What are the levels of DOM? DOM provides a platform and language- neutral interface to allow developers to programmatically access and modify the content and structure documents. It has Level 0, Level 1, Level 2, Level 3
43.	Compare CSS and XSL. CSS can be used with HTML. But XSL can't be used in HTML Both can be used in XML CSS is not a transformation language but XSL.
Part – B	
1.	<p>List and explain the XML syntax rules in detail. Explain how a XML document can be displayed on a browser. (APR/MAY 2011)</p> <ul style="list-style-type: none"> • All XML Elements Must Have a Closing Tag In HTML, some elements do not have to have a closing tag: In XML, it is illegal to omit the closing tag. All elements must have a closing tag: <p>This is a paragraph.</p>
 • XML Tags are Case Sensitive XML tags are case sensitive. The tag <Letter> is different from the tag <letter>. Opening and closing tags must be written with the same case: <Message>This is incorrect</message> <message>This is correct</message> Note: "Opening and closing tags" are often referred to as "Start and end tags". Use whatever you prefer. It is exactly the same thing. • XML Elements Must be Properly Nested In XML, all elements must be properly nested within each other: <i>This text is bold and italic</i> In the example above, "Properly nested" simply means that since the <i> element is opened inside the element, it must be closed inside the element. • XML Documents Must Have a Root Element XML documents must contain one element that is the parent of all other elements. This element is called the root element. <root> <child> <subchild>.....</subchild> </child> </root> • XML Attribute Values Must be Quoted XML elements can have attributes in name/value pairs just like in HTML. In XML, the attribute values must always be quoted. <note date="12/11/2007"> <to>Tove</to>

```
<from>Jani</from>
```

```
</note>
```

The error in the first document is that the date attribute in the note element is not quoted.

- **Entity References**

Some characters have a special meaning in XML.

If you place a character like "<" inside an XML element, it will generate an error because the parser interprets it as the start of a new element.

This will generate an XML error:

```
<message>if salary < 1000 then</message>
```

To avoid this error, replace the "<" character with an **entity reference**:

```
<message>if salary &lt; 1000 then</message>
```

There are 5 pre-defined entity references in XML:

<	<	less than
>	>	greater than
&	&	ampersand
'	'	apostrophe
"	"	quotation mark

Note: Only the characters "<" and "&" are strictly illegal in XML. The greater than character is legal, but it is a good habit to replace it.

- **Comments in XML**

The syntax for writing comments in XML is similar to that of HTML.

```
<!-- This is a comment -->
```

- **White-space is Preserved in XML**

XML does not truncate multiple white-spaces in a document (while HTML truncates multiple white-spaces to one single white-space):

```
XML: Hello Tove
```

```
HTML: Hello Tove
```

- **XML Stores New Line as LF**

Windows applications store a new line as: carriage return and line feed (CR+LF).

Unix and Mac OSX uses LF.

Old Mac systems uses CR.

XML stores a new line as LF.

- **Well Formed XML**

XML documents that conform to the syntax rules above are said to be "Well Formed" XML documents.

Displaying XML document on a browser

XML documents do not carry information about how to display the data. XML is a technology for describing the structure of data.

Since XML tags are "invented" by the author of the XML document, browsers do not know if a tag like <table> describes an HTML table or a dining table.

Without any information about how to display the data, most browsers will just display the XML document as it is.

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

Look at the XML file above in the browser: note.xml

An XML document will be displayed with color-coded root and child elements. A plus (+) or minus sign (-) to the left of the elements can be clicked to expand or collapse the element structure. To view the raw XML source (without the + and - signs), select "View Page Source" or "View Source" from the browser menu.

2. Explain the role of XML namespaces with examples. (MAY/JUNE 2012)

XML allows document authors to create custom elements. This extensibility can result in naming collisions (i.e. different elements that have the same name) among elements in an XML document.

An XML namespace is a collection of element and attribute names. Each namespace has a unique name that provides a means for document authors to unambiguously refer to elements with the same name (i.e. prevent collisions).

For example,

```
<subject>Geometry</subject>
```

and

```
<subject>Cardiology</subject>
```

use element subject to markup data. In the first case the subject is something one studies in school, whereas in the second case the subject is in the field of medicine. Namespaces can differentiate these two subject elements. For example,

```
<school:subject>Math</school:subject>
```

and

```
<medical:subject>Thrombosis</medical:subject>
```

Both school and medical are namespace prefixes. A document author prepends a namespace prefix to an element or attribute name to specify the namespace for that element or attribute. Each namespace prefix has a corresponding uniform resource identifier (URI) that uniquely identifies the namespace. A URI is simply a series of characters for differentiating names.

For example, the string urn:deitel:book could be a URI for a namespace that contains elements and attributes related to Deitel & Associates, Inc. publications. Document authors can create their own namespace prefixes using virtually any name, except the reserved namespace xml.

Differentiating Elements with namespaces:

Namespaces differentiate two distinct elements i.e., **file** element related to text file and **file** document related to an image file.

```
<?xml version="1.0"?>
<!-- namespace.xml -->
<text:directory
  xmlns:text="urn:deitel:textInfo"
  xmlns:image="urn:deitel:imageInfo">

  <text : file filename="book.xml">
  <text : description>A book list</text:description>
</text : file>

  <image : file filename="funny.jpg">
```

```
<image : description>A funny picture</image:description>
<image : size height="100" width="200"/>
</image : file>
</text : directory>
```

xmlns attribute: - to create namespaces prefixes. Eg. text and image. Each name space prefix is bound to a series of characters called a Uniform Resource Identifier (URI) that uniquely identifies the name space. A URI is a way to identifying a resource on the Internet. Two popular types of URI are Uniform Resource Name (URN) and Uniform Resource Locator (URL).

Another common practice is to use URL – specify the location of a file or a resource on the Internet.

```
<text:directory
  Xmlns : text="http://www.deitel.com/xmlns-text"
  Xmlns : image="http://deitel.com/xmlns-image">
```

Namespace prefix are required for elements such as file, description etc., Attributes do not require namespace prefix because each attribute is already part of an element the specifies the namespace prefix.

Specifying a Default Namespace:

To eliminate the need to place namespace prefixes in each element, document authors may specify a default namespace for an element and its children.

```
<?xml version="1.0"?>
<!--defaultnamespace.xml -->
```

```
<directory xmlns : image="urn:deitel:imagelInfo"
  xmlns="urn:deitel:textInfo">

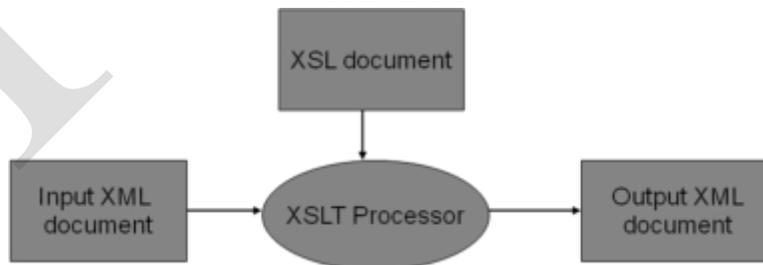
  <file filename="book.xml">
    <description>A book list</description>
  </file>

  <image : file filename="funny.jpg">
    <image:description>A funny picture</image:description>
    <image:size height="100" width="200"/>
  </image:file>

</directory>
```

3. Given an XSLT document and a source XML document explain the XSLT transformation process that produces a single result XML document. (NOV/DEC 2012)

- The Extensible Stylesheet Language (XSL) is an XML vocabulary typically used to transform XML documents from one form to another form



- JAXP allows a Java program to use the **Extensible Stylesheet Language (XSL)** to extract data from one XML document, process that data, and produce another XML document containing the processed data.

For example, XSL can be used to extract information from an XML document and embed it within an XHTML document so that the information can be viewed using a web browser.

TRANSFORMER:

Main.java:

```
import java.io.FileOutputStream;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.stream.StreamResult;
import javax.xml.transform.stream.StreamSource;
public class Main
{
    static String xml="D://WebTech//trans.XML";
    static String xslt="D://WebTech//trans.XSL";
    static String output="D://WebTech//trans.HTML";
    public static void main(String[] args)
    {
        try
        {
            TransformerFactory tf = TransformerFactory.newInstance();
            Transformer tr = tf.newTransformer(new StreamSource(xslt));
            tr.transform(new StreamSource(xml),new StreamResult(new FileOutputStream(output)));
            System.out.println("Output to " + output);
        }
        catch(Exception e)
        {
        }
    }
}
```

trans.xsl:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <html>
      <body>
        <h1>Indian Languages details</h1>
        <table border="1">
          <tr>
            <th>Language</th>
            <th>Family/Origin</th>
            <th>No. of speakers</th>
```

```

        <th>Region</th>
    </tr>
    <xsl:for-each select="language">
        <tr>
            <td><xsl:value-of select="name"/></td>
            <td><xsl:value-of select="family"/></td>
            <td><xsl:value-of select="users"/></td>
            <td><xsl:value-of select="region"/></td>
        </tr>
    </xsl:for-each>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>

```

trans.xml:

```

<?xml version="1.0"?>
<!--<?xml-stylesheet type="text/xsl" href="trans.xsl"?-->
<language>
<name>Kannada</name>
<region>Karnataka</region>
<users>38M</users>
<family>Dravidian</family>
</language>

```

trans.html:**Indian Languages details**

Language	Family/Origin	No. of speakers	Region
Kannada	Dravidian	38M	Karnataka

4. Write short notes on Event-oriented parsing (MAY/JUNE 2014)**SAX:**

- An alternative approach is to have the parser interact with an application as it reads an XML document. This is the approach taken by SAX (Simple API for XML).
- SAX allows an application to register event listeners with the XML parser.
- SAX parser calls these listeners as events occur and passes them the information about the events.

Main.Java:

```

import javax.xml.parsers.SAXParser;
import javax.xml.parsers.SAXParserFactory;
public class Main
{
    public static void main(String args[])
    {
        try

```

```
{
    SAXParserFactory factory = SAXParserFactory.newInstance();
    SAXParser saxParser = factory.newSAXParser();
    saxParser.parse("D://Staff1.XML",new CountHelper());
}
catch(Exception e)
{
    e.printStackTrace();
}
}
```

CountHelper.JAVA:

```
import org.xml.sax.Attributes;
import org.xml.sax.SAXException;
import org.xml.sax.helpers.DefaultHandler;
public class CountHelper extends DefaultHandler
{
    int no_elms;
    /*CountHelper()
    {
    super();
    }*/
    @Override
    public void startDocument() throws SAXException
    {
        no_elms=0;
        //return;
    }
    @Override
    public void startElement(String u,String ln,String qname,Attributes atts)
        throws SAXException
    {
        if(qname.equals("firstname"))
        {
            no_elms++;
        }
        // return;
    }
    @Override
    public void endDocument() throws SAXException
    {
        System.out.println("I/p Doc has " + no_elms + "firstname Elements");
    }
}
```

}

Staff1.xml:

```

<?xml version="1.0"?>
<company>
  <staff id="1001">
    <firstname>yong</firstname>
    <lastname>mook kim</lastname>
    <nickname>mkyong</nickname>
    <salary>100000</salary>
  </staff>
  <staff id="2001">
    <firstname>low</firstname>
    <lastname>yin fong</lastname>
    <nickname>fong fong</nickname>
    <salary>200000</salary>
  </staff>
</company>

```

OUTPUT:**DOM OUTPUT:**

RootSystem element :company
 Input Elements has:2nodes

SAX OUTPUT:

I/p Doc has 2 firstname Elements

5. **Explain the following: i) XML namespace ii) XML style sheet. iii) XML attributes iv) XML Schema**
i) XML Namespaces

XML Namespaces provide a method to avoid element name conflicts.

Name Conflicts

In XML, element names are defined by the developer. This often results in a conflict when trying to mix XML documents from different XML applications. This XML carries HTML table information:

```

<table>
  <tr>
    <td>Apples</td>
    <td>Bananas</td>
  </tr>
</table>

```

This XML carries information about a table (a piece of furniture):

```

<table>
  <name>African Coffee Table</name>
  <width>80</width>
  <length>120</length>
</table>

```

If these XML fragments were added together, there would be a name conflict. Both contain a

<table> element, but the elements have different content and meaning. A user or an XML application will not know how to handle these differences.

Solving the Name Conflict Using a Prefix

Name conflicts in XML can easily be avoided using a name prefix. This XML carries information about an HTML table, and a piece of furniture:

```
<h:table>
  <h:tr>
    <h:td>Apples</h:td>
    <h:td>Bananas</h:td>
  </h:tr>
</h:table>

<f:table>
  <f:name>African Coffee
  Table</f:name>
  <f:width>80</f:width>
  <f:length>120</f:length>
</f:table>
```

In the example above, there will be no conflict because the two <table> elements have different names.

XML Namespaces - The xmlns Attribute

When using prefixes in XML, a so-called **namespace** for the prefix must be defined. The namespace is defined by the **xmlns attribute** in the start tag of an element. The namespace declaration has the following syntax. xmlns:prefix="URI".

```

<root>

<h:table xmlns:h="http://www.w3.org/TR/html4/">
  <h:tr>
    <h:td>Apples</h:td>
    <h:td>Bananas</h:td>
  </h:tr>
</h:table>

<f:table xmlns:f="http://www.w3schools.com/furniture">
  <f:name>African Coffee Table</f:name>
  <f:width>80</f:width>
  <f:length>120</f:length>
</f:table>

</root>

```

In the example above, the xmlns attribute in the <table> tag give the h: and f: prefixes a qualified namespace. When a namespace is defined for an element, all child elements with the same prefix are associated with the same namespace. Namespaces can be declared in the elements where they are used or in the XML root element:

```

<root
xmlns:h="http://www.w3.org/TR/html4/"
xmlns:f="http://www.w3schools.com/furnitur
e">
  <h:table>
    <h:tr>
      <h:td>Apples</h:td>
      <h:td>Bananas</h:td>
    </h:tr>
  </h:table>

  <f:table>
    <f:name>African Coffee Table</f:name>
    <f:width>80</f:width>
    <f:length>120</f:length>
  </f:table>
</root>

```

Uniform Resource Identifier (URI)

A **Uniform Resource Identifier** (URI) is a string of characters which identifies an Internet Resource. The most common URI is the **Uniform Resource Locator** (URL) which identifies an Internet domain address. Another, not so common type of URI is the **Universal Resource Name** (URN). In our examples we will only use URLs.

Default Namespaces

Defining a default namespace for an element saves us from using prefixes in all the child elements. It has the following syntax:

```
xmlns="namespaceURI"
```

This XML carries HTML table information:

```
<table xmlns="http://www.w3.org/TR/html4/">
  <tr>
    <td>Apples</td>
    <td>Bananas</td>
  </tr>
</table>
```

This XML carries information about a piece of furniture:

```
<table
xmlns="http://www.w3schools.com/furniture">
  <name>African Coffee Table</name>
  <width>80</width>
  <length>120</length>
</table>
```

Namespaces in Real Use

XSLT is an XML language that can be used to transform XML documents into other formats, like HTML. In the XSLT document below, you can see that most of the tags are HTML tags. The tags that are not HTML tags have the prefix `xsl`, identified by the namespace

```
xmlns:xsl="http://www.w3.org/1999/XSL/Transform":
<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
<html>
<body>
  <h2>My CD Collection</h2>
```

```

<table border="1">
  <tr>
    <th style="text-align:left">Title</th>
    <th style="text-align:left">Artist</th>
  </tr>
  <xsl:for-each select="catalog/cd">
    <tr>
      <td><xsl:value-of select="title"/></td>
      <td><xsl:value-of select="artist"/></td>
    </tr>
  </xsl:for-each>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>

```

ii) XML Stylesheet

Displaying XML with XSLT

XSLT (eXtensible Stylesheet Language Transformations) is the recommended style sheet language for XML.

XSLT is far more sophisticated than CSS. With XSLT you can add/remove elements and attributes to or from the output file. You can also rearrange and sort elements, perform tests and make decisions about which elements to hide and display, and a lot more.

XSLT uses XPath to find information in an XML document.

XSLT Example

We will use the following XML document:

```

<?xml version="1.0" encoding="UTF-8"?>
<breakfast_menu>

```

```

  <food>
    <name>Belgian Waffles</name>
    <price>$5.95</price>
    <description>Two of our famous Belgian Waffles with plenty of real maple syrup</description>
    <calories>650</calories>
  </food>

```

```

  <food>
    <name>Strawberry Belgian Waffles</name>
    <price>$7.95</price>
    <description>Light Belgian waffles covered with strawberries and whipped cream</description>
    <calories>900</calories>
  </food>

```

```

  <food>
    <name>Berry-Berry Belgian Waffles</name>

```

```

<price>$8.95</price>
<description>Light Belgian waffles covered with an assortment of fresh berries and whipped
cream</description>
<calories>900</calories>
</food>

<food>
<name>French Toast</name>
<price>$4.50</price>
<description>Thick slices made from our homemade sourdough bread</description>
<calories>600</calories>
</food>

<food>
<name>Homestyle Breakfast</name>
<price>$6.95</price>
<description>Two eggs, bacon or sausage, toast, and our ever-popular hash browns</description>
<calories>950</calories>
</food>

</breakfast_menu>

```

Use XSLT to transform XML into HTML, before it is displayed in a browser:

Example XSLT Stylesheet:

```

<?xml version="1.0" encoding="UTF-8"?>
<html xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<body style="font-family:Arial;font-size:12pt;background-color:#EEEEEE">
<xsl:for-each select="breakfast_menu/food">
  <div style="background-color:teal;color:white;padding:4px">
    <span style="font-weight:bold"><xsl:value-of select="name"/> - </span>
    <xsl:value-of select="price"/>
  </div>
  <div style="margin-left:20px;margin-bottom:1em;font-size:10pt">
    <p>
      <xsl:value-of select="description"/>
      <span style="font-style:italic"> (<xsl:value-of select="calories"/> calories per serving)</span>
    </p>
  </div>
</xsl:for-each>
</body>
</html>

```

iii) XML Attributes:

XML elements can have attributes, just like HTML. Attributes provide additional information about an element.

XML Attributes

In HTML, attributes provide additional information about elements:

```

```

```
<a href="demo.asp">
```

Attributes often provide information that is not a part of the data. In the example below, the file type is irrelevant to the data, but can be important to the software that wants to manipulate the element:

```
<file type="gif">computer.gif</file>
```

- **XML Attributes Must be Quoted**

Attribute values must always be quoted. Either single or double quotes can be used. For a person's gender, the person element can be written like this:

```
<person gender="female"> or like this:
```

```
<person gender='female'>
```

If the attribute value itself contains double quotes you can use single quotes, like in this example:

```
<gangster name='George "Shotgun" Ziegler'> or you can use character entities:
```

```
<gangster name="George &quot;Shotgun&quot; Ziegler">
```

- **XML Elements vs. Attributes**

Take a look at these examples:

```
<person gender="female">
  <firstname>Anna</firstname>
  <lastname>Smith</lastname>
</person>
```

```
<person>
  <gender>female</gender>
  <firstname>Anna</firstname>
  <lastname>Smith</lastname>
</person>
```

In the first example gender is an attribute. In the last, gender is an element. Both examples provide the same information. There are no rules about when to use attributes or when to use elements.

- **Avoid XML Attributes?**

Some of the problems with using attributes are:

- attributes cannot contain multiple values (elements can)
- attributes cannot contain tree structures (elements can)
- attributes are not easily expandable (for future changes)

Attributes are difficult to read and maintain. Use elements for data. Use attributes for information that is not relevant to the data.

- **XML Attributes for Metadata**

Sometimes ID references are assigned to elements. These IDs can be used to identify XML elements in much the same way as the id attribute in HTML. This example demonstrates this:

```
<messages>
  <note id="501">
    <to>Tove</to>
    <from>Jani</from>
    <heading>Reminder</heading>
    <body>Don't forget me this weekend!</body>
  </note>
  <note id="502">
    <to>Jani</to>
    <from>Tove</from>
    <heading>Re: Reminder</heading>
    <body>I will not</body>
  </note>
</messages>
```

The id attributes above are for identifying the different notes. It is not a part of the note itself. The metadata (data about data) should be stored as attributes, and the data itself should be stored as elements.

iv) XML Schema

- An XML Schema describes the structure of an XML document, just like a DTD.
- An XML document with correct syntax is called "Well Formed".
- An XML document validated against an XML Schema is both "Well Formed" and "Valid".

The purpose of an XML Schema is to define the legal building blocks of an XML document, just like a DTD.

An XML Schema:

- defines elements that can appear in a document
- defines attributes that can appear in a document
- defines which elements are child elements
- defines the order of child elements
- defines the number of child elements
- defines whether an element is empty or can include text
- defines data types for elements and attributes
- defines default and fixed values for elements and attributes

XML Schema Example

