T.Y. Diploma : Sem. VI

[IF/CM]

Object Oriented Modeling and Design

Time: 3 Hrs.] Prelim Question Paper Solution **[Marks : 100**

Q.1 Attempt any FIVE of the following [20]

Q.1(a) Explain four stages of OMT by Rumbaugh. [4]