```

<?xml version="1.0"?>
<xs:schema
xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="note">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="to" type="xs:string"/>
      <xs:element name="from" type="xs:string"/>
      <xs:element name="heading" type="xs:string"/>
      <xs:element name="body" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

</xs:schema>

```

The Schema above is interpreted like this:

- <xs:element name="note"> defines the element called "note"
 - <xs:complexType> the "note" element is a complex type
 - <xs:sequence> the complex type is a sequence of elements
 - <xs:element name="to" type="xs:string"> the element "to" is of type string (text)
 - <xs:element name="from" type="xs:string"> the element "from" is of type string
 - <xs:element name="heading" type="xs:string"> the element "heading" is of type string
 - <xs:element name="body" type="xs:string"> the element "body" is of type string
- Everything is wrapped in "Well Formed" XML.

6. Explain XSL with suitable example

XSL is a language for expressing style sheets. An XSL style sheet is, like with CSS, a file that describes how to display an XML document of a given type. XSL shares the functionality and is compatible with CSS2 (although it uses a different syntax). It also adds:

- A transformation language for XML documents: **XSLT**. Originally intended to perform complex styling operations, like the generation of tables of contents and indexes, it is now used as a general purpose XML processing language. XSLT is thus widely used for purposes other than XSL, like generating HTML web pages from XML data.
- Advanced styling features, expressed by an XML document type which defines a set of elements called **Formatting Objects**, and attributes (in part borrowed from CSS2 properties and adding more complex ones).

How Does It Work?

Styling requires a source XML documents, containing the information that the style sheet will display and the style sheet itself which describes how to display a document of a given type.

Example:

In this XSL example, two lists in XML and use the XSL style sheet to set the style display the way I want to be created. Also HTML is embedded in the style sheet which allows an actual ordered list to be created.

The XML File

```
<?xml version="1.0"?>
<!DOCTYPE LANGLIST SYSTEM "langlist.dtd">
<?xml-stylesheet type="text/xsl" href="xmlstyle.xsl"?>
<LANGLIST>
  <TITLE>List of Items Important to Markup Languages</TITLE>
  <TITLE1>Languages</TITLE1>
  <LIST1>
    <LANGUAGES>SGML</LANGUAGES>
    <LANGUAGES>XML</LANGUAGES>
    <LANGUAGES>HTML</LANGUAGES>
  </LIST1>
  <TITLE2>Other Support</TITLE2>
  <LIST2>
    <OTHER>DTD</OTHER>
    <OTHER>DSSSL</OTHER>
    <OTHER>Style Sheets</OTHER>
  </LIST2>
</LANGLIST>
```

The DTD File

```
<!ELEMENT LANGLIST ANY>
<!ENTITY % Shape "(rect|circle|poly|default)">
<!ELEMENT TITLE (#PCDATA)>
<!ELEMENT TITLE1 (#PCDATA)>
<!ELEMENT TITLE2 (#PCDATA)>
<!ELEMENT LIST1 (LANGUAGES)+>
<!ELEMENT LIST2 (OTHER)+>
<!ELEMENT LANGUAGES (#PCDATA)>
<!ATTLIST LANGUAGES
  type %Shape; #IMPLIED>
<!ELEMENT OTHER (#PCDATA)>
<!ATTLIST OTHER
  type (disc|square|circle) #IMPLIED>
```

This DTD file employs an ENTITY, just for demonstration purposes. An entity, in this case is similar to a variable with a value that is set to a string value. In this case the entity name is "Shape" and the string value is "(rect|circle|poly|default)". This value is used to set the attribute list (ATTLIST) for the LANGUAGES and OTHER elements, although these attributes are not used in this example.

The XML Style File

```
<?xml version="1.0"?>

<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl">

  <xsl:template match="/">
```

```

        <xsl:apply-templates select="LANGLIST/TITLE" />
        <xsl:apply-templates select="LANGLIST/TITLE1" />
        <xsl:apply-templates select="LANGLIST/LIST1" />
        <xsl:apply-templates select="LANGLIST/TITLE2" />
        <xsl:apply-templates select="LANGLIST/LIST2" />
    </xsl:template>

    <xsl:template match="TITLE">
        <SPAN STYLE="display: 'block'; font-family: 'arial';
color: '#008000'; font-weight: '600'; font-size: '22'; margin-top:
'12pt'; text-align: 'center'">
            <xsl:value-of />
        </SPAN>
        <BR/>
    </xsl:template>

    <xsl:template match="TITLE1">
        <SPAN STYLE="display: 'block'; font-family: 'arial';
color: '#000080'; font-weight: '400'; font-size: '20'; margin-top:
'12pt'">
            <xsl:value-of />
        </SPAN>
        <BR/>
    </xsl:template>

    <xsl:template match="LIST1">
        <UL style="display: 'list-item'; list-style-image:
url('bullet8.gif'); font-family: 'arial'; color: '#000000'; font-
weight: '400'; margin-left: '15pt'; margin-top: '12pt'; font-size:
'18'">
            <xsl:for-each select="LANGUAGES">
                <LI style="display: 'list-item'; list-style-type:
'square'; list-style-image: url('bullet8.gif'); font-family: 'arial';
color: '#ff0000'; font-weight: '300'; margin-left: '15pt'; margin-
top: '12pt'; font-size: '16'">
                    <xsl:value-of />
                </LI>
            </xsl:for-each>
        </UL>
    </xsl:template>

    <xsl:template match="TITLE2">
        <SPAN STYLE="display: 'block'; font-family: 'arial';
color: '#000080'; font-weight: '400'; font-size: '20'; margin-top:
'12pt'">
            <xsl:value-of />
        </SPAN>

```

	<pre>
 </xsl:template> <xsl:template match="LIST2"> <UL style="display: 'list-item'; list-style-image: url('bullet8.gif'); font-family: 'arial'; color: '#000000'; font- weight: '400'; margin-left: '15pt'; margin-top: '12pt'; font-size: '18'"> <xsl:for-each select="OTHER"> <LI style="display: 'list-item'; list-style-type: 'square'; list-style-image: url('bullet8.gif'); font-family: 'arial'; color: '#0000ff'; font-weight: '200'; margin-left: '15pt'; margin- top: '12pt'; font-size: '14'"> <xsl:value-of "."/> </xsl:for-each> </xsl:template> </xsl:stylesheet> </pre> <p>Between the <code><xsl:template match="/"></code> tag and <code></xsl:template></code> tag, the order of the "apply-templates" statements is very important since this determines the order they are displayed in. Also note that the text <code><xsl:value-of "."/></code> is used to refer to the XML object currently being processed in order to display the value of that object.</p>
7.	<p>Explain the architectural revolution of XML.</p> <p><u>XML: The Three Revolutions</u></p> <p>Three areas of impact are i) data, which XML frees from the confines of fixed, ii) program-dependent formats; architecture, iii) with a change in emphasis from tightly coupled distributed systems to a more loosely coupled confederation based on the Web; and software, with the realization that software evolution is a better path to managing complexity than building monolithic applications.</p>

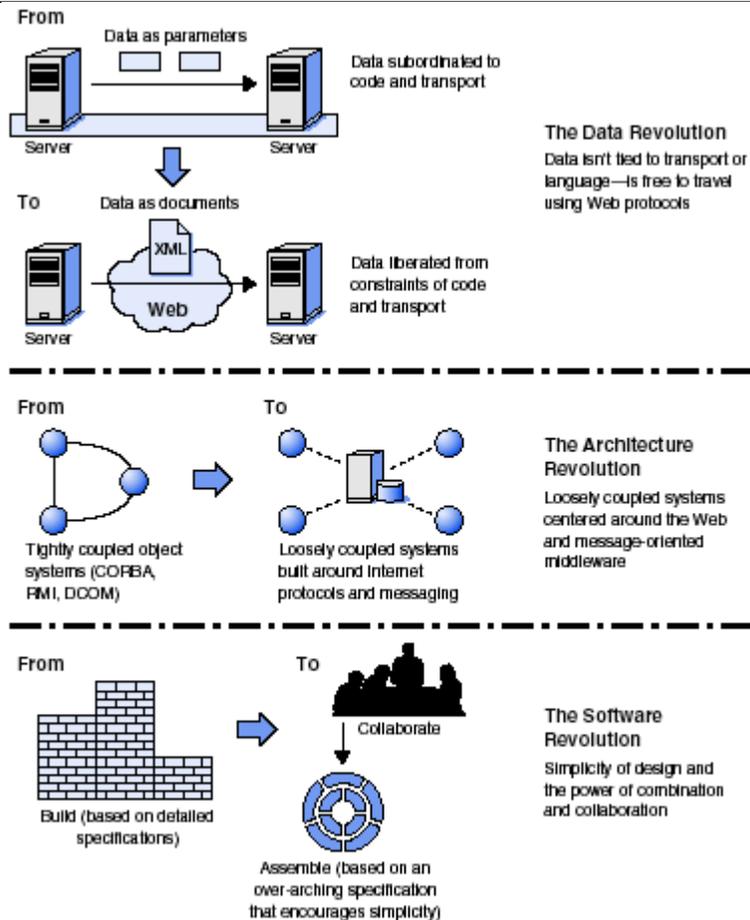


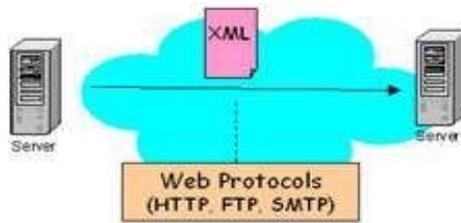
Figure: The three XML revolutions: data, architecture, and software.

The Data Revolution

Prior to XML, data was very much proprietary, closely associated with applications that understood how data was formatted and how to process it. Now, XML-based industry-specific data vocabularies provide alternatives to specialized Electronic Data Interchange (EDI) solutions by facilitating B2B data exchange and playing a key role as a messaging infrastructure for distributed computing.

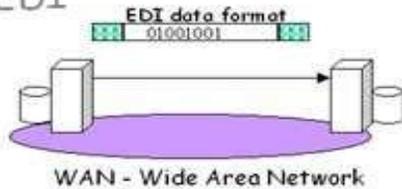
XML's strength is its data independence. XML is pure data description, not tied to any programming language, operating system, or transport protocol. In the grand scheme of distributed computing this is a radical idea. The implication is that we don't require lock-in to programmatic infrastructures to make data available to Web-connected platforms. In effect, data is free to move about globally without the constraints imposed by tightly coupled transport-dependent architectures. XML's sole focus on data means that a variety of transport technologies may be used to move XML across the Web. As a result, protocols such as HTTP have had a tremendous impact on XML's viability and have opened the door to alternatives to CORBA, RMI, and DCOM, which don't work over TCP/IP. XML does this by focusing on data and leaving other issues to supporting technologies.

The Data Revolution



Data is free to move about the Web
- not dependent on programming language or transport protocol

EDI



- > data formats and messages defined by EDI standards
- > applications run in batch mode outside the internet
- > proprietary wide area network (WAN) required to deliver EDI messages

CORBA, RMI, DCOM



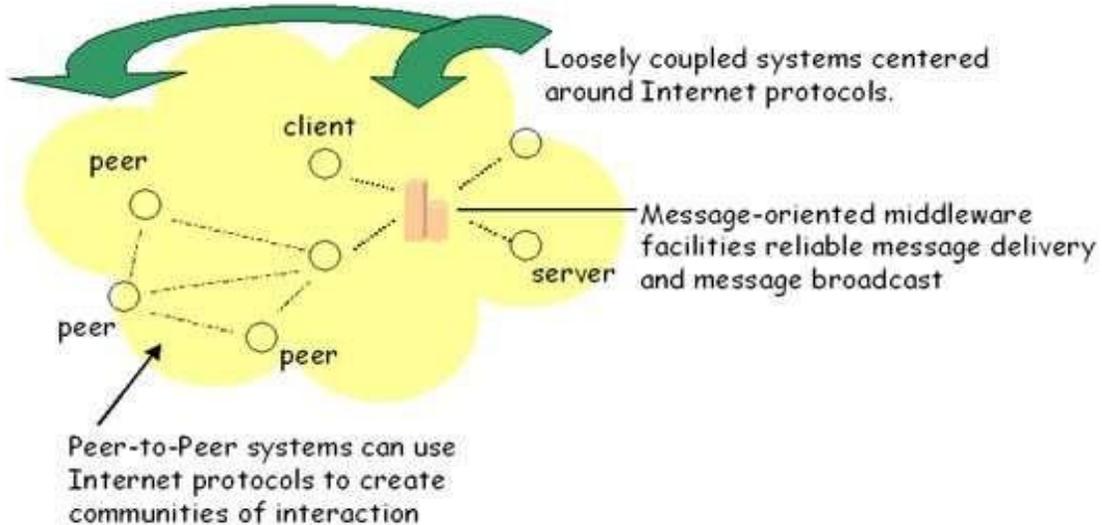
- > data passed as parameters to method calls of an object-oriented language
- > platforms require code to interface with ORB

The Data Revolution and The Architectural Revolution

Together these XML-based technology initiatives open up new possibilities for distributed computing that leverage the existing infrastructure of the Web and create a transition from object-based distributed systems to architectures based on Web services that can be discovered, accessed, and assembled using open Web technologies. The focal point of this change in architectural thinking has been a move from tightly coupled systems based on established infrastructures such as CORBA, RMI, and DCOM, each with their own transport protocol, to loosely coupled systems riding atop standard Web protocols such as TCP/IP. Although the transport protocols underlying CORBA, RMI, and DCOM provide for efficient communication between nodes, their drawback is their inability to communicate with other tightly coupled systems or directly with the Web.

Loosely coupled Web-based systems, on the other hand, provide what has long been considered the Holy Grail of computing: universal connectivity. Using TCP/IP as the transport, systems can establish connections with each other using common open-Web protocols. Although it is possible to build software bridges linking tightly coupled systems with each other and the Web, such efforts are not trivial and add another layer of complexity on top of an already complex infrastructure.

The Architecture Revolution



The Architecture Revolution and The Software Revolution

XML is also part of a revolution in how we build software. During the 1970s and 1980s, software was constructed as monolithic applications built to solve specific problems. The problem with large software projects is that, by trying to tackle multiple problems at once, the software is often ill-suited to adding new functionality and adapting to technological change. In the 1990s a different model for software emerged based on the concept of simplicity.

8. Write a program using PHP that creates the web application for result publication

```
<?php
$username = "your_name";
$password = "your_password";
$hostname = "localhost";

//connection to the database
$dbhandle = mysql_connect($hostname, $username, $password)
or die("Unable to connect to MySQL");
echo "Connected to MySQL<br>";

//select a database to work with
$dbselected = mysql_select_db("results",$dbhandle)
or die("Could not select examples");

//execute the SQL query and return records
$result = mysql_query("SELECT regno, sub1,sub2 FROM res_tab");
```

```
//fetch the data from the database
while ($row = mysql_fetch_array($result)) {
    echo "REGNO:". $row{'regno'}." SUB1:". $row{'sub1'}."SUB2: ". //display the results
    $row{'sub2'}."<br>";
}
//close the connection
mysql_close($dbhandle);
?>
```

To create 'results' database on your MySQL server you should run the following script:

```
CREATE DATABASE `results`;
USE `results`;
CREATE TABLE `res_tab` (
    `regno` int UNIQUE NOT NULL,
    `sub1` varchar(2),
    `sub2` varchar(2),
    PRIMARY KEY(id)
);
INSERT INTO cars VALUES(1,'A','B');
INSERT INTO cars VALUES(2,'C','D');
INSERT INTO cars VALUES(3,'Si','A');
```

9. a) Design simple calculator using PHP.

HTML TextBox value, in other words operand value is accepted in this way:

```
$_REQUEST['name']
```

In that way you can simply store the TextBox value or drop down list value for calculation in a PHP variable.

```
<html>
<head>
<title>Simple PHP Calculator</title>
</head>

<body>
<form method='post' action='calculator.php'>

<input type='text' name='value1'>
<input type='text' name='value2'> <select name='action'>
<option>+</option>
<option>-</option>
<option>*</option>
<option>/</option>
</select>
<input type='submit' name='submit' value='Calculate Now'></form>

<?php
if(isset($_POST['submit']))
{ $value1 = $_POST['value1'];
  $value2 = $_POST['value2'];
  $action = $_POST['action'];
```

```

if($action==""){
echo "<b>Your Answer is:</b><br>";
echo $value1+$value2;
}

if($action=="-"){
echo "<b>Your Answer is:</b><br>";
echo $value1-$value2;
}

if($action=="*"){
echo "<b>Your Answer is:</b><br>";
echo $value1*$value2;
}

if($action=="/"){
echo "<b>Your Answer is:</b><br>";
echo $value1/$value2;
}
}
?>

```

</body>

</html>

Output:

Your Answer is:

200

b) Design application to send a email using PHP**Description of mail function:**

```

bool mail ( string $to , string $subject , string $message [, string $additional_headers [, string
$additional_parameters ]] )

```

<?php

//if "email" variable is filled out, send email

if (isset(\$_REQUEST['email'])) {

//Email information

\$admin_email = "someone@example.com";

\$email = \$_REQUEST['email'];

\$subject = \$_REQUEST['subject'];

\$comment = \$_REQUEST['comment'];

//send email

mail(\$email, "\$subject", \$comment, "From:" . \$admin_email);

```
//Email response
echo "Thank you for contacting us!";
}

//if "email" variable is not filled out, display the form
else {
?>

<form method="post">
Email: <input name="email" type="text" /><br />
Subject: <input name="subject" type="text" /><br />
Message:<br />
<textarea name="comment" rows="15" cols="40"></textarea><br />
<input type="submit" value="Submit" />
</form>

<?php
}
?>
```

Output:**10. Develop a shopping cart application using PHP with use of cookies.**

Use of each files and folders in the shopping cart application.

1. **config** is a folder that contains another file called **db_connect.php** which is the file used to connect to the MySQL database for retrieving product information.
2. **add_to_cart.php** is executed when the user clicks on the “add to cart” button. It processes the product data by adding it to the cookie JSON string to make it a cart item. Each cart item is processed and converted to a JSON string to be stored in the cookie.
3. **cart.php** is where the cart items are read and displayed to the user. We retrieve the JSON string from a cookie variable, convert it to an associated array and loop through it to be displayed on an HTML table.
4. **layout_foot.php** is used as a closing HTML wrapper, and is always included the the end of **cart.php** and **products.php** because it closes the tags started by **layout_head.php**, it also contains any JavaScript source and jQuery codes used in this script.
5. **layout_head.php** is used as an opening HTML page wrapper. It is always included at the beginning of **cart.php** and **products.php** because it contains the opening HTML tag, head tags, title tags and any CSS resource and scripts used.

6. navigation.php contains the links to products.php and cart.php pages that the user can click. The cart item count is also shown here, beside the “Cart” link.

7. products.php displays all the products retrieve from the database. We are using PDO extension to connect and retrieve data from the MySQL database.

8. remove_from_cart.php is executed when the user clicks on the “Remove from cart” button. It removes the product item from the JSON string saved in the cookie variable.

Important note: This source code focuses on using cookies for storing shopping cart items, the “checkout” is not in its scope.

Step 1: create a database and run the following SQL queries to create the sample tables.

```
CREATE TABLE IF NOT EXISTS `products` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `name` varchar(32) NOT NULL,  
  `price` int(11) NOT NULL,  
  PRIMARY KEY (`id`)  
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=4 ;
```

```
--  
-- Dumping data for table `products`  
--
```

```
INSERT INTO `products` (`id`, `name`, `price`) VALUES  
(1, 'LG P880 4X HD', 336),  
(2, 'Google Nexus 4', 299),  
(3, 'Samsung Galaxy S4', 600);
```

Step 2: In your chosen root directory, create a folder named config.

Step 3: Inside that config folder, create a file named db_connect.php, and put the following code inside it, just change to database credentials to your own.

```
<?php  
$host = "your_host";  
$db_name = "your_database_name";  
$username = "your_database_username";  
$password = "your_database_password";  
try {  
    $con = new PDO("mysql:host={$host};dbname={$db_name}",  
$username, $password);  
}  
  
//to handle connection error  
catch(PDOException $exception){  
    echo "Connection error: " . $exception->getMessage();  
}  
?>
```

Step 4: Create a file called products.php, we will retrieve the products using the code below.

```
<?php  
$page_title="Products";
```

```

include 'layout_head.php';

// to prevent undefined index notice
$action = isset($_GET['action']) ? $_GET['action'] : "";
$name = isset($_GET['name']) ? $_GET['name'] : "";

// show messages based on given action
if($action=='added'){
    echo "<div class='alert alert-info'>";
        echo "<strong>{$name}</strong> was added to your cart!";
    echo "</div>";
}

else if($action=='exists'){
    echo "<div class='alert alert-info'>";
        echo "<strong>{$name}</strong> already exists in your
cart!";
    echo "</div>";
}

// query the products
$query = "SELECT id, name, price FROM products ORDER BY name";
$stmt = $con->prepare( $query );
$stmt->execute();

$num = $stmt->rowCount();

if($num>0){
    //start table
    echo "<table class='table table-hover table-responsive table-
bordered'>";

        // our table heading
        echo "<tr>";
            echo "<th class='textAlignLeft'>Product Name</th>";
            echo "<th>Price (USD)</th>";
            echo "<th>Action</th>";
        echo "</tr>";

        while ($row = $stmt->fetch(PDO::FETCH_ASSOC)) {
            extract($row);

            //creating new table row per record
            echo "<tr>";
                echo "<td>{$name}</td>";
                echo "<td>&#36;{$price}</td>";
                echo "<td>";

```

```

        echo "<a
href='add_to_cart.php?id={$id}&name={$name}' class='btn btn-
primary'>";
        echo "<span class='glyphicon
glyphicon-shopping-cart'></span> Add to cart";
        echo "</a>";
        echo "</td>";
    echo "</tr>";
}
echo "</table>";
}
// tell the user if there's no products in the database
else{
    echo "No products found.";
}
include 'layout_foot.php';
?>

```

Step 5: products.php on step 4 above will not actually work without the layout_head.php and layout_foot.php, so first, we'll create the layout_head.php with the following code:

```

<?php
// connect to database
include 'config/db_connect.php';
?>
<!DOCTYPE html>
<html lang="en">
<head>

    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1">

    <title><?php echo isset($page_title) ? $page_title : "The Code of
a Ninja"; ?> - LIVE DEMO</title>

    <!-- Bootstrap -->
    <!-- Latest compiled and minified CSS -->
    <link rel="stylesheet"
href="//netdna.bootstrapcdn.com/bootstrap/3.1.1/css/bootstrap.min.css"
>

    <!-- HTML5 Shiv and Respond.js IE8 support of HTML5 elements and
media queries -->

```

```

    <!-- WARNING: Respond.js doesn't work if you view the page via
file:// -->
    <!--[if lt IE 9]>
    <script
src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></scrip
t>
    <script
src="https://oss.maxcdn.com/libs/respond.js/1.4.2/respond.min.js"></sc
ript>
    <![endif]-->

</head>
<body>

    <?php include 'navigation.php'; ?>

    <!-- container -->
    <div class="container">

        <div class="page-header">
            <h1><?php echo isset($page_title) ? $page_title : "The
Code of a Ninja"; ?></h1>
        </div>

```

Step 6: layout_head.php includes another PHP file called navigation.php, so we'll create it and put the following code.

```

<!-- navbar -->
<div class="navbar navbar-default navbar-static-top"
role="navigation">
    <div class="container">

        <div class="navbar-header">
            <button type="button" class="navbar-toggle" data-
toggle="collapse" data-target=".navbar-collapse">
                <span class="sr-only">Toggle navigation</span>
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
            </button>
            <a class="navbar-brand" href="products.php">Your
Site</a>
        </div>

        <div class="navbar-collapse collapse">
            <ul class="nav navbar-nav">
                <li <?php echo $page_title=="Products" ?

```

```

"class='active'" : ""; ?> >
        <a href="products.php">Products</a>
    </li>
    <li <?php echo $page_title=="Cart" ?
"class='active'" : ""; ?> >
        <a href="cart.php">
            <?php
                // count products in cart
                $cookie =
$_COOKIE['cart_items_cookie'];
                $cookie = stripslashes($cookie);
                $saved_cart_items =
json_decode($cookie, true);

                $cart_count=count($saved_cart_items);
                ?>
                Cart <span class="badge"
id="comparison-count"><?php echo $cart_count; ?></span>
            </a>
        </li>
    </ul>
</div><!--/.nav-collapse -->
</div>
</div>
<!-- /navbar -->
Step 7: Now we'll create the layout_foot.php
    </div>
    <!-- /container -->

<!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.0/jquery.min.js
"></script>

<!-- Include all compiled plugins (below), or include individual files
as needed -->
<!-- Latest compiled and minified JavaScript -->
<script
src="//netdna.bootstrapcdn.com/bootstrap/3.1.1/js/bootstrap.min.js"></
script>

</body>
</html>

```

Step 8: products.php has links to the add_to_cart.php file, we'll create that file and put the code below.

```

<?php
// initialize empty cart items array
$cart_items=array();

// get the product id and name
$id = isset($_GET['id']) ? $_GET['id'] : "";
$name = isset($_GET['name']) ? $_GET['name'] : "";

// add new item on array
$cart_items[$id]=$name;

// read the cookie
$cookie = $_COOKIE['cart_items_cookie'];
$cookie = stripslashes($cookie);
$saved_cart_items = json_decode($cookie, true);

// if $saved_cart_items is null, prevent null error
if(!$saved_cart_items){
    $saved_cart_items=array();
}

// check if the item is in the array, if it is, do not add
if(array_key_exists($id, $saved_cart_items)){
    // redirect to product list and tell the user it was already
    added to the cart
    header('Location: products.php?action=exists&id' . $id .
'&name=' . $name);
}
else{
    // if cart has contents
    if(count($saved_cart_items)>0){
        foreach($saved_cart_items as $key=>$value){
            // add old item to array, it will prevent duplicate
            $cart_items[$key]=$value;
        }
    }

    // put item to cookie
    $json = json_encode($cart_items, true);
    setcookie('cart_items_cookie', $json);

    // redirect
    header('Location: products.php?action=added&id=' . $id .
'&name=' . $name);
}

```

```
?>
```

Step 9: Now if the products were able to be added on the cart, we'll have to view it using cart.php, we'll create that file with the following codes.

```
<?php
$page_title="Cart";
include 'layout_head.php';

$action = isset($_GET['action']) ? $_GET['action'] : "";
$name = isset($_GET['name']) ? $_GET['name'] : "";

if($action=='removed'){
    echo "<div class='alert alert-info'>";
        echo "<strong>{$name}</strong> was removed from your
cart!";
    echo "</div>";
}

$cookie = $_COOKIE['cart_items_cookie'];
$cookie = stripslashes($cookie);
$saved_cart_items = json_decode($cookie, true);

if(count($saved_cart_items)>0){
    // get the product ids
    $sids = "";
    foreach($saved_cart_items as $id=>$name){
        $sids = $sids . $id . ",";
    }

    // remove the last comma
    $sids = rtrim($sids, ',');

    //start table
    echo "<table class='table table-hover table-responsive table-
bordered'>";

    // our table heading
    echo "<tr>";
        echo "<th class='textAlignLeft'>Product Name</th>";
        echo "<th>Price (USD)</th>";
            echo "<th>Action</th>";
        echo "</tr>";

        $query = "SELECT id, name, price FROM products WHERE id
IN ({$sids}) ORDER BY name";
        $stmt = $con->prepare( $query );
```

```

$stmt->execute();

$total_price=0;
while ($row = $stmt->fetch(PDO::FETCH_ASSOC)) {
    extract($row);

    echo "<tr>";
        echo "<td>{$name}</td>";
        echo "<td>&#36;{$price}</td>";
        echo "<td>";
            echo "<a
href='remove_from_cart.php?id={$id}&name={$name}' class='btn btn-
danger'>";
                echo "<span class='glyphicon
glyphicon-remove'></span> Remove from cart";
                echo "</a>";
            echo "</td>";
        echo "</tr>";

        $total_price+=$price;
    }
    echo "<tr>";
        echo "<td><b>Total</b></td>";
        echo "<td>&#36;{$total_price}</td>";
        echo "<td>";
            echo "<a href='#' class='btn btn-
success'>";
                echo "<span class='glyphicon
glyphicon-shopping-cart'></span> Checkout";
                echo "</a>";
            echo "</td>";
        echo "</tr>";
    echo "</table>";
}
else{
    echo "<div class='alert alert-danger'>";
        echo "<strong>No products found</strong> in your cart!";
    echo "</div>";
}

include 'layout_foot.php';
?>

```

Step 10: cart.php links to a file called remove_from_cart.php, to remove an item from the cart.

We'll create `remove_from_cart.php` with the codes below.

```
<?php
// get the product id
$id = isset($_GET['id']) ? $_GET['id'] : "";
$name = isset($_GET['name']) ? $_GET['name'] : "";

// read
$cookie = $_COOKIE['cart_items_cookie'];
$cookie = stripslashes($cookie);
$saved_cart_items = json_decode($cookie, true);

// remove the item from the array
$saved_cart_items = array_diff($saved_cart_items, array($id=>$name));

// delete cookie value
setcookie("cart_items_cookie", "", time()-3600);

// enter new value
$json = json_encode($saved_cart_items, true);
setcookie('cart_items_cookie', $json);

// redirect to product list and tell the user it was added to cart
header('Location: cart.php?action=removed&id=' . $id . '&name=' .
$name);

?>
```

Enter or Update Cart Item Quantity

But what if your users want to enter or update the quantity of items in the cart? It is possible using cookies.

1 Add a new column named “Quantity” in **products.php** and **cart.php**. The word “Quantity” will be the column header and for the rest of the table row, it will be input or text boxes where the user can enter the product quantity before clicking the “Add to cart” or “Update cart” button.

2 On **products.php**, the “Add to cart” button will not be a direct link to `add_to_cart.php` file. It will be a button that will execute a jQuery code because it has to get the quantity entered by the user.

3 On **cart.php**, there will be an “Update cart” button beside the quantity textbox. Clicking it will run a jQuery script that gets the new quantity entered by the user and saves the changes with the help of a new file called “`update_quantity.php`”.

11. Explain about the control statements in PHP with example.

PHP Conditional Statements

In PHP we have the following conditional statements:

- **if statement** - executes some code only if a specified condition is true
- **if...else statement** - executes some code if a condition is true and another code if the condition is false
- **if...elseif...else statement** - specifies a new condition to test, if the first condition is false

- **switch statement** - selects one of many blocks of code to be executed

The if Statement

The if statement is used to execute some code **only if a specified condition is true.**

Syntax

```
if (condition) {  
    code to be executed if condition is true;  
}
```

The example below will output "Have a good day!" if the current time (HOUR) is less than 20:

Example

```
<?php  
$t = date("H");  
if ($t < "20") {  
    echo "Have a good day!";  
}  
?>
```

The if...else Statement

Use the if...else statement to execute some code **if a condition is true and another code if the condition is false.**

Syntax

```
if (condition) {  
    code to be executed if condition is true;  
} else {  
    code to be executed if condition is false;  
}
```

The example below will output "Have a good day!" if the current time is less than 20, and "Have a good night!" otherwise:

Example

```
<?php  
$t = date("H");  
  
if ($t < "20") {  
    echo "Have a good day!";  
} else {  
    echo "Have a good night!";  
}  
?>
```

The if...elseif...else Statement

Use the if...elseif...else statement to **specify a new condition to test, if the first condition is false.**

Syntax

```
if (condition) {  
    code to be executed if condition is true;  
} elseif (condition) {  
    code to be executed if condition is true;  
} else {
```

code to be executed if condition is false;

}

The example below will output "Have a good morning!" if the current time is less than 10, and "Have a good day!" if the current time is less than 20. Otherwise it will output "Have a good night!":

Example

```
<?php
```

```
$t = date("H");
```

```
if ($t < "10") {
```

```
    echo "Have a good morning!";
```

```
} elseif ($t < "20") {
```

```
    echo "Have a good day!";
```

```
} else {
```

```
    echo "Have a good night!";
```

```
}
```

```
?>
```

PHP - The switch Statement

Use the switch statement to **select one of many blocks of code to be executed.**

Syntax

```
switch (n) {
```

```
    case label1:
```

```
        code to be executed if n=label1;
```

```
        break;
```

```
    case label2:
```

```
        code to be executed if n=label2;
```

```
        break;
```

```
    case label3:
```

```
        code to be executed if n=label3;
```

```
        break;
```

```
    ...
```

```
    default:
```

```
        code to be executed if n is different from all labels;
```

```
}
```

how it works: First we have a single expression *n* (most often a variable), that is evaluated once. The value of the expression is then compared with the values for each case in the structure. If there is a match, the block of code associated with that case is executed. Use **break** to prevent the code from running into the next case automatically. The **default** statement is used if no match is found.

Example

```
<?php
```

```
$favcolor = "red";
```

```
switch ($favcolor) {
```

```
    case "red":
```

```
        echo "Your favorite color is red!";
```

```

break;
case "blue":
    echo "Your favorite color is blue!";
    break;
case "green":
    echo "Your favorite color is green!";
    break;
default:
    echo "Your favorite color is neither red, blue, or green!";
}
?>

```

12. **Explain about cookies in PHP with example.**

Cookie:

A cookie is often used to identify a user. A cookie is a small file that the server embeds on the user's computer. Each time the same computer requests a page with a browser, it will send the cookie too. With PHP, you can both create and retrieve cookie values.

Create Cookies With PHP

A cookie is created with the `setcookie()` function.

Syntax

`setcookie(name, value, expire, path, domain, secure, httponly);`

Only the **name** parameter is required. All other parameters are optional.

PHP Create/Retrieve a Cookie

The following example creates a cookie named "user" with the value "John Doe". The cookie will expire after 30 days (86400 * 30). The "/" means that the cookie is available in entire website (otherwise, select the directory you prefer).

We then retrieve the value of the cookie "user" (using the global variable `$_COOKIE`). We also use the `isset()` function to find out if the cookie is set:

Example

```

<?php
$cookie_name = "user";
$cookie_value = "John Doe";
setcookie($cookie_name, $cookie_value, time() + (86400 * 30), "/"); // 86400 = 1 day
?>
<html>
<body>
<?php
if(!isset($_COOKIE[$cookie_name])) {
    echo "Cookie named '" . $cookie_name . "' is not set!";
} else {
    echo "Cookie '" . $cookie_name . "' is set!<br>";
    echo "Value is: " . $_COOKIE[$cookie_name];
}

```

```
?>
</body>
</html>
```

Note: The setcookie() function must appear BEFORE the <html> tag.

Note: The value of the cookie is automatically URLencoded when sending the cookie, and automatically decoded when received (to prevent URLencoding, use setrawcookie() instead).

Modify a Cookie Value

To modify a cookie, just set (again) the cookie using the setcookie() function:

Example

```
<?php
$cookie_name = "user";
$cookie_value = "Alex Porter";
setcookie($cookie_name, $cookie_value, time() + (86400 * 30), "");
?>
<html>
<body>
<?php
if(!isset($_COOKIE[$cookie_name])) {
    echo "Cookie named '" . $cookie_name . "' is not set!";
} else {
    echo "Cookie '" . $cookie_name . "' is set!<br>";
    echo "Value is: " . $_COOKIE[$cookie_name];
}
?>
</body>
</html>
```

Output:

```
Cookie 'user' is set!
Value is: John Doe
```

Note: You might have to reload the page to see the new value of the cookie.

Delete a Cookie

To delete a cookie, use the setcookie() function with an expiration date in the past:

Example

```
<?php
// set the expiration date to one hour ago
setcookie("user", "", time() - 3600);
?>
<html>
<body>
<?php
echo "Cookie 'user' is deleted.";
```

```
?>
</body>
</html>
Output:
Cookie 'user' is deleted.
```

Check if Cookies are Enabled

The following example creates a small script that checks whether cookies are enabled. First, try to create a test cookie with the setcookie() function, then count the \$_COOKIE array variable:

Example

```
<?php
setcookie("test_cookie", "test", time() + 3600, '/');
?>
<html>
<body>
<?php
if(count($_COOKIE) > 0) {
    echo "Cookies are enabled.";
} else {
    echo "Cookies are disabled.";
}
?>
</body>
</html>
Output:
Cookies are enabled.
```

13. Describe the data base connections in PHP with suitable example.

Connection to MySQL database using PHP

Before you can get content out of your MySQL database, you must know how to establish a connection to MySQL from inside a PHP script. To perform basic queries from within MySQL is very easy.

The first thing to do is connect to the database. The function to connect to MySQL is called mysql_connect. This function returns a resource which is a pointer to the database connection. It's also called a database handle, and we'll use it in later functions.

```
<?php
$username = "your_name";
$password = "your_password";
$hostname = "localhost";

//connection to the database
$dbhandle = mysql_connect($hostname, $username, $password)
```

```

    or die("Unable to connect to MySQL");
    echo "Connected to MySQL<br>";
    ?>

```

All going well, you should see "Connected to MySQL" when you run this script. If you can't connect to the server, make sure your password, username and hostname are correct.

Once you've connected, you're going to want to select a database to work with. Let's assume the database is called 'examples'. To start working in this database, you'll need the `mysql_select_db()` function:

```

<?php
//select a database to work with
$selected = mysql_select_db("examples",$dbhandle)
    or die("Could not select examples");
?>

```

Now that you're connected, let's try and run some queries. The function used to perform queries is named `mysql_query()`. The function returns a resource that contains the results of the query, called the result set. To examine the results, you're going to use the `mysql_fetch_array()` function, which returns the results row by row. In the case of a query that returns a single value, the resource that the function returns is simply a value true or false.

A convenient way to access all the rows is with a while loop. Let's add the code to our script:

```

<?php
//execute the SQL query and return records
$result = mysql_query("SELECT id, model, year FROM cars");
//fetch the data from the database
while ($row = mysql_fetch_array($result)) {
    echo "ID:". $row{'id'}." Name:". $row{'model'}."
        ". $row{'year'}."<br>";
}
?>

```

Finally, we close the connection. Although this isn't strictly speaking necessary, PHP will automatically close the connection when the script ends, you should get into the habit of closing what you open.

```

<?php
//close the connection
mysql_close($dbhandle);
?>

```

Here is a code in full:

```

<?php
$username = "your_name";

```

```

$password = "your_password";
$hostname = "localhost";

//connection to the database
$dbhandle = mysql_connect($hostname, $username, $password)
  or die("Unable to connect to MySQL");
echo "Connected to MySQL<br>";

//select a database to work with
$selectd = mysql_select_db("examples",$dbhandle)
  or die("Could not select examples");

//execute the SQL query and return records
$result = mysql_query("SELECT id, model,year FROM cars");

//fetch the data from the database
while ($row = mysql_fetch_array($result)) {
  echo "ID:". $row{'id'}." Name:". $row{'model'}."Year: ". //display the results
  $row{'year'}."<br>";
}
//close the connection
mysql_close($dbhandle);
?>

```

To create 'examples' database on your MySQL server you should run the following script:

```

CREATE DATABASE `examples`;
USE `examples`;
CREATE TABLE `cars` (
  `id` int UNIQUE NOT NULL,
  `name` varchar(40),
  `year` varchar(50),
  PRIMARY KEY(id)
);
INSERT INTO cars VALUES(1,'Mercedes','2000');
INSERT INTO cars VALUES(2,'BMW','2004');
INSERT INTO cars VALUES(3,'Audi','2001');

```

14. Explain the steps in the PHP code for querying a database with suitable examples.

- Create a database connection
- Select database you wish to use
- Perform a SQL query
- Do some processing on query results
- Close database connection

1. Creating Database Connection

- Use either `mysql_connect` or `mysql_pconnect` to create database connection
 - `mysql_connect`: connection is closed at end of script (end of page)
 - `mysql_pconnect`: creates persistent connection
 - connection remains even after end of the page
 - Parameters
 - Server – hostname of server
 - Username – username on the database
 - Password – password on the database
 - New Link (`mysql_connect` only) – reuse database connection created by previous call to `mysql_connect`
 - Client Flags
 - `MYSQL_CLIENT_SSL` :: Use SSL
 - `MYSQL_CLIENT_COMPRESS` :: Compress data sent to MySQL
- Username and password fields imply that database password is sitting there in the source code
- If someone gains access to source code, can compromise the database
 - Servers are sometimes configured to view PHP source code when a resource is requested with “.phps” instead of “.php”
 - One approach to avoid this: put this information in Web server config. File
- Then ensure the Web server config. file is not externally accessible

2. Selecting a Database

- `mysql_select_db()`
 - Pass it the database name
- Related:
 - `mysql_list_dbs()`
- List databases available
 - `Mysql_list_tables()`
- List database tables available

3. Perform SQL Query

- Create query string
 - `$query = 'SQL formatted string'`
 - `$query = 'SELECT * FROM table'`
- Submit query to database for processing
 - `$result = mysql_query($query);`
 - For UPDATE, DELETE, DROP, etc, returns TRUE or FALSE
 - For SELECT, SHOW, DESCRIBE or EXPLAIN, `$result` is an identifier for the results, and does not contain the results themselves
- `$result` is called a “resource” in this case
- A result of FALSE indicates an error
- If there is an error
 - `mysql_error()` returns error string from last MySQL call

4. Process Results

- Many functions exist to work with database results
- `mysql_num_rows()`
 - Number of rows in the result set
 - Useful for iterating over result set

- `mysql_fetch_array()`
 - Returns a result row as an array
 - Can be associative or numeric or both (default)
 - `$row = mysql_fetch_array($result);`
 - `$row['column name']` :: value comes from database row with specified column name
 - `$row[0]` :: value comes from first field in result set

Process Results Loop

- Easy loop for processing results:


```
$result = mysql_query($qstring);
$num_rows = mysql_num_rows($result);
for ($i=0; $i<$num_rows; $i++)
{
    $row = mysql_fetch_array($result);
    // take action on database results here
}
```

5. Closing Database Connection

- `mysql_close()`
 - Closes database connection
 - Only works for connections opened with `mysql_connect()`
 - Connections opened with `mysql_pconnect()` ignore this call
 - Often not necessary to call this, as connections created by `mysql_connect` are closed at the end of the script anyway

15. With example explain about XSL and XSLT transformation

XSL stands for EXtensible Stylesheet Language.

XSLT:

- XSLT stands for XSL Transformations
- XSLT is the most important part of XSL
- XSLT transforms an XML document into another XML document
- XSLT uses XPath to navigate in XML documents
- XSLT is a W3C Recommendation.

XSLT Elements

Description of all the XSLT elements from the W3C Recommendation, and information about browser support.

XSLT Functions

XSLT includes over 100 built-in functions. There are functions for string values, numeric values, date and time comparison, node and QName manipulation, sequence manipulation, Boolean values, and more.

XSLT = XSL Transformations

XSLT is the most important part of XSL. XSLT is used to transform an XML document into another XML document, or another type of document that is recognized by a browser, like HTML and XHTML. Normally XSLT does this by transforming each XML element into an (X)HTML element. With XSLT you can add/remove elements and attributes to or from the output file. You

can also rearrange and sort elements, perform tests and make decisions about which elements to hide and display, and a lot more. A common way to describe the transformation process is to say that **XSLT transforms an XML source-tree into an XML result-tree**.

XSLT Uses XPath

XSLT uses XPath to find information in an XML document. XPath is used to navigate through elements and attributes in XML documents.

Correct Style Sheet Declaration

The root element that declares the document to be an XSL style sheet is `<xsl:stylesheet>` or `<xsl:transform>`.

Note: `<xsl:stylesheet>` and `<xsl:transform>` are completely synonymous and either can be used!

The correct way to declare an XSL style sheet according to the W3C XSLT Recommendation is:

```
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

To get access to the XSLT elements, attributes and features we must declare the XSLT namespace at the top of the document.

The `xmlns:xsl="http://www.w3.org/1999/XSL/Transform"` points to the official W3C XSLT namespace. If you use this namespace, you must also include the attribute `version="1.0"`.

Start with a Raw XML Document

We want to transform the following XML document ("cdcatalog.xml") into XHTML:

```
<?xml version="1.0" encoding="UTF-8"?>
<catalog>
  <cd>
    <title>Empire Burlesque</title>
    <artist>Bob Dylan</artist>
    <country>USA</country>
    <company>Columbia</company>
    <price>10.90</price>
    <year>1985</year>
  </cd>
  .
  .
</catalog>
```

1. Create an XSL Style Sheet

Then you create an XSL Style Sheet ("cdcatalog.xsl") with a transformation template:

```
<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
  <body>
  <h2>My CD Collection</h2>
  <table border="1">
  <tr bgcolor="#9acd32">
    <th>Title</th>
    <th>Artist</th>
  </tr>
  <xsl:for-each select="catalog/cd">
  <tr>
    <td><xsl:value-of select="title"/></td>
    <td><xsl:value-of select="artist"/></td>
  </tr>
  </xsl:for-each>
  </table>
  </body>
  </html>
</xsl:template>

</xsl:stylesheet>
```

2. Link the XSL Style Sheet to the XML Document

Add the XSL style sheet reference to your XML document ("cdcatalog.xml"):

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="cdcatalog.xsl"?>
<catalog>
  <cd>
    <title>Empire Burlesque</title>
    <artist>Bob Dylan</artist>
    <country>USA</country>
    <company>Columbia</company>
    <price>10.90</price>
    <year>1985</year>
  </cd>
```

</catalog>

The result is:



16. Explain about DOM with the XML data processing.
A huge benefit of XML – standard parsers and standard (cross-language) APIs for processing it

The DOM is a W3C (World Wide Web Consortium) standard.

The DOM defines a standard for accessing documents like XML and HTML:

"The W3C Document Object Model (DOM) is a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a document."

DOM: an object-oriented representation of the XML parse tree (roughly like the Data Model graph)

- DOM objects have methods like “getFirstChild()”, “getNextSibling”
- Common way of traversing the tree
- Can also modify the DOM tree – alter the XML – via insertAfter(), etc.

The XML DOM is:

- A standard object model for XML
- A standard programming interface for XML
- Platform- and language-independent
- A W3C standard

The XML DOM defines the **objects and properties** of all XML elements, and the **methods** (interface) to access them.

DOM Nodes

According to the DOM, everything in an XML document is a **node**.

The DOM says:

- The entire document is a document node
- Every XML element is an element node

- The text in the XML elements are text nodes
- Every attribute is an attribute node
- Comments are comment nodes

DOM Example

Look at the following XML file ([books.xml](#)):

```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title lang="en">XQuery Kick Start</title>
    <author>James McGovern</author>
    <author>Per Bothner</author>
    <author>Kurt Cagle</author>
    <author>James Linn</author>
    <author>Vaidyanathan Nagarajan</author>
    <year>2003</year>
    <price>49.99</price>
  </book>
  <book category="web" cover="paperback">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

The root node in the XML above is named `<bookstore>`. All other nodes in the document are contained within `<bookstore>`.

The root node `<bookstore>` holds four `<book>` nodes.

The first `<book>` node holds four nodes: `<title>`, `<author>`, `<year>`, and `<price>`, which contains one text

node each, "Everyday Italian", "Giada De Laurentiis", "2005", and "30.00".

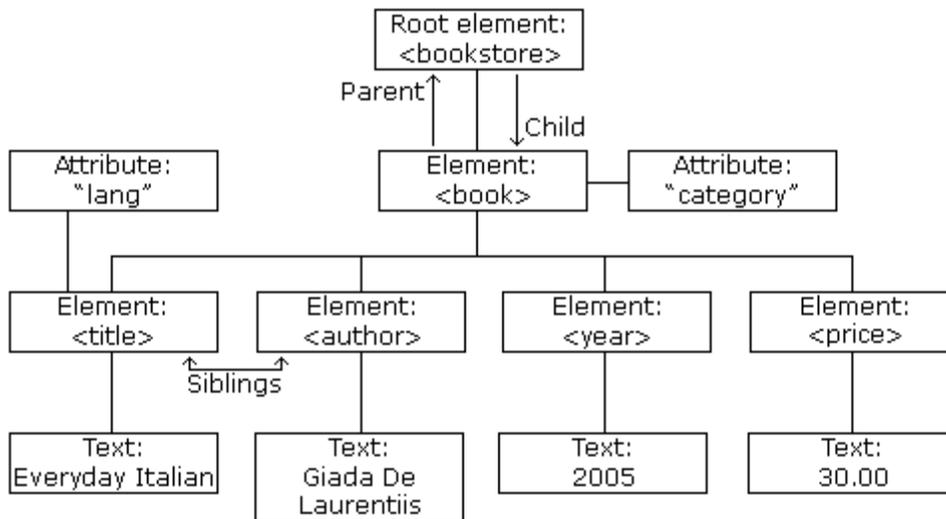
Text is Always Stored in Text Nodes

A common error in DOM processing is to expect an element node to contain text. However, the text of an element node is stored in a text node. In this example: `<year>2005</year>`, the element node `<year>`, holds a text node with the value "2005". "2005" is **not** the value of the `<year>` element!

The XML DOM Node Tree

The XML DOM views an XML document as a tree-structure. The tree structure is called a **node-tree**. All nodes can be accessed through the tree. Their contents can be modified or deleted, and new elements can be created.

The node tree shows the set of nodes, and the connections between them. The tree starts at the root node and branches out to the text nodes at the lowest level of the tree:



Node Parents, Children, and Siblings

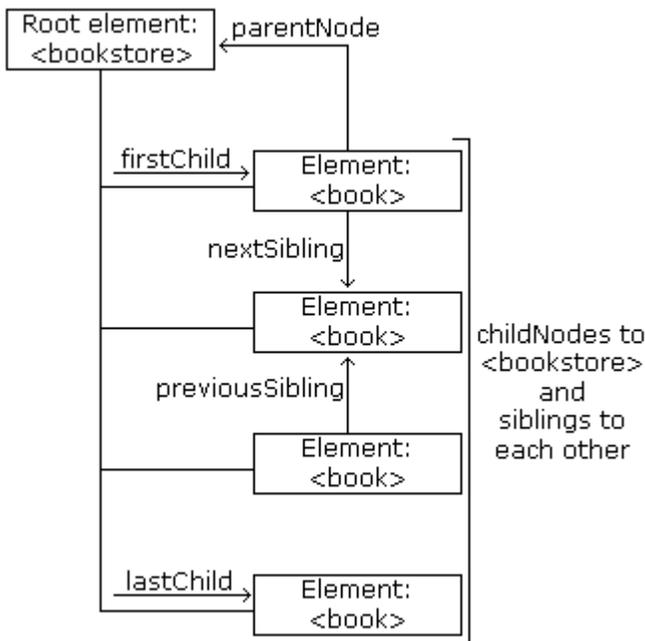
The nodes in the node tree have a hierarchical relationship to each other.

The terms parent, child, and sibling are used to describe the relationships. Parent nodes have children.

Children on the same level are called siblings (brothers or sisters).

- In a node tree, the top node is called the root
- Every node, except the root, has exactly one parent node
- A node can have any number of children
- A leaf is a node with no children
- Siblings are nodes with the same parent

The following image illustrates a part of the node tree and the relationship between the nodes:



XML Parser

The XML DOM contains methods to traverse XML trees, access, insert, and delete nodes. However, before an XML document can be accessed and manipulated, it must be loaded into an XML DOM object.

An XML parser reads XML, and converts it into an XML DOM object that can be accessed with JavaScript.

Most browsers have a built-in XML parser.

Load an XML Document

The following JavaScript fragment loads an XML document ("books.xml"):

```

if (window.XMLHttpRequest)
{
  xmlhttp=new XMLHttpRequest();
}
else // code for IE5 and IE6
{
  xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
}
xmlhttp.open("GET","books.xml",false);
xmlhttp.send();
xmlDoc=xmlhttp.responseXML;
  
```

The following code loads and parses an XML string:

```

if (window.DOMParser)
{
  parser=new DOMParser();
  xmlDoc=parser.parseFromString(text,"text/xml");
}
  
```

```
}  
else // code for IE  
{  
  xmlDoc=new ActiveXObject("Microsoft.XMLDOM");  
  xmlDoc.async=false;  
  xmlDoc.loadXML(text);  
}
```

Accessing Nodes

You can access a node in three ways:

1. By using the `getElementsByTagName()` method
2. By looping through (traversing) the nodes tree.
3. By navigating the node tree, using the node relationships.

The `getElementsByTagName()` Method

`getElementsByTagName()` returns all elements with a specified tag name.

Syntax

```
node.getElementsByTagName("tagname");
```

Example

The following example returns all `<title>` elements under the `x` element:

```
x.getElementsByTagName("title");
```

Note that the example above only returns `<title>` elements under the `x` node. To return all `<title>` elements in the XML document use:

```
xmlDoc.getElementsByTagName("title");
```

where `xmlDoc` is the document itself (document node).

DOM Node List

The `getElementsByTagName()` method returns a node list. A node list is an array of nodes.

The following code loads ["books.xml"](#) into `xmlDoc` using [loadXMLDoc\(\)](#) and stores a list of `<title>` nodes (a node list) in the variable `x`:

```
xmlDoc=loadXMLDoc("books.xml");
```

```
x=xmlDoc.getElementsByTagName("title");
```

The `<title>` elements in `x` can be accessed by index number. To access the third `<title>` you can write::

```
y=x[2];
```

Note: The index starts at 0.

You will learn more about node lists in a later chapter of this tutorial.

DOM Node List Length

The `length` property defines the length of a node list (the number of nodes).

You can loop through a node list by using the `length` property:

Example

```
var xmlDoc=loadXMLDoc("books.xml");
```

```
var x=xmlDoc.getElementsByTagName("title");
```

```
for (i=0;i<x.length;i++)
```

```
{
```

```
document.write(x[i].childNodes[0].nodeValue);
document.write("<br>");
}
```

Node Types

The **documentElement** property of the XML document is the root node.

The **nodeName** property of a node is the name of the node.

The **nodeType** property of a node is the type of the node.

You will learn more about the node properties in the next chapter of this tutorial.

Traversing Nodes

The following code loops through the child nodes, that are also element nodes, of the root node:

Example

```
var xmlDoc=loadXMLDoc("books.xml");
var x=xmlDoc.documentElement.childNodes;

for (i=0;i<x.length;i++)
{
// Process only element nodes (type 1)
if (x[i].nodeType==1)
{
document.write(x[i].nodeName);
document.write("<br>");
}
}
```

Example explained:

1. Load "[books.xml](#)" into xmlDoc using [loadXMLDoc\(\)](#)
2. Get the child nodes of the root element
3. For each child node, check the node type of the node. If the node type is "1" it is an element node
4. Output the name of the node if it is an element node

Navigating Node Relationships

The following code navigates the node tree using the node relationships:

Example

```
var xmlDoc=loadXMLDoc("books.xml");

var x=xmlDoc.getElementsByTagName("book")[0].childNodes;
var y=xmlDoc.getElementsByTagName("book")[0].firstChild;

for (i=0;i<x.length;i++)
{
// Process only element nodes (type 1)
if (y.nodeType==1)
```

```

{
  document.write(y.nodeName + "<br>");
}
y=y.nextSibling;
}

```

1. Load "books.xml" into xmlDoc using [loadXMLDoc\(\)](#)
2. Get the child nodes of the first book element
3. Set the "y" variable to be the first child node of the first book element
4. For each child node (starting with the first child node "y"):
5. Check the node type. If the node type is "1" it is an element node
6. Output the name of the node if it is an element node
7. Set the "y" variable to be the next sibling node, and run through the loop again

Traversing the Node Tree

Often you want to loop an XML document, for example: when you want to extract the value of each element.

This is called "Traversing the node tree"

The example below loops through all child nodes of <book>, and displays their names and values:

Example

```

<html>
<head>
<script src="loadxmlstring.js"></script>
</head>
<body>
<script>
var text="<book>";
text=text+"<title>Everyday Italian</title>";
text=text+"<author>Giada De Laurentiis</author>";
text=text+"<year>2005</year>";
text=text+"</book>";

var xmlDoc=loadXMLString(text);

// documentElement always represents the root node
var x=xmlDoc.documentElement.childNodes;

for (i=0;i<x.length;i++)
{ document.write(x[i].nodeName);
  document.write(": ");
  document.write(x[i].childNodes[0].nodeValue);
  document.write("<br>");
}
</script>
</body>

```

```
</html>
```

Output:

title: Everyday Italian

author: Giada De Laurentiis

year: 2005

Example explained:

1. [loadXMLString\(\)](#) loads the XML string into xmlDoc
2. Get the child nodes of the root element
3. For each child node, output the node name and the node value of the text node

17. Discuss in detail about the XML DTD**XML DTD**

An XML document with correct syntax is called "Well Formed".

An XML document validated against a DTD is "Well Formed" and "Valid".

Valid XML Documents

A "Valid" XML document is a "Well Formed" XML document, which also conforms to the rules of a DTD:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE note SYSTEM "Note.dtd">
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

The DOCTYPE declaration, in the example above, is a reference to an external DTD file. The content of the file is shown in the paragraph below.

The purpose of a DTD is to define the structure of an XML document. It defines the structure with a list of legal elements:

```
<!DOCTYPE note
[
<!ELEMENT note (to,from,heading,body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
]>
```

The DTD above is interpreted like this:

- !DOCTYPE note defines that the root element of the document is note
- !ELEMENT note defines that the note element must contain four elements: "to, from, heading, body"
- !ELEMENT to defines the to element to be of type "#PCDATA"
- !ELEMENT from defines the from element to be of type "#PCDATA"
- !ELEMENT heading defines the heading element to be of type "#PCDATA"
- !ELEMENT body defines the body element to be of type "#PCDATA"

#PCDATA means parse-able text data.

Using DTD for Entity Declaration

A doctype declaration can also be used to define special characters and character strings, used in the document:

Example

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE note [
<!ENTITY nbsp "&#xA0;">
<!ENTITY writer "Writer: Donald Duck.">
<!ENTITY copyright "Copyright: W3Schools.">
]>

<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
<footer>&writer;&nbsp;&copyright;</footer>
</note>
```

With a DTD, independent groups of people can agree on a standard for interchanging data. With a DTD, you can verify that the data you receive from the outside world is valid.

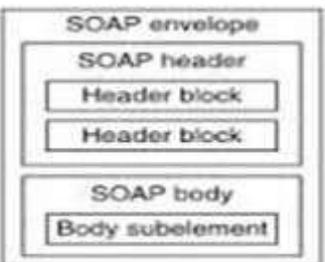
Unit-V

Part – A

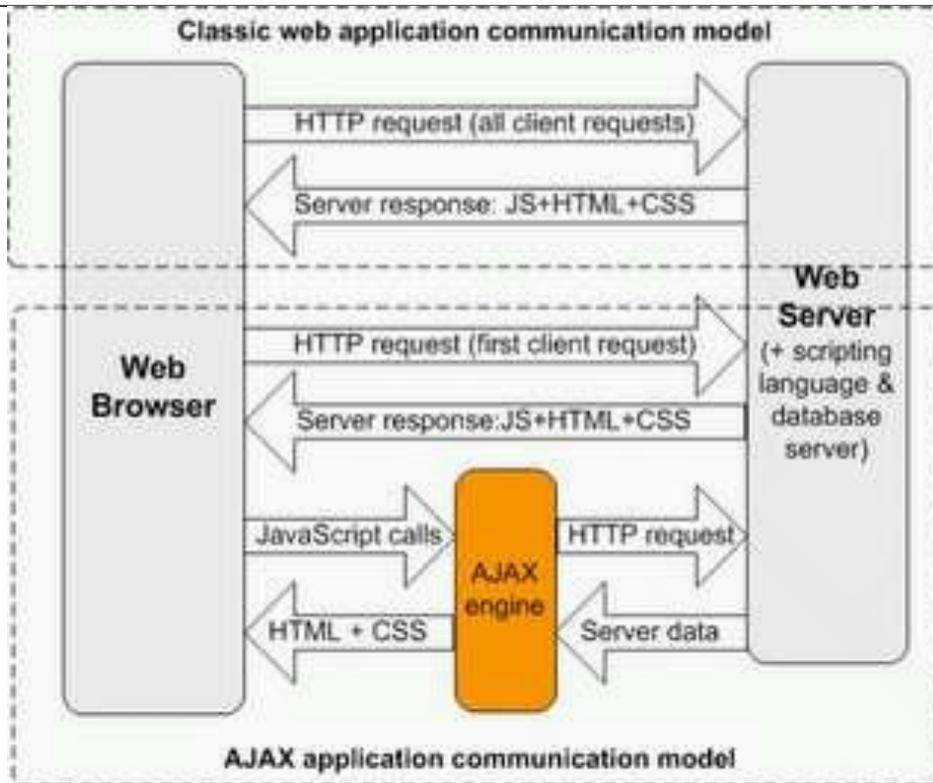
- 1. What is Ajax?**
Ajax is a set of client side technologies that provides asynchronous communication between user interfaces and web server. So the advantages of using Ajax are asynchronous communication, minimal data transfer and server is not overloaded with unnecessary load.
- 2. What technologies are being used in AJAX?**
AJAX uses four technologies, which are as follows:
JavaScript, XMLHttpRequest, Document Object Model (DOM), Extensible HTML (XHTML) and Cascading Style Sheets (CSS)
- 3. Explain the limitations of AJAX.**

	It is difficult to bookmark a particular state of the application, Function provided in the code-behind file do not work because the dynamic pages cannot register themselves on browsers history engine automatically
4.	Describe AJAX Control Extender Toolkit. AJAX Control Toolkit is a set of extenders that are used to extend the functionalities of the ASP.NET controls. The extenders use a block of JavaScript code to add new and enhanced capabilities to the ASP.NET controls. AJAX Control Toolkit is a free download available on the Microsoft site. You need to install this toolkit on your system before using extenders.
5.	30) What is the syntax to create AJAX objects? AJAX uses the following syntax to create an object: Var myobject = new AjaxObject("page path"); The page path is the URL of the Web page containing the object that you want to call. The URL must be of the same domain as the Web page.
6.	How can you find out that an AJAX request has been completed? You can find out that an AJAX request has been completed by using the readyState property. If the value of this property equals to four, it means that the request has been completed and the data is available.
7.	What are the different ways to pass parameters to the server? We can pass parameters to the server using either the GET or POST method. The following code snippets show the example of both the methods: Get: XmlHttpRequest.Open("GET", "file1.txt", true); Post: XmlHttpRequest.Open("POST", "file2.txt", true);
8.	What are the extender controls? The extender controls uses a block of JavaScript code to add new and enhanced capabilities to ASP.NET. The developers can use a set of sample extender controls through a separate download - AJAX Control Toolkit (ACT).
9.	List out the advantages of AJAX. (May 2014) <ul style="list-style-type: none"> • Better interactivity • Easier navigation • Compact • Backed by reputed brands
10.	Define Web service? (Nov 2011) A Web service is a method of communication between two electronic devices over the web. The W3C defines a "Web service" as "a software system designed to support interoperable machine-to-machine interaction over a network". It has an interface described in a machine-processable format specifically Web Services Description Language (WSDL).
11.	What are the different applications that could use web services?? <ul style="list-style-type: none"> ❖ Data providers, for example, those that provide data such as a stock quote ❖ Business-to-business process integrations, such as those that send a purchase order from one company to another ❖ Integration with multiple partners, and even with competitors ❖ Enterprise application integration, for example, integration of a company's e-mail database with its human resources (HR) database
12.	What are the features of web service? Web services are having the features such as heterogeneous, interoperable, loosely coupled, and implementation-independent programs and modular design
13.	What are the rules to be followed in designing the web service? <ul style="list-style-type: none"> ❖ Allow extensibility points. ❖ Keep your namespaces easy to version by placing dates in them. ❖ Don't try to solve every problem with one schema, WSDL, or other file. Break out the problem into pieces
14.	What is meant by WSDL? (APR/MAY 2011) <ul style="list-style-type: none"> • WSDL stands for Web Services Description Language • WSDL is based on XML

	<ul style="list-style-type: none"> • WSDL is used to describe Web services • WSDL is used to locate Web services • WSDL is an XML-based language for locating and describing Web services 												
15.	<p>Why do you want to describe a web service? (MAY/JUNE 2014) Web Services Description Language (WSDL) is a document written in XML. The document describes a Web service. It specifies the location of the service and the operations (or methods) the service exposes.</p>												
16.	<p>17. What are the elements of WSDL?</p> <table border="1"> <thead> <tr> <th>Element Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>types</td> <td>A container for abstract type definitions defined using XML Schema</td> </tr> <tr> <td>message</td> <td>A definition of an abstract message that may consist of multiple parts, each part may be of a different type</td> </tr> <tr> <td>portType</td> <td>An abstract set of operations supported by one or more endpoints (commonly known as an interface); operations are defined by an exchange of messages</td> </tr> <tr> <td>binding</td> <td>A concrete protocol and data format specification for a particular portType</td> </tr> <tr> <td>service</td> <td>A collection of related endpoints, where an endpoint is defined as a combination of a binding and an address (URI)</td> </tr> </tbody> </table>	Element Name	Description	types	A container for abstract type definitions defined using XML Schema	message	A definition of an abstract message that may consist of multiple parts, each part may be of a different type	portType	An abstract set of operations supported by one or more endpoints (commonly known as an interface); operations are defined by an exchange of messages	binding	A concrete protocol and data format specification for a particular portType	service	A collection of related endpoints, where an endpoint is defined as a combination of a binding and an address (URI)
Element Name	Description												
types	A container for abstract type definitions defined using XML Schema												
message	A definition of an abstract message that may consist of multiple parts, each part may be of a different type												
portType	An abstract set of operations supported by one or more endpoints (commonly known as an interface); operations are defined by an exchange of messages												
binding	A concrete protocol and data format specification for a particular portType												
service	A collection of related endpoints, where an endpoint is defined as a combination of a binding and an address (URI)												
18.	<p>What is the use of web services?</p> <ul style="list-style-type: none"> • Web services encompass a set of related standards that can enable two computers • The data is passed back and forth using standard protocols such as HTTP, the same protocol used to transfer ordinary web pages. • Web services operate using open, text-based standards that enable components written in different languages and on different platforms to communicate. • They are ready to use pieces of software on the Internet. XML, SOAP, Web Services Description Language (WSDL) and Universal Description, Discovery and Integration (UDDI) are the standards on which web services rely. • UDDI is another XML based format that enables developers and business to publish and locate Web services on a network. 												
19.	<p>State the uses of WSDL. (APR/MAY 2012)</p> <ul style="list-style-type: none"> • WSDL stands for Web Services Description Language. • WSDL is a document written in XML. • WSDL is an XML-based language for locating and describing Web services. 												
20.	<p>What are the four transmission types of WSDL?</p> <ul style="list-style-type: none"> ❖ One-way ❖ Request-response ❖ Solicit-response ❖ Notification 												
21.	<p>State the significance of a WSDL document. (NOV/DEC 2012) The WSDL is a Web Service Descriptor Language which is based on XML.</p> <table border="1"> <thead> <tr> <th>ELEMENT</th> <th>DESCRIPTION</th> </tr> </thead> <tbody> <tr> <td>Types</td> <td>It Specifies the data types of the symbols used by the web services.</td> </tr> <tr> <td>Messages</td> <td>It specifies the messages used by the web services.</td> </tr> <tr> <td>Porttype</td> <td>It specifies the name of the operations</td> </tr> <tr> <td>Binding</td> <td>It specifies the name of the protocol of the web services, typically it is SOAP.</td> </tr> </tbody> </table>	ELEMENT	DESCRIPTION	Types	It Specifies the data types of the symbols used by the web services.	Messages	It specifies the messages used by the web services.	Porttype	It specifies the name of the operations	Binding	It specifies the name of the protocol of the web services, typically it is SOAP.		
ELEMENT	DESCRIPTION												
Types	It Specifies the data types of the symbols used by the web services.												
Messages	It specifies the messages used by the web services.												
Porttype	It specifies the name of the operations												
Binding	It specifies the name of the protocol of the web services, typically it is SOAP.												
22.	<p>What is UDDI? (NOV/DEC 2011) UDDI means Universal Description, Discovery and Integration. UDDI - platform-independent framework for describing services, discovering businesses, and integrating business</p>												

	<p>services by using the Internet.</p> <ul style="list-style-type: none"> - directory for storing information about web services - directory of web service interfaces described by WSDL - communicates via SOAP - The core of UDDI is the UDDI Business Registry, a global, public, online directory.
23.	<p>What are the benefits of UDDI?</p> <p>Problems the UDDI specification can help to solve:</p> <ul style="list-style-type: none"> • Making it possible to discover the right business from the millions currently online • Defining how to enable commerce once the preferred business is discovered • Reaching new customers and increasing access to current customers • Expanding offerings and extending market reach • Solving customer-driven need to remove barriers to allow for rapid participation in the global Internet economy • Describing services and business processes programmatically in a single, open, and secure environment
24.	<p>What are the core elements of UDDI?</p> <p>UDDI defines four core data elements within the data model:</p> <ul style="list-style-type: none"> • businessEntity (modeling business information) • businessService (describing a service) • tModel (describing specifications, classifications, or identifications) • binding Template (mapping between a businessService and the set of tModels that describe its technical fingerprint)
25.	<p>List some examples of web services. (APR/MAY 2012)</p> <ul style="list-style-type: none"> • Geo IP: http://www.websvcex.net/geopiservice.asmx?op=GetGeoIP • Whois: http://www.websvcex.net/whois.asmx?op=GetWhoIS <p>SMS: http://www.websvcex.net/sendsmsworld.asmx</p>
26.	<p>List out some web service technologies?</p> <ul style="list-style-type: none"> • XML • SOAP • WSDL
27.	<p>What is SOAP?</p> <p>SOAP - Simple Object Access Protocol</p> <ul style="list-style-type: none"> - protocol specification for exchanging structured information in the implementation of Web Services in computer networks. - relies on Extensible Markup Language (XML) for its message format, and usually relies on other Application Layer protocols, most notably Hypertext Transfer Protocol (HTTP) and Simple Mail Transfer Protocol (SMTP), for message negotiation and transmission.
28.	<p>Define SOAP structure.</p> <div style="text-align: center;">  <pre> graph TD subgraph SOAP_envelope [SOAP envelope] subgraph SOAP_header [SOAP header] HB1[Header block] HB2[Header block] end subgraph SOAP_body [SOAP body] BS[Body subelement] end end </pre> </div> <p>The SOAP envelope <Envelope> is the root element in every SOAP message, and contains two child elements, an optional <Header> element, and a mandatory <Body> element.</p> <p>The SOAP header</p>

	<p><Header> is an optional sub element of the SOAP envelope, and is used to pass application-related information that is to be processed by SOAP nodes along the message path; see The SOAP header.</p> <p>The SOAP body</p> <p><Body> is a mandatory sub element of the SOAP envelope, which contains information intended for the ultimate recipient of the message; see The SOAP body.</p> <p>The SOAP fault</p> <p><Fault> is a sub element of the SOAP body, which is used for reporting errors; see The SOAP fault.</p> <p>XML elements in <Header> and <Body> are defined by the applications that make use of them, although the SOAP specification imposes some constraints on their structure.</p>
29.	<p>Define the need for SOAP. (APR/MAY 2013)</p> <p>Simple Object Access Protocol (SOAP) is a protocol based on XML. It is used by the web services for exchange of information. The Client- Server communication is based on RPC. The HTTP does not design to handle the distributed objects that are required by the RPC. Hence another application protocol is build over HTTP which popularly known as SOAP. SOAP allows talking different applications that are running in two different operating systems.</p>
30.	<p>What are the descriptions in SOAP service?</p> <p>To describe everything a SOAP service needs to describe the following:</p> <ul style="list-style-type: none"> • The operations • The schema for each message in an operation • The SOAPAction headers • The URL endpoint of the service
31.	<p>Give an example of a web services registry and its function. (NOV/DEC 2012)</p> <p>It refers to a place in which service providers can impart information about their offered services and potential clients can search for services</p> <p><i>Example:</i> IBM - WebSphere Service Registry, Oracle Service Registry etc.,</p>
32.	<p>Mention some of the disadvantageous of web services (MAY/JUNE 2014)</p> <p>Web services standards features such as transactions are currently nonexistent or still in their infancy compared to more mature distributed computing open standards such as CORBA. Web services may suffer from poor performance compared to other distributed computing approaches such as RMI, CORBA, or DCOM.</p>
33.	<p>What is JWSDP?</p> <p>Java Web Service Developer Pack (JWSDP) is a tool. Using the JWSDP tool the simple implementation files written in java can be converted to Web Service.</p>
34.	<p>What are the specifications of web service architecture?</p> <p>The specifications are</p> <ul style="list-style-type: none"> ❖ Standards based ❖ Modular ❖ Federated ❖ General purpose
Part-B	
1.	<p>Explain about the object that helps AJAX reload parts of a web page without reloading the whole page. (NOV/DEC 2011, MAY/JUNE 2014)</p> <p>AJAX = Asynchronous JavaScript and XML.</p> <p>AJAX is a technique for creating fast and dynamic web pages.</p> <p>AJAX allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes. This means that it is possible to update parts of a web page, without reloading the whole page.</p> <p>Classic web pages, (which do not use AJAX) must reload the entire page if the content should change.</p> <p>AJAX is about updating parts of a web page, without reloading the whole page.</p>



Ajax request and response objects

The core of AJAX is the **XMLHttpRequest** object (available in client side scripting languages like javascript). The XMLHttpRequest object is used to exchange data with a server behind the scenes. All modern browsers (IE7+, Firefox, Chrome, Safari, and Opera) have a built-in XMLHttpRequest object. **If you are using IE 5 or IE6** (I wonder if someone still uses it), then **ActiveXObject** will be used for server communication to send ajax requests.

A new object of XMLHttpRequest is created like this :

```
//Creating a new XMLHttpRequest object
var xmlhttp;
if (window.XMLHttpRequest)
{
    xmlhttp = new XMLHttpRequest(); //for IE7+, Firefox,
    Chrome, Opera, Safari
}
else
{
    xmlhttp = new ActiveXObject("Microsoft.XMLHTTP"); //for
    IE6, IE5
}
```

This xmlhttp variable can be re-used to send multiple ajax requests, without creating new objects.

XMLHttpRequest is subject to the browser's same origin policy for security reasons. It means that requests will only succeed if they are made to the same server that served the original web page.

Useful methods to work with XMLHttpRequest

To send request and set request attributes, XMLHttpRequest object has some methods.

a) **open(method, url, isAsync, userName, password)**

The HTTP and HTTPS requests of the **XMLHttpRequest** object must be initialized through the **open method**. This method specifies the type of request (GET, POST etc.), the URL, and if the request should be handled asynchronously or not. I will cover this third parameter in next section.

The fourth and fifth parameters are the username and password, respectively. These parameters, or just the username, may be provided for authentication and authorization if required by the server for this request.

```
Example: xmlhttp.open("GET", "report_data.xml", true);  
xmlhttp.open("GET", "sensitive_data.xml", false);  
xmlhttp.open("POST", "saveData", true, "myUserName", "somePassword");
```

b) **setRequestHeader(name, value)**

Upon successful initialization of a request, the **setRequestHeader** method of the **XMLHttpRequest** object can be invoked to **send HTTP headers with the request**.

Example:

```
//Tells server that this call is made for ajax  
purposes.  
1 xmlhttp.setRequestHeader('X-Requested-With',  
2 'XMLHttpRequest');
```

c) **send(payload)**

To **send an HTTP request**, the **send** method of the **XMLHttpRequest** must be invoked. This method accepts a single parameter containing the content to be sent with the request.

The content is necessary in POST requests. For GET methods, simply pass null as parameter.

Example:

```
xmlhttp.send(null); //Request with no data in request body;  
1 Mostly used in GET requests.  
2 xmlhttp.send( {"id": "23423"} ); //Request with data in request  
body; Mostly used in POST/ PUT requests.
```

4) **abort()**

This method **aborts the request if the readyState of the XMLHttpRequest object has not yet become 4** (request complete). The **abort** method ensures that the callback method does not get invoked in an asynchronous request.

Syntax:

```
1 //Abort the processing  
2 xmlhttp.abort();
```

Apart from above method, **onreadystatechange** event listener is very important which we will discuss in next section.

Synchronous and Asynchronous requests

XMLHttpRequest object is capable of sending synchronous and asynchronous requests, as required within webpage. The behavior is controlled by **third parameter of open method**. This parameter is set to **true for an asynchronous requests, and false for synchronous requests**.

```
xmlhttp.open("GET", "report_data.xml", true); //Asynchronous
1request as third parameter is true
2xmlhttp.open("GET", "report_data.xml", false); Synchronous
request as third parameter is false
```

The default value of this parameter is “true” if it is not provided.

Asynchronous Ajax Requests do not block the webpage and user can continue to interact with other elements on the page, while the request is processed on server. You should always use asynchronous Ajax Requests because a synchronous Ajax Request makes the UI (browser) unresponsive. It means user will not be able to interact with the webpage, until the request is complete.

Synchronous requests should be used in rare cases with utmost care. For example, synchronous Ajax Request should be used if you’re embedding a new JavaScript file on the client using ajax and then referencing types and/or objects from that JavaScript file. Then the fetching of this new JS file should be included through using a synchronous Ajax Request.

Example synchronous request

```
var request = new XMLHttpRequest();
request.open('GET', '/bar/foo.txt', false); // "false" makes
the request synchronous
request.send(null);
```

```
if(request.status === 200)
{
    //request successful; handle the response
}
else
{
    //Request failed; Show error message
}
```

Example asynchronous request

```
var request = new XMLHttpRequest();
request.open('GET', '/bar/foo.txt', true); // "true" makes the
request asynchronous
```

```
request.onreadystatechange = function() {
    if (request.readyState == 4) {
        if (request.status == 200)
        {
            //request succeed
        }
        else
        {
            //request failed
        }
    }
};
```

```
request.send(null)
```

In above example, onreadystatechange is a event listener registered with XMLHttpRequest request. onreadystatechange stores a function that will process the response returned from the server. It will be called for all important events in request's life cycle. Every time an step is completed in request processing, the value of readyState will be changed and set to some other value.

0 : request not initialized

1 : server connection established

2 : request received

3 : processing request

4 : request finished and response is ready to be handled

Handling returned response from server

To get the response from a server, responseText or responseXML property of the XMLHttpRequest object is used. If the response from the server is XML, and you want to parse it as an XML object, use the responseXML property. If the response from the server is not XML, use the responseText property.

responseText : Get the response from server as a string

responseXML : Get the response from server as XML

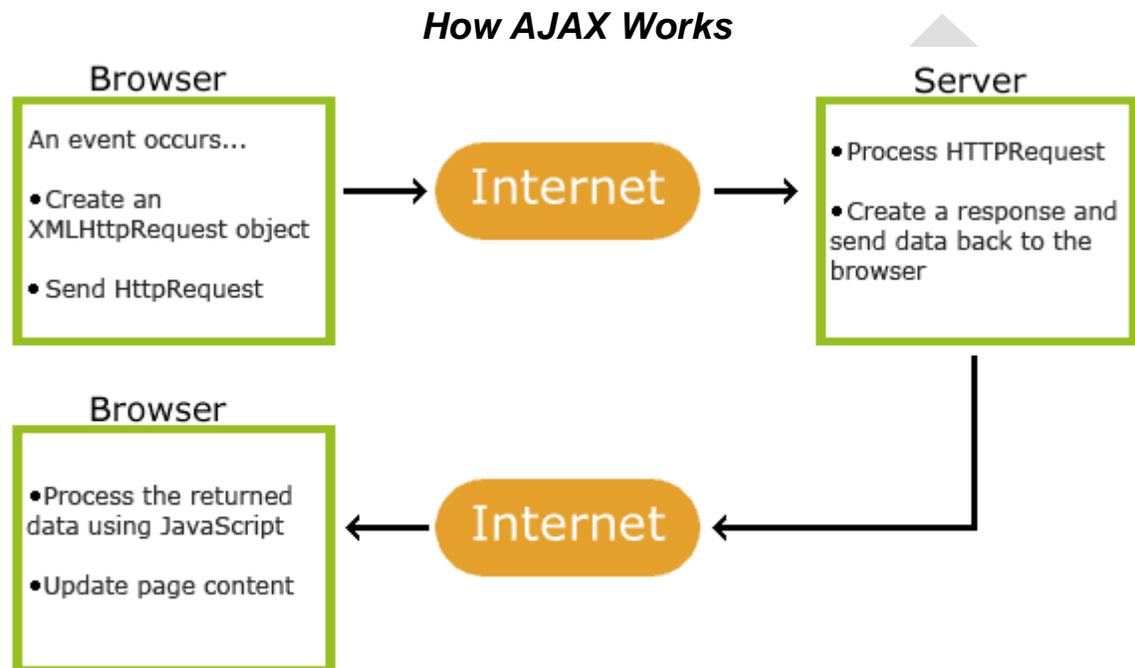
Example code:

```
if (xmlhttp.readyState == 4) {
    if (xmlhttp.status == 200)
    {
        document.getElementById("message").innerHTML =
xmlhttp.responseText;
    }
    else {
        alert('Something is wrong !!');
    }
}
```

2. Explain technologies are being used in AJAX?

- AJAX stands for Asynchronous JavaScript and XML. AJAX is a new technique for creating better, faster, and more interactive web applications with the help of XML, HTML, CSS, and JavaScript.
- Ajax uses XHTML for content, CSS for presentation, along with Document Object Model and JavaScript for dynamic content display.
- Conventional web applications transmit information to and from the sever using synchronous requests. It means you fill out a form, hit submit, and get directed to a new page with new information from the server.
- With AJAX, when you hit submit, JavaScript will make a request to the server, interpret the results, and update the current screen. In the purest sense, the user would never know that anything was even transmitted to the server.
- XML is commonly used as the format for receiving server data, although any format, including plain text, can be used.
- AJAX is a web browser technology independent of web server software.
- A user can continue to use the application while the client program requests information from the server in the background.

- Intuitive and natural user interaction. Clicking is not required, mouse movement is a sufficient event trigger.
- Data-driven as opposed to page-driven.



AJAX is Based on Open Standards

- AJAX is based on the following open standards:
 - Browser-based presentation using HTML and Cascading Style Sheets (CSS).
 - Data is stored in XML format and fetched from the server.
 - Behind-the-scenes data fetches using XMLHttpRequest objects in the browser.
 - JavaScript to make everything happen.

AJAX cannot work independently. It is used in combination with other technologies to create interactive webpages.

JavaScript

- Loosely typed scripting language.
- JavaScript function is called when an event occurs in a page.
- Glue for the whole AJAX operation.

DOM

- API for accessing and manipulating structured documents.
- Represents the structure of XML and HTML documents.

CSS

- Allows for a clear separation of the presentation style from the content and may be changed programmatically by JavaScript.

XMLHttpRequest

- JavaScript object that performs asynchronous interaction with the server.

Steps of AJAX Operation

1. A client event occurs.
2. An XMLHttpRequest object is created.
3. The XMLHttpRequest object is configured.
4. The XMLHttpRequest object makes an asynchronous request to the Webserver.
5. The Webserver returns the result containing XML document.
6. The XMLHttpRequest object calls the callback() function and processes the result.
7. The HTML DOM is updated.

1. A Client Event Occurs

- A JavaScript function is called as the result of an event.
- Example: `validateUserId()` JavaScript function is mapped as an event handler to an `onkeyup` event on input form field whose id is set to `"userid"`
- `<input type="text" size="20" id="userid" name="id" onkeyup="validateUserId();">`.

2. The XMLHttpRequest Object is Created

```
var ajaxRequest; // The variable that makes Ajax possible!
function ajaxFunction() {
    try{

        // Opera 8.0+, Firefox, Safari
        ajaxRequest = new XMLHttpRequest();
    }catch (e){

        // Internet Explorer Browsers
        try{
            ajaxRequest = new ActiveXObject("Msxml2.XMLHTTP");
        }catch (e) {

            try{
                ajaxRequest = new ActiveXObject("Microsoft.XMLHTTP");
            }catch (e){

                // Something went wrong
                alert("Your browser broke!");
                return false;
            }
        }
    }
}
```

3. The XMLHttpRequest Object is Configured

In this step, we will write a function that will be triggered by the client event and a callback function `processRequest()` will be registered.

```
function validateUserId() {
    ajaxFunction();

    // Here processRequest() is the callback function.
```

```

ajaxRequest.onreadystatechange = processRequest;

if (!target) target = document.getElementById("userid");
var url = "validate?id=" + escape(target.value);

ajaxRequest.open("GET", url, true);
ajaxRequest.send(null);
}

```

4. Making Asynchronous Request to the Webserver

Source code is available in the above piece of code. Code written in bold typeface is responsible to make a request to the webserver. This is all being done using the XMLHttpRequest object *ajaxRequest*.

```

function validateUserId() {
    ajaxFunction();

    // Here processRequest() is the callback function.
    ajaxRequest.onreadystatechange = processRequest;

    if (!target) target = document.getElementById("userid");
    var url = "validate?id=" + escape(target.value);

    ajaxRequest.open("GET", url, true);
    ajaxRequest.send(null);
}

```

Assume you enter *Zara* in the *userid* box, then in the above request, the URL is set to "validate?id=Zara".

5. Webserver Returns the Result Containing XML Document

You can implement your server-side script in any language, however its logic should be as follows.

- Get a request from the client.
- Parse the input from the client.
- Do required processing.
- Send the output to the client.

If we assume that you are going to write a servlet, then here is the piece of code.

```

public void doGet(HttpServletRequest request, HttpServletResponse
response) throws IOException, ServletException
{
    String targetId = request.getParameter("id");

    if ((targetId != null) && !accounts.containsKey(targetId.trim()))
    {
        response.setContentType("text/xml");
        response.setHeader("Cache-Control", "no-cache");
        response.getWriter().write("true");
    }
    else
    {
        response.setContentType("text/xml");
        response.setHeader("Cache-Control", "no-cache");
    }
}

```

```

        response.getWriter().write("false");
    }
}

```

6. Callback Function processRequest() is Called

The XMLHttpRequest object was configured to call the processRequest() function when there is a state change to the readyState of the XMLHttpRequest object. Now this function will receive the result from the server and will do the required processing. As in the following example, it sets a variable message on true or false based on the returned value from the Webserver.

```

function processRequest() {
    if (req.readyState == 4) {
        if (req.status == 200) {
            var message = ...;
        }
    }
}

```

7. The HTML DOM is Updated

This is the final step and in this step, your HTML page will be updated. It happens in the following way:

- JavaScript gets a reference to any element in a page using DOM API.
- The recommended way to gain a reference to an element is to call.

```

document.getElementById("userIdMessage"),
// where "userIdMessage" is the ID attribute
// of an element appearing in the HTML document

```

- JavaScript may now be used to modify the element's attributes; modify the element's style properties; or add, remove, or modify the child elements. Here is an example:

```

<script type="text/javascript">
<!--
function setMessageUsingDOM(message) {
    var userMessageElement = document.getElementById("userIdMessage");
    var messageText;

    if (message == "false") {
        userMessageElement.style.color = "red";
        messageText = "Invalid User Id";
    }
    else
    {
        userMessageElement.style.color = "green";
        messageText = "Valid User Id";
    }

    var messageBody = document.createTextNode(messageText);

    // if the messageBody element has been created simple
    // replace it otherwise append the new element
    if (userMessageElement.childNodes[0]) {
        userMessageElement.replaceChild(messageBody,

```

```

userMessageElement.childNodes[0]);
    }
    else
    {
        userMessageElement.appendChild(messageBody);
    }
}
-->
</script>
<body>
<div id="userIdMessage"><div>
</body>

```

3. Explain the concept of JSON concept with example.

JSON: JavaScript Object Notation.

- JSON is a syntax for storing and exchanging data.
- JSON is an easier-to-use alternative to XML.
- JSON is a lightweight data-interchange format
- JSON is language independent
- JSON is "self-describing" and easy to understand

The following JSON example defines an employees object, with an array of 3 employee records:

JSON Example

```

{"employees":[
  {"firstName":"John", "lastName":"Doe"},
  {"firstName":"Anna", "lastName":"Smith"},
  {"firstName":"Peter", "lastName":"Jones"}
]}

```

The following XML example also defines an employees object with 3 employee records:

XML Example

```

<employees>
  <employee>
    <firstName>John</firstName> <lastName>Doe</lastName>
  </employee>
  <employee>
    <firstName>Anna</firstName> <lastName>Smith</lastName>
  </employee>
  <employee>
    <firstName>Peter</firstName> <lastName>Jones</lastName>
  </employee>
</employees>

```

JSON Example

```
<!DOCTYPE html>
<html>
<body>

<h2>JSON Object Creation in JavaScript</h2>

<p id="demo"></p>

<script>
var text = '{"name":"John Johnson","street":"Oslo West 16","phone":"555 1234567"}';

var obj = JSON.parse(text);

document.getElementById("demo").innerHTML =
obj.name + "<br>" +
obj.street + "<br>" +
obj.phone;
</script>

</body>
</html>
```

Output:

JSON Object Creation in JavaScript

```
John Johnson
Oslo West 16
555 1234567
```

A common use of JSON is to read data from a web server, and display the data in a web page.

The following example reads a menu from **myTutorials.txt**, and displays the menu in a web page:

JSON Example

```
<div id="id01"></div>

<script>
var xmlhttp = new XMLHttpRequest();
var url = "myTutorials.txt";

xmlhttp.onreadystatechange = function() {
    if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
        var myArr = JSON.parse(xmlhttp.responseText);
        myFunction(myArr);
    }
}
```

```

    }
  }
  xmlhttp.open("GET", url, true);
  xmlhttp.send();

function myFunction(arr) {
  var out = "";
  var i;
  for(i = 0; i < arr.length; i++) {
    out += '<a href="' + arr[i].url + "'>' +
    arr[i].display + '</a><br>';
  }
  document.getElementById("id01").innerHTML = out;
}
</script>

```

Example Explained

1: Create an array of objects.

Use an **array literal** to declare an **array of objects**.

Give each object two properties: **display** and **url**.

Name the array **myArray**:

myArray

```

var myArray = [
{
  "display": "JavaScript Tutorial",
  "url": "http://www.w3schools.com/js/default.asp"
},
{
  "display": "HTML Tutorial",
  "url": "http://www.w3schools.com/html/default.asp"
},
{
  "display": "CSS Tutorial",
  "url": "http://www.w3schools.com/css/default.asp"
}
]

```

2: Create a JavaScript function to display the array.

Create a function **myFunction()** that loops the array objects, and display the content as HTML links:

myFunction()

```

function myFunction(arr) {
  var out = "";
  var i;
  for(i = 0; i < arr.length; i++) {
    out += '<a href="' + arr[i].url + "'>' + arr[i].display + '</a><br>';
  }
}

```

	<pre>document.getElementById("id01").innerHTML = out; } </pre> <p>Call myFunction() with myArray as argument:</p> <p>Example</p> <pre>myFunction(myArray);</pre> <p>3: Create a text file</p> <p>Put the array literal in a file named myTutorials.txt:</p> <p>myTutorials.txt</p> <pre>[{ "display": "JavaScript Tutorial", "url": "http://www.w3schools.com/js/default.asp" }, { "display": "HTML Tutorial", "url": "http://www.w3schools.com/html/default.asp" }, { "display": "CSS Tutorial", "url": "http://www.w3schools.com/css/default.asp" }]</pre> <p>4: Read the text file with an XMLHttpRequest</p> <p>Write an XMLHttpRequest to read the text file, and use myFunction() to display the array:</p> <p>XMLHttpRequest</p> <pre>var xmlhttp = new XMLHttpRequest(); var url = "myTutorials.txt"; xmlhttp.onreadystatechange = function() { if (xmlhttp.readyState == 4 && xmlhttp.status == 200) { var myArr = JSON.parse(xmlhttp.responseText); myFunction(myArr); } } xmlhttp.open("GET", url, true); xmlhttp.send();</pre>
4.	<p>Explain about Ajax Client Server Architecture.</p> <p>A Microsoft Ajax Web application consists of either a client-only solution or a client and server solution. A client-only solution uses Microsoft Ajax Library but does not use any ASP.NET server controls. For instance, an HTML can include script elements that reference the Microsoft Ajax Library .js files. The Microsoft Ajax Library allows Ajax applications to perform all processing on the client. A client and</p>

server solution consists of using both the Microsoft Ajax Library and ASP.NET server controls.

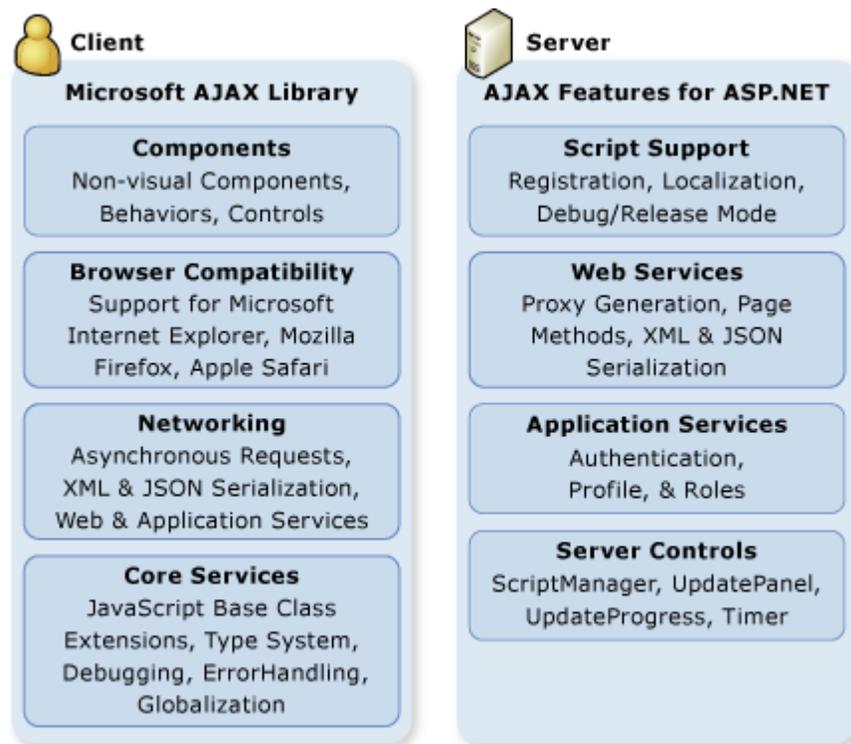


Figure: Microsoft Ajax client and server architecture

The illustration shows the functionality of the client-based Microsoft Ajax Library, which includes support for creating client components, browser compatibility, and networking and core services. The illustration also shows the functionality of server-based Microsoft Ajax features, which include script support, Web services, application services, and server controls. The following sections describe the illustration in more detail.

Microsoft Ajax Client Architecture

The client architecture includes libraries for component support, browser compatibility, networking, and core services.

Components

Client components enable rich behaviors in the browser without postbacks. Components fall into three categories:

- Components, which are non-visual objects that encapsulate code.
- Behaviors, which extend the behavior of existing DOM elements.
- Controls, which represent a new DOM element that has custom behavior.

The type of component that you use depends on the type of client behavior you want. For example, a watermark for an existing text box can be created by using a behavior that is attached to the text box

Browser Compatibility

The browser compatibility layer provides Microsoft Ajax scripting compatibility for the most frequently used browsers (including Microsoft Internet Explorer, Mozilla Firefox, and Apple Safari). This enables you to write the same script regardless of which supported browser you are targeting.

Networking

The networking layer handles communication between script in the browser and Web-based services and applications. It also manages asynchronous remote method calls. In many scenarios, such as partial-page updates that use the UpdatePanel control, the networking layer is used automatically and does not require that you write any code.

The networking layer also provides support for accessing server-based forms authentication, role information, and profile information in client script. This support is also available to Web applications that are not created by using ASP.NET, as long as the application has access to the Microsoft Ajax Library.

Core Services

The Ajax client-script libraries in ASP.NET consist of JavaScript (.js) files that provide features for object-oriented development. The object-oriented features included in the Microsoft Ajax client-script libraries enable a high level of consistency and modularity in client scripting. The following core services are part of the client architecture:

- Object-oriented extensions to JavaScript, such as classes, namespaces, event handling, inheritance, data types, and object serialization.
- A base class library, which includes components such as string builders and extended error handling.
- Support for JavaScript libraries that are either embedded in an assembly or are provided as standalone JavaScript (.js) files. Embedding JavaScript libraries in an assembly can make it easier to deploy applications and can help solve versioning issues. Debugging and Error Handling

The core services include the Sys.Debug class, which provides methods for displaying objects in readable form at the end of a Web page. The class also shows trace messages, enables you to use assertions, and lets you break into the debugger. An extended Error object API provides helpful exception details with support for release and debug modes.

Globalization

The Ajax server and client architecture in ASP.NET provides a model for localizing and globalizing client script. This enables you to design applications that use a single code base to provide UI for many locales (languages and cultures). For example, the Ajax architecture enables JavaScript code to format Date or Number objects automatically according to culture settings of the user's browser, without requiring a postback to the server.

Ajax Server Architecture

The server pieces that support Ajax development consist of ASP.NET Web server controls and components that manage the UI and flow of an application. The server pieces also manage serialization, validation, and control extensibility. There are also ASP.NET Web services that enable you to access ASP.NET application services for forms authentication, roles, and user profiles.

Script Support

Ajax features in ASP.NET are commonly implemented by using client script libraries that perform processing strictly on the client. You can also implement Ajax features by using server controls that support scripts sent from the server to the client.

You can also create custom client script for your ASP.NET applications. In that case, you can also use Ajax features to manage your custom script as static .js files (on disk) or as .js files embedded as resources in an assembly.

Ajax features include a model for release and debug modes. Release mode provides error checking and exception handling that is optimized for performance, with minimized script size. Debug mode provides more robust debugging features, such as type and argument checking. ASP.NET runs the debug versions when the application is in debug mode. This enables you to throw exceptions in debug scripts while

minimizing the size of release code.

Script support for Ajax in ASP.NET is used to provide two important features:

- The Microsoft Ajax Library, which is a type system and a set of JavaScript extensions that provide namespaces, inheritance, interfaces, enumerations, reflection, and additional features.
- Partial-page rendering, which updates regions of the page by using an asynchronous postback.

Localization

The Microsoft Ajax architecture builds on the foundation of the ASP.NET 2.0 localization model. It provides additional support for localized .js files that are embedded in an assembly or that are provided on disk. ASP.NET can serve localized client scripts and resources automatically for specific languages and regions.

Web Services

With Ajax functionality in an ASP.NET Web page, you can use client script to call both ASP.NET Web services (.asmx) and Windows Communication Foundation (WCF) services (.svc). The required script references are automatically added to the page, and they in turn automatically generate the Web service proxy classes that you use from client script to call the Web service.

You can also access ASP.NET Web services without using Microsoft Ajax server controls (for example, if you are using a different Web development environment). To do so, in the page, you can manually include references to the Microsoft Ajax Library, to script files, and to the Web service itself. At run time, ASP.NET generates the proxy classes that you can use to call the services.

Application Services

Application services in ASP.NET are built-in Web services that are based on ASP.NET forms authentication, roles, and user profiles. These services can be called by client script in an Ajax-enabled Web page, by a Windows client application, or by a WCF-compatible client.

Server Controls

Ajax server controls consist of server and client code that integrate to produce rich client behavior. When you add an Ajax-enabled control to an ASP.NET Web page, the page automatically sends supporting client script to the browser for Ajax functionality. You can provide additional client code to customize the functionality of a control, but this is not required.

The following list describes the most frequently used Ajax server controls.

Control	Description
ScriptManager	Manages script resources for client components, partial-page rendering, localization, globalization, and custom user scripts. The ScriptManager control is required in order to use the UpdatePanel, UpdateProgress, and Timer controls. However, the ScriptManager control is not required when creating a client-only solution.
UpdatePanel	Enables you to refresh selected parts of the page, instead of refreshing the whole page by using a synchronous postback.
UpdateProgress	Provides status information about partial-page updates in UpdatePanel controls.
Timer	Performs postbacks at defined intervals. You can use the Timer control to post the whole page, or use it together with the UpdatePanel control to perform partial-page updates at a defined interval.

You can also create custom ASP.NET server controls that include Ajax client behaviors. Custom controls that enhance the capabilities of other ASP.NET Web controls are referred to as extender controls.

Ajax Control Toolkit

The Ajax Control Toolkit contains controls that you can use to build highly responsive and interactive

	<p>Ajax-enabled Web applications. These controls do not require knowledge of JavaScript or Ajax. They are designed using concepts that are familiar to ASP.NET Web Forms application developers. Using the Ajax Control Toolkit, you can build Ajax-enabled ASP.NET Web Forms applications and ASP.NET MVC Web applications by dragging the controls from the Visual Studio Toolbox onto a page. The Ajax Control Toolkit is an open-source project that is part of the CodePlex Foundation</p>
5.	<p>Develop a web application for Airline Reservation System using AJAX. <u>Airline Reservation System using AJAX</u></p> <p>This example application simulates an online reservation system that allows users to search for the best flights available between two cities. The system processes requests and provides the flight numbers, seating information and price for each leg of a three-leg flight, computing the total price at the end. This Flight Reservation example application consists of two parts:</p> <ol style="list-style-type: none"> 1. Client-side Frontend Application A client-side application which accepts end user requests from an AJAX-based graphical user interface (GUI). This application enables users to select between one departure city and two possible destinations. 2. Server-side Backend Application A server-side application that processes end-user requests.
6.	<p>With a simple example illustrate the steps to create a java web service. (NOV/DEC 2012)</p> <p>Writing a java web service</p> <p>Currency conversion Service</p> <ul style="list-style-type: none"> ◆ Writing a server for a service using JWSDP 1.3 tools ◆ Application: currency converter <ul style="list-style-type: none"> ■ Three operations: <ul style="list-style-type: none"> ● fromDollars ● fromEuros ● fromYen ■ Input: value in specified currency ■ Output: object containing input value and equivalent values in other two currencies <p>Writing server software</p> <ol style="list-style-type: none"> 1. Write service endpoint interface <ul style="list-style-type: none"> • May need to write additional classes representing data structures 2. Write class implementing the interface 3. Compile classes 4. Create configuration files and run JWSDP tools to create web service 5. Deploy web service to Tomcat <p>service endpoint interface</p> <ul style="list-style-type: none"> ◆ The Web service endpoint interface is used to define the ‘Web services methods’. ◆ A Web service endpoint interface must conform to the rules of a JAX-RPC service definition interface. ◆ a service endpoint interface (SEI) that defines the interface of the web service. ◆ Configuration files are XML files that can be changed as needed. Developers can use configuration files to change settings without recompiling applications. Administrators can use configuration files to set policies that affect how applications run on their computers. ◆ config.xml : Defines the URL for WSDL file location. Each Web services has a corresponding WSDL (Web service Definition Language) document. <p>JWSDP: Server</p>

- ◆ **Rules for Service endpoint interface**

- Must extend java.rmi.Remote
 - declares a set of methods that may be invoked from a remote Java Virtual Machine(JVM)
- Every method must throw java.rmi.RemoteException
- Parameter/return value data types are restricted
- No public static final declarations (global constants) It must not have constant declarations

- ◆ **Allowable parameter/return value data types**

- Java primitives (int, boolean, etc.)
- Primitive wrapper classes (Integer, etc.)
- String, Date, Calendar, BigDecimal, BigInteger
- java.xml.namespace.QName, java.net.URI
- Struct: class consisting entirely of public instance variables
- Array of any of the above

- ◆ **Struct for currency converter app (data type for return values)**

```
package myCurCon;

public class ExchangeValues {
    public double dollars;
    public double euros;
    public double yen;
}
```

- ◆ **Service endpoint interface**

```
package myCurCon;

public interface CurCon extends java.rmi.Remote {
    public ExchangeValues fromDollars(double dollars)
        throws java.rmi.RemoteException;
    public ExchangeValues fromEuros(double euros)
        throws java.rmi.RemoteException;
    public ExchangeValues fromYen(double yen)
        throws java.rmi.RemoteException;
}
```

- ◆ Three files ExchangeValues.java, CurCon.java and CurConImpl.java written for the web service
- ◆ Class CurConImpl contains methods implements service endpoint interface, for *example*:

```
public ExchangeValues fromDollars(double dollars)
    throws java.rmi.RemoteException
{
    ExchangeValues ev = new ExchangeValues();
    ev.dollars = dollars;
    ev.euros = dollars * dollar2euro;
    ev.yen = dollars * dollar2yen;
    return ev;
}
```

Packaging server software

- ◆ Configuration file input to wscompile to create server

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration
  xmlns="http://java.sun.com/xml/ns/jax-rpc/ri/config">
  <service
    name="HistoricCurrencyConverter"
    targetNamespace="http://tempuri.org/wsdl"
    typeNamespace="http://tempuri.org/types"
    packageName="myCurCon">
    <interface name="myCurCon.CurCon" />
  </service>
</configuration>
```

- ◆ Configuration file for web service

```
<?xml version="1.0" encoding="UTF-8"?>
<webServices
  xmlns="http://java.sun.com/xml/ns/jax-rpc/ri/dd"
  version="1.0"
  targetNamespaceBase="http://tempuri.org/wsdl"
  typeNamespaceBase="http://tempuri.org/types">
```

- ◆ Configuration file for web service

```
urlPatternBase="/converter">
```

Context path

```
<endpoint
  name="CurrConverter"
  displayName="Currency Converter"
  description=
    "Converts between dollars, euros, and yen."
  interface="myCurCon.CurCon"
  model="/WEB-INF/model.xml.gz"
  implementation="myCurCon.CurConImpl"/>
```

Like
servlet
in
web.xml

```
<endpointMapping
  endpointName="CurrConverter"
  urlPattern="/currency" />
```

Like servlet-mapping
in web.xml

```
</webServices>
```

- ◆ Also need a minimal web.xml

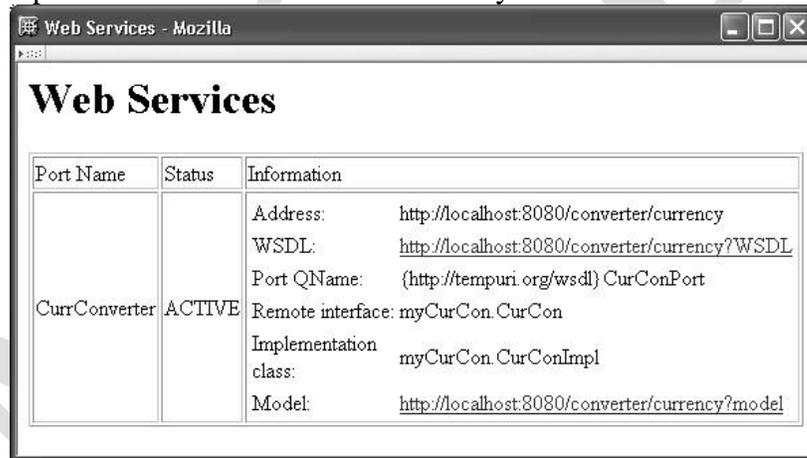
```
<web-app
  xmlns="http://java.sun.com/xml/ns/j2ee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
  version="2.4">
```

```
<display-name>Historic Currency Converter</display-name>
<description>
  This web service converts between three currencies using their
  exchange rates as of a fixed date.
</description>
```

```
</description>
```

```
</web-app>
```

- ◆ Run jar and wsdeploy to create a Web Archive (WAR) file converter.war
 - Name must match urlPatternBase value
- ◆ jaxrpc-ri.xml: Defines the various end points for referencing a Web service.
- ◆ *wscompile*: The wscompile tool generates stubs, and WSDL files used in JAX-RPC clients and services. The tool reads as input a configuration file and either a WSDL file or an RMI interface that defines the service.
- ◆ *wsdeploy*: Reads a WAR file (something like Jar file) and the jaxrpc-ri.xml file and then generates another WAR file that is ready for deployment
- ◆ Write service endpoint interface
 - May need to write additional classes representing data structures
- ◆ Write class implementing the interface
- ◆ Compile classes
- ◆ Create configuration files and run JWSDP tools to create web service
- ◆ Deploy web service to Tomcat
- ◆ Just copy converter.war to Tomcat webapps directory
 - May need to use Manager app to deploy
 - Enter converter.war in “WAR or Directory URL” text box
- ◆ Testing success:
 - Visit <http://localhost:8080/converter/currency>



7. Show the relationship between SOAP, UDDI, WSIL and WSDL

The following standards play key roles in Web services:

- Universal Description, Discovery and Integration (UDDI),
- Web Services Description Language (WSDL),
- Web Services Inspection Language (WSIL), SOAP, and
- Web Services Interoperability (WS-I).

The relationship between these standards is described in the following figure.

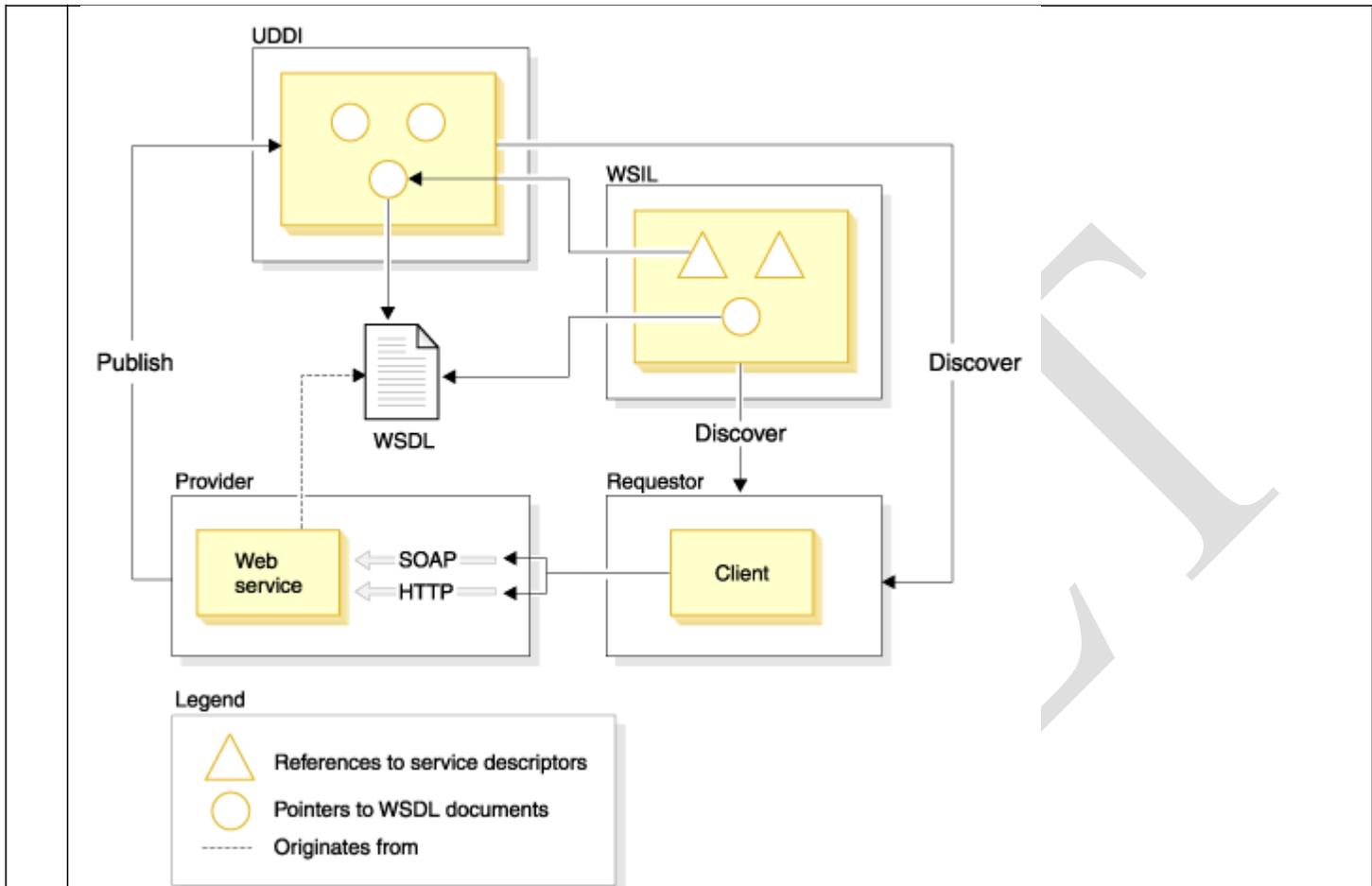


Figure: Relationships between SOAP, UDDI, WSIL and WSDL.

The UDDI specification defines open, platform-independent standards that enable businesses to share information in a global business registry, discover services on the registry, and define how they interact over the Internet.

WSIL (Web Services Inspection Language)

- XML-based open specification that defines a distributed service discovery method that supplies references to service descriptions at the service provider's point-of-offering, by specifying how to inspect a Web site for available Web services.
- A WSIL document defines the locations on a Web site where you can look for Web service descriptions.
- Since WSIL focuses on distributed service discovery, the WSIL specification complements UDDI by facilitating the discovery of services that are available on Web sites that may not be listed yet in a UDDI registry.

WSDL

- XML-based open specification that describes the interfaces to and instances of Web services on the network.

- It is extensible, so endpoints can be described regardless of the message formats or network protocols that are used to communicate.
- Businesses can make the WSDL documents for their Web services available through UDDI, WSIL, or by broadcasting the URLs to their WSDL via email or Web sites.

SOAP

- XML-based standard for messaging over HTTP and other Internet protocols.
- It is a lightweight protocol for the exchange of information in a decentralized, distributed environment.
- It is based on XML and consists of three parts:
 - An envelope that defines a framework for describing what is in a message and how to process it.
 - A set of encoding rules for expressing instances of application-defined data types.
 - A convention for representing remote procedure calls and responses.
- SOAP enables the binding and usage of discovered Web services by defining a message path for routing messages.
- SOAP may be used to query UDDI for Web services.

A service provider hosts a Web service and makes it accessible using protocols such as SOAP/HTTP or SOAP/JMS. The Web service is described by a WSDL document that is stored on the provider's server or in a special repository. The WSDL document may be referenced by the UDDI business registry and WSIL documents. These contain pointers to the Web service's WSDL files.

8. Explain the creation of a java web service Client in detail with examples. (MAY/JUNE 2012)

Writing a Java Web service Client

- ◆ Goal: write a JSP-based client
 - Input: currency and value
 - Output: table of equivalent values
- ◆ Use wscompile tool to develop client



Comparative Values	
Currency	Value
Dollars	\$59,034.34
Euros	€44,088.38
Yen	¥6,054,561.91

- ◆ Input xml document for wscompile tool
- ◆ Configuration file input to wscompile to create client
- ◆ Child element wsdl specifies the URL of a WSDL document

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration
  xmlns="http://java.sun.com/xml/ns/jax-rpc/ri/config">
  <wsdl
    location=
      "http://localhost:8080/converter/currency?WSDL"
    packageName="myCurConClient" />
</configuration>
```

- Wscompile generate proxy object
- Proxy object is a java object called on by a client software in order to access the web service

JWSDP: Client

- ♦ **Directory structure (wscompile generates content of classes and src)**
- ♦ **Run wscompile**

webapps

```
[[ other web application document base directories ]]
```

```
ConverterClient
  WEB-INF
    classes
      myCurConClient
    src
      myCurConClient
```

- ♦ **Run wscompile**

Wscompile -gen -keep -d classes -s src config.xml

- ♦ Wscompile tool creates a class implementing the interface
- ♦ Interface is shared between webservice server and clients via the wsdl document.
- ♦ On server side the class implementing the interface is written
- ♦ On client side the interface is automatically generated by wscompile tool

Structs will be represented as JavaBeans classes, regardless of how they are defined on the server

```
public class ExchangeValues {
  protected double dollars;
  protected double euros;
  protected double yen;
  ...
  public double getDollars() {
    return dollars;
  }

  public void setDollars(double dollars) {
    this.dollars = dollars;
  }
  ...
}
```

- ♦ Bean obtaining and calling proxy object:
- ♦ JSP document convert.jsp calls on javaBeans class to perform currency conversion and displays result in HTML table
- ♦ Document is placed in ConverterClient directory

```

public ExchangeValues getExValues() {
    ExchangeValues ev = null;
    CurCon curCon =
        (new HistoricCurrencyConverter_Impl()).getCurConPort();
    try {
        if (currency.equals("euros")) {
            ev = curCon.fromEuros(value);
        }
        else if (currency.equals("yen")) {
            ev = curCon.fromYen(value);
        }
        else {
            ev = curCon.fromDollars(value);
        }
    }
    catch (Exception e) {
        e.printStackTrace();
    }
    return ev;
}

```

◆ **JSP document using the bean:**

```

<c:set var="exvals" value="${client.exValues}" />
...
<tr>
    <td>Euros</td>
    <td style="text-align:right">
        <fmt:formatNumber
            type="currency" currencySymbol="&#8364;">
            ${exvals.euros}
        </fmt:formatNumber>
    </td>
</tr>

```

9. Describe Messaging protocol in web services with its functionalities.

SOAP is one of the more popular standards, and is one of the most significant standards in communicating web services over the network. XML provides a means for communicating over the Web using an XML document that both requests and responds to information between two disparate systems. SOAP allows the sender and the receiver of XML documents to support a common data transfer protocol for effective networked communication.

SOAP Building Blocks

A SOAP message is an ordinary XML document containing the following elements:

- A required Envelope element that identifies the XML document as a SOAP message
- An optional Header element that contains header information
- A required Body element that contains call and response information
- An optional Fault element that provides information about errors that occurred while processing the message

All the elements above are declared in the default namespace for the SOAP envelope:

<http://www.w3.org/2001/12/soap-envelope>

and the default namespace for SOAP encoding and data types is:

<http://www.w3.org/2001/12/soap-encoding>

Syntax Rules

Here are some important syntax rules:

- A SOAP message MUST be encoded using XML
- A SOAP message MUST use the SOAP Envelope namespace
- A SOAP message MUST use the SOAP Encoding namespace
- A SOAP message must NOT contain a DTD reference
- A SOAP message must NOT contain XML Processing Instructions

Skeleton SOAP Message

```
<?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-
envelope" soap:encodingStyle="http://www.w3.org/2001/12/soap-
encoding">
<soap:Header>
...
...
</soap:Header>
<soap:Body>
...
...
<soap:Fault>
...
...
</soap:Fault>
</soap:Body>
</soap:Envelope>
```

SOAP Elements:

SOAP Envelope Element

The mandatory SOAP Envelope element is the root element of a SOAP message.

The required SOAP Envelope element is the root element of a SOAP message. It defines the XML document as a SOAP message.

Note the use of the xmlns:soap namespace. It should always have the value of:

http://www.w3.org/2001/12/soap-envelope

and it defines the Envelope as a SOAP Envelope:

```
<?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-
envelope" soap:encodingStyle="http://www.w3.org/2001/12/soap-
encoding">
```

...
Message information goes here

```
...
</soap:Envelope>
```

The xmlns:soap Namespace

A SOAP message must always have an Envelope element associated with the "http://www.w3.org/2001/12/soap-envelope" namespace.

If a different namespace is used, the application must generate an error and discard the message.

The encodingStyle Attribute

The SOAP encodingStyle attribute is used to define the data types used in the document. This attribute may appear on any SOAP element, and it will apply to that element's contents and all child elements. A

SOAP message has no default encoding.

Syntax

```
soap:encodingStyle="URI"
```

SOAP Header Element

The optional SOAP Header element contains header information.

The optional SOAP Header element contains application specific information (like authentication, payment, etc) about the SOAP message. If the Header element is present, it must be the first child element of the Envelope element.

Note: All immediate child elements of the Header element must be namespace-qualified.

```
<?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-
envelope" soap:encodingStyle="http://www.w3.org/2001/12/soap-
encoding">
<soap:Header>
<m:Trans
xmlns:m="http://www.w3schools.com/transaction/"
soap:mustUnderstand="1">234</m:Trans>
</soap:Header>
...
...
</soap:Envelope>
```

The example above contains a header with a "Trans" element, a "mustUnderstand" attribute value of "1", and a value of 234.

SOAP defines three attributes in the default namespace ("http://www.w3.org/2001/12/soap-envelope"). These attributes are: actor, mustUnderstand, and encodingStyle. The attributes defined in the SOAP Header defines how a recipient should process the SOAP message.

The actor Attribute

A SOAP message may travel from a sender to a receiver by passing different endpoints along the message path. Not all parts of the SOAP message may be intended for the ultimate endpoint of the SOAP message but, instead, may be intended for one or more of the endpoints on the message path.

The SOAP actor attribute may be used to address the Header element to a particular endpoint.

```
Syntax soap:actor="URI"
```

Example

```
<?xml version="1.0"?>
```

The mustUnderstand Attribute

The SOAP mustUnderstand attribute can be used to indicate whether a header entry is mandatory or optional for the recipient to process.

If you add "mustUnderstand="1" to a child element of the Header element it indicates that the receiver processing the Header must recognize the element. If the receiver does not recognize the element it must fail when processing the Header.

```
Syntax soap:mustUnderstand="0|1"
```

SOAP Body Element

The mandatory SOAP Body element contains the actual SOAP message.
 The required SOAP Body element contains the actual SOAP message intended for the ultimate endpoint of the message.

Immediate child elements of the SOAP Body element may be namespace-qualified. SOAP defines one element inside the Body element in the default namespace ("http://www.w3.org/2001/12/soap-envelope"). This is the SOAP Fault element, which is used to indicate error messages.

```
<?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-envelope" soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
<soap:Body>
<m:GetPrice xmlns:m="http://www.w3schools.com/prices">
<m:Item>Apples</m:Item>
</m:GetPrice>
</soap:Body>
</soap:Envelope>
```

The example above requests the price of apples. Note that the m:GetPrice and the Item elements above are application-specific elements. They are not a part of the SOAP standard.

```
A SOAP response could look something like this: <?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-envelope" soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
<soap:Body>
<m:GetPriceResponse xmlns:m="http://www.w3schools.com/prices">
<m:Price>1.90</m:Price>
</m:GetPriceResponse>
</soap:Body>
</soap:Envelope>
```

SOAP Fault Element

The optional SOAP Fault element is used to hold error and status information for a SOAP message. An error message from a SOAP message is carried inside a Fault element.

If a Fault element is present, it must appear as a child element of the Body element. A Fault element can only appear once in a SOAP message.

The SOAP Fault element has the following sub elements:

Sub Element	Description
<faultcode>	A code for identifying the fault
<faultstring>	A human readable explanation of the fault
<faultactor>	Information about who caused the fault to happen
<detail>	Holds application specific error information related to the Body element

SOAP Fault Codes

	The faultcode values defined below must be used in the faultcode element when describing faults: Error	Description
	VersionMismatch	Found an invalid namespace for the SOAP Envelope element
	MustUnderstand	An immediate child element of the Header element, with the mustUnderstand attribute set to "1", was not understood
	Client	The message was incorrectly formed or contained incorrect information
	Server	There was a problem with the server so the message could not proceed

10. Explain the anatomy of UDDI.**UDDI is**

- A federated and open database
- A standard SOAP interface for querying and submitting information
- A repository that includes custom metadata for querying and finding Web services

UDDI enables companies to register their electronic services—everything from an e-mail address for technical support to XML-based Web services for purchasing. They can do this from a Web page or by programmatically using the UDDI interface.

There are two large pieces to UDDI, and one smaller piece. The larger pieces are:

- The data format
- The API for querying and submitting

More than two dozen data structures exist in the UDDI specification; therefore, covering them all would turn this book into a reference. What we should look at are the major data structures, and there are five very important ones, each with its own sub-elements:

- **businessEntity**— Contains information about a specific business, such as Microsoft.
- **businessService**— Contains information about a specific electronic service offered by a specific company.
- **tModel**— Categorizes a type. A service may contain multiple types.
- **bindingTemplate**— Collects tModels and other specific information about a service.
- **categoryBag**— Collects name–value pairs for a general categorization. **businessEntity**.

The following sections discuss **businessEntity**, **businessService**, and **tModel** in greater detail.

businessEntity

The **businessEntity** type holds information about specific businesses. Of course, there is no requirement that these documents precisely correspond to a single business entity. They could correspond to corporate entities, such as subsidiaries or corporate divisions.

This structure contains the following main pieces of information:

- **businessServices**— The service that this entity hosts
- **categoryBag**— A generic bag of names and values for properties that this entity has
- **identifierBag**— Similar to the category bag, a set of name–value pairs for generic information about the business entity
- **contacts**— A list of people to contact
- **discoveryURLs**— The location of documents that contain more information

businessService

The businessService document holds information about a specific service for a particular service. It contains the following data:

- serviceKey— A GUID that uniquely identifies the service
- name— A friendly, human-readable name for the service
- description— A human-readable description of the service and what it offers
- bindingTemplates— A set of properties that define the taxonomy of the service
- categoryBag— A generic name–value pair that helps to define the categories to which the service belongs

an example of a businessService document:

```
<businessService
  businessKey="some GUID here" serviceKey="some GUID here">
  <name>Keith's Service</name>
  <description>This service doesn't do anything</description>
  <bindingTemplate>
    ...
  </bindingTemplate>
</businessService>
```

tModel

The tModel is the hardest to understand and most misunderstood data structure in the UDDI Schema. It's a generic keyed reference to something.

Arguably, this incredible flexibility makes tModel harder to use. But it enables us to describe all kinds of things, and then link them to all kinds of other things. (Notice that I have to use the vague word thing because of this abstraction and flexibility.)

With UDDI, tModels can have many different purposes, but one stands out in my mind: to reference Web service features in a transparent and generic fashion. I can create separate tModels for services that implement transactions, that implement reliable messaging, and that are described by WSDLs. I can then attach all of the tModels that apply to any of my individual Web services. Even better, I can search for services based on those tModels!

The tModel structure contains the following information:

- tModelKey (technical model)
- name— The friendly, human-readable name of the service
- authorizedName
- operator
- description(s)— A description of the tModel
- Idbag— A set of identifying name–value pairs
- category bag— A series of categories to which this tModel belongs, expressed as a set of name–value pairs

The following [Listing](#) shows a sample of what a tModel can look like.

[Listing: A tModel Structure in the UDDI Schema](#)

```
<tModel
  xmlns="urn:uddi-org:api"
  tModelKey="UUID:1111111-1111-1111-1111-11111111">
  <name>KeithBa.Com:PurchaseOrders</name>
  <description xml:lang="en">
    Purchase orders services
  </description>
```

```

<overviewDoc>
  <overviewURL>http://keithba.com/po.wsdl</overviewURL>
</overviewDoc>
<categoryBag>
  <keyedReference
    tModelKey="UUID:C1ACF26D-9672-4404-9D70-39B756E62AB4"
    keyName="types"
    keyValue="wsdlSpec"/>
</categoryBag>
</tModel>

```

Programmer's API

In addition to defining a series of data structures, UDDI defines how to interact with those data structures—in other words, how to use them with SOAP. There are two major sets of APIs within UDDI:

- Inquiry Operations— For searching for information
- Publisher Operations— For saving, editing, and deleting information

Inquiry

The piece of UDDI used the most is the set of inquiry operations, often called the inquiry API. Inquiry operations with UDDI take two basic forms: browsing operations and retrieval operations.

Browsing operations are used to find something. You use them as broad-based queries to figure out what you want. All of these operations start with the pattern "find_XXX", where XXX is something specific, such as find_business or find_service.

There are only four operations in this category of UDDI operations:

- find_binding
- find_business
- find_service
- find_tModel

Once you know what you want, you can use the drill-down information retrieval operations to get all of the details you need about a specific business or service. These operations require specific UUIDs (Universally Unique Identifiers) that you probably will get via the browsing "find" operations. They all follow the pattern of "get_XXX," where XXX is the specific information you need. Using Microsoft's UDDI SDK, you can easily use these find operations:

```
Inquire.Url = "http://uddi.rte.microsoft.com/inquire";
```

```
FindBusiness findBusiness = new FindBusiness();
```

```
findBusiness.Names.Add("KeithBa");
```

```
BusinessList list = fb.Send();
```

There are five retrieval operations:

- get_bindingDetail
- get_businessDetail
- get_businessDetailExt
- get_serviceDetail
- get_tModelDetail

And again, the UDDI SDK from Microsoft makes it easy to call these:

```
Inquire.Url = "http://uddi.rte.microsoft.com/inquire";
```

```
FindBusiness findBusiness = new FindBusiness();
```

```
findBusiness.Names.Add("KeithBa, inc.");
```

```

BusinessList list = findBusiness.Send();

if (list.BusinessInfos.Count > 0)
{
    GetBusinessDetail gb = new GetBusinessDetail();
    gb.BusinessKeys.Add(bizList.BusinessInfos[0].BusinessKey);
    BusinessDetail bizDetail = gb.Send();
    if (bizDetail.BusinessEntities.Count > 0)
    {
        //do something interesting
    }
}

```

WSDL and UDDI

UDDI offers a way to store abstract WSDL documents (WSDLs that don't point to a specific service, but instead can be implemented by any number of services), as well. Actually, it provides for a specific tModel structure to which each businessService can point.

The basic idea is that it is possible to create WSDL documents that are abstract. Technically, almost any WSDL that is missing a service element and the child port pointing to a specific address is abstract. In practice, the UDDI binding is for abstract WSDLs that are described down through the binding section. Of course, nothing prevents you from using the true point of abstraction in a WSDL: the portType section.

Once you've defined this abstract WSDL (e.g., as part of a standards organization), you can then create a tModel in UDDI that references this WSDL. The important part to remember is that there must be a keyed-Reference to the tModelKey that represents abstract WSDLs.

Also, the overviewURL element should point to the WSDL file.

Listing: A tModel That References a WSDL Document

```

<tModel tModelKey="UUID:1111111-1111-1111-111111">
  <Name>Standard WSDL for AutoParts</Name>
  <OverviewDoc>
    <overviewURL>http://autoparts.org/autoparts.wsdl</overviewURL>
  </OverviewDoc>
  <categoryBag>
    <keyedReference
      tModelKey="uudi:C1ACF26D-9672-4404-9D70-39B756E62AB4"
      keyName="uddi-org:types"
      keyValue="wsdlSpec"
    />
  </categoryBag>
</tModel>

```

Now, we can reference this tModel from other tModels or (more likely) from businessService entries that represent specific implementations of this WSDL.

11. Explain the anatomy of WSDL.

WSDL

- **Web Service Description Language.**
- **An XML language used to *describe* and *locate* web services.**
 - **Written in XML.**

- Describe functionality of a web service
- Specify how to access the service (binding protocol, message format, and etc.)

Main Structure of WSDL

```
<definition namespace = "http/... ">
  <type> xschema types </type>
  <message> ... </message>
  <port> a set of operations </port>
  <binding> communication protocols </binding>
  <service> a list of binding and ports </service>
</definition>
```

A WSDL document can also contain other elements, like extension elements and a service element that makes it possible to group together the definitions of several web services in one single WSDL document.

A WSDL document describes a web service using these major elements:

Element	Defines
<types>	The data types used by the web service
<message>	The messages used by the web service
<portType>	The operations performed by the web service
<binding>	The communication protocols used by the web service

A WSDL document can also contain other elements, like extension elements, and a service element that makes it possible to group together the definitions of several web services in one single WSDL document.

Types

- <types> define types used in message declaration
- XML Schema, DTD, and etc.
- XML Schema must be supported by any vendor of WSDL conformant products.

WSDL Messages

- The <message> element defines the data elements of an operation.
- Each messages can consist of one or more parts. The parts can be compared to the parameters of a function call in a traditional programming language.

WSDL Ports

- The <portType> element is the most important WSDL element.
- It defines a **web service**, the **operations** that can be performed, and the **messages** that are involved.
- The <port> defines the connection point to a web service, an instance of <portType>.

It can be compared to a function library (or a module, or a class) in a traditional programming language. Each operation can be compared to a function in a traditional programming language

Operation Types

- The request-response type is the most common operation type, but WSDL defines four types:
 - **One-way**: The operation can receive a message but will not return a response
 - **Request-response**: The operation can receive a request and will return a response
 - **Solicit-response**: The operation can send a request and will wait for a response
 - **Notification**: The operation can send a message but will not wait for a response

Binding

- Binding defines how message are transmitted, and the location of the service.

WSDL Example

	<p>This is a simplified fraction of a WSDL document:</p> <pre><message name="getTermRequest"> <part name="term" type="xs:string"/> </message> <message name="getTermResponse"> <part name="value" type="xs:string"/> </message> <portType name="glossaryTerms"> <operation name="getTerm"> <input message="getTermRequest"/> <output message="getTermResponse"/> </operation> </portType></pre> <p>In this example the <portType> element defines "glossaryTerms" as the name of a port, and "getTerm" as the name of an operation. The "getTerm" operation has an input message called "getTermRequest" and an output message called "getTermResponse". The <message> elements define the parts of each message and the associated data types.</p>
12.	<p>Describe the major elements of SOAP. (NOV/DEC 2011, MAY/JUNE 2014) (APR/MAY 2013)</p> <p>SOAP is a simple XML based protocol to let applications exchange information over HTTP. Or more simply: SOAP is a protocol for accessing a Web Service.</p> <p>SOAP Building Blocks</p> <p>A SOAP message is an ordinary XML document containing the following elements:</p> <ul style="list-style-type: none"> • A required Envelope element that identifies the XML document as a SOAP message • An optional Header element that contains header information • A required Body element that contains call and response information • An optional Fault element that provides information about errors that occurred while processing the message <p>All the elements above are declared in the default namespace for the SOAP envelope: http://www.w3.org/2001/12/soap-envelope and the default namespace for SOAP encoding and data types is: http://www.w3.org/2001/12/soap-encoding</p> <p>Syntax Rules</p> <p>Here are some important syntax rules:</p> <ul style="list-style-type: none"> • A SOAP message MUST be encoded using XML • A SOAP message MUST use the SOAP Envelope namespace • A SOAP message MUST use the SOAP Encoding namespace • A SOAP message must NOT contain a DTD reference • A SOAP message must NOT contain XML Processing Instructions <p>Skeleton SOAP Message</p> <pre><?xml version="1.0"?> <soap:Envelope</pre>

```

xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
<soap:Header>
...
...
</soap:Header>
<soap:Body>
...
...
<soap:Fault>
...
...
</soap:Fault>
</soap:Body>
</soap:Envelope>

```

SOAP Elements:

SOAP Envelope Element

The mandatory SOAP Envelope element is the root element of a SOAP message.

The required SOAP Envelope element is the root element of a SOAP message. It defines the XML document as a SOAP message.

Note the use of the xmlns:soap namespace. It should always have the value of:

http://www.w3.org/2001/12/soap-envelope

and it defines the Envelope as a SOAP Envelope:

```
<?xml version="1.0"?>
```

```
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-
envelope" soap:encodingStyle="http://www.w3.org/2001/12/soap-
encoding">
```

```
...
Message information goes here
```

```
...
</soap:Envelope>
```

The xmlns:soap Namespace

A SOAP message must always have an Envelope element associated with the "http://www.w3.org/2001/12/soap-envelope" namespace.

If a different namespace is used, the application must generate an error and discard the message.

The encodingStyle Attribute

The SOAP encodingStyle attribute is used to define the data types used in the document. This attribute may appear on any SOAP element, and it will apply to that element's contents and all child elements. A SOAP message has no default encoding.

Syntax

```
soap:encodingStyle="URI"
```

SOAP Header Element

The optional SOAP Header element contains header information.

The optional SOAP Header element contains application specific information (like authentication,

payment, etc) about the SOAP message. If the Header element is present, it must be the first child element of the Envelope element.

Note: All immediate child elements of the Header element must be namespace-qualified.

```
<?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-
envelope" soap:encodingStyle="http://www.w3.org/2001/12/soap-
encoding">
<soap:Header>
<m:Trans
xmlns:m="http://www.w3schools.com/transaction/"
soap:mustUnderstand="1">234</m:Trans>
</soap:Header>
...
...
</soap:Envelope>
```

The example above contains a header with a "Trans" element, a "mustUnderstand" attribute value of "1", and a value of 234.

SOAP defines three attributes in the default namespace ("http://www.w3.org/2001/12/soap-envelope"). These attributes are: actor, mustUnderstand, and encodingStyle. The attributes defined in the SOAP Header defines how a recipient should process the SOAP message.

The actor Attribute

A SOAP message may travel from a sender to a receiver by passing different endpoints along the message path. Not all parts of the SOAP message may be intended for the ultimate endpoint of the SOAP message but, instead, may be intended for one or more of the endpoints on the message path.

The SOAP actor attribute may be used to address the Header element to a particular endpoint.

```
Syntax soap:actor="URI"
```

Example

```
<?xml version="1.0"?>
```

The mustUnderstand Attribute

The SOAP mustUnderstand attribute can be used to indicate whether a header entry is mandatory or optional for the recipient to process.

If you add "mustUnderstand="1" to a child element of the Header element it indicates that the receiver processing the Header must recognize the element. If the receiver does not recognize the element it must fail when processing the Header.

```
Syntax soap:mustUnderstand="0|1"
```

SOAP Body Element

The mandatory SOAP Body element contains the actual SOAP message.

The required SOAP Body element contains the actual SOAP message intended for the ultimate endpoint of the message.

Immediate child elements of the SOAP Body element may be namespace-qualified. SOAP defines one element inside the Body element in the default namespace ("http://www.w3.org/2001/12/soap-envelope"). This is the SOAP Fault element, which is used to indicate error messages.

```
<?xml version="1.0"?>
<soap:Envelope
```

```

xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
<soap:Body>
<m:GetPrice xmlns:m="http://www.w3schools.com/prices">
<m:Item>Apples</m:Item>
</m:GetPrice>
</soap:Body>
</soap:Envelope>
    
```

The example above requests the price of apples. Note that the m:GetPrice and the Item elements above are application-specific elements. They are not a part of the SOAP standard.

```

A SOAP response could look something like this: <?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-
envelope" soap:encodingStyle="http://www.w3.org/2001/12/soap-
encoding">
<soap:Body>
<m:GetPriceResponse xmlns:m="http://www.w3schools.com/prices">
<m:Price>1.90</m:Price>
</m:GetPriceResponse>
</soap:Body>
</soap:Envelope>
    
```

SOAP Fault Element

The optional SOAP Fault element is used to hold error and status information for a SOAP message. An error message from a SOAP message is carried inside a Fault element.

If a Fault element is present, it must appear as a child element of the Body element. A Fault element can only appear once in a SOAP message.

The SOAP Fault element has the following sub elements:

Sub Element	Description
<faultcode>	A code for identifying the fault
<faultstring>	A human readable explanation of the fault
<faultactor>	Information about who caused the fault to happen
<detail>	Holds application specific error information related to the Body element

SOAP Fault Codes

The faultcode values defined below must be used in the faultcode element when describing faults: Error	Description
VersionMismatch	Found an invalid namespace for the SOAP Envelope element
MustUnderstand	An immediate child element of the Header element, with the mustUnderstand attribute set to "1", was

		not understood
	Client	The message was incorrectly formed or contained incorrect information
	Server	There was a problem with the server so the message could not proceed

EMNCET

V.S.B ENGINEERING COLLEGE, KARUR

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ASSIGNMENT TOPICS

SEM/ YEAR / SEC: V / III / B

S.NO	REGSITER NUMBER	STUDENT NAME	ASSIGNMENT TOPICS
1.	922516104001	AARTHI K	Java - Serialization
2.	922516104002	AISWARYA .R	Java - Networking
3.	922516104003	AJITH V	Java –AWT(Abstract Window Toolkit)
4.	922516104004	AKASH J	Java - Swing
5.	922516104005	AMBIKA M	Java –Layout Manager
6.	922516104006	ANAND GANESAN S	Java - Reflection
7.	922516104007	ARUN G	Java - Conversion
8.	922516104009	ARUN PRAKASH P	Java - Collection
9.	922516104008	ARUNKUMAR J	Java - FX
10.	922516104010	AZHARUDEEN S	Web Browser Types
11.	922516104012	BHARANIDHARAN J	Web Server Types
12.	922516104015	DEEPA .R	HTML5- Web Forms 2.0
13.	922516104016	DHAMAYANTHI R	HTML 5 - SVG
14.	922516104017	DHANASEKARAN M	HTML5- Geolocation
15.	922516104018	DHARANIPRIYA K	HTML5 – Web Storage
16.	922516104019	DHILEEBAN P	HTML5 – SQL Database
17.	922516104020	DHINESH M	CSS3 – 2D Transform
18.	922516104022	GLADIS SMERNA P	CSS3 – 3D Transform
19.	922516104023	GOWSIK V	CSS3 - Animation
20.	922516104024	GOWTHAM R	CSS3 – Box Sizing

21.	922516104025	GURUSARAN L	Java Script – Error Handling
22.	922516104026	HAMEEDA BANU S	Java Script – Animation
23.	922516104027	HARIVIVEK M	Java – Debugging
24.	922516104028	HARSHAVARDINI R	Servlet – Writing files
25.	922516104029	JAGATHAMBAL M	Servlet – DataBaseAccess
26.	922516104030	JEGADESAN S	Servlet – Debugging
27.	922516104031	KANIMOZHI M	Servlet – Annotations
28.	922516104032	KANNAN E	JSP – Database Access
29.	922516104033	KANNAN V	JSP –Expression Language
30.	922516104034	KARPAGAVALLI S	JSP – Filters
31.	922516104035	KARTHIKEYAN K	JSP – Java Beans
32.	922516104036	KATHIRESAN R	PHP – Web Concepts
33.	922516104037	KAVIN S	PHP – Session
34.	922516104038	KAVIN KUMAR P	PHP – GET&POST
35.	922516104039	KAVYA V	PHP – Coding Standard
36.	922516104040	KIRUTHIKA K	PHP – File Inclusion
37.	922516104041	KIRUTHIKA S	XML – Encoding
38.	922516104042	KISHORE SM	XML –Character Entities
39.	922516104043	KOWSALYA G	XSLT – Message
40.	922516104044	KOWSHIKA K	XSLT – Key
41.	922516104045	LAVANYA S	AJAX – Action
42.	922516104046	MADHAVAN N	AJAX – Security Issues
43.	922516104047	MATHUMITHA A	AJAX – Database Operations
44.	922516104048	MAHENDRAN.V	Web services standards & Security
45.	922516104049	MAHENDRA PRASANTH D G	Web services Components & Example

46.	922516104050	MANIKANDAN.C	JDBC – Driver Types
47.	922516104051	MANIKANDAN J	JDBC – Result Sets & Exceptions
48.	922516104053	MANJU C	JDBC – Datatypes
49.	922516104054	NAGAJOTHI M	JDBC – Transactions
50.	922516104055	NANDHINI .R	JDBC – Batch Processing
51.	922516104057	NAVANEEDHAKRISHNAN D	JDBC – SQL Syntax
52.	922516104058	NETHAJI M	JAVA – Modifier Types
53.	922516104059	NITHIYAPRAKASH B	Java –Data Structure
54.	922516104060	NIVETHA R	Java – Collections

Class Advisor

HOD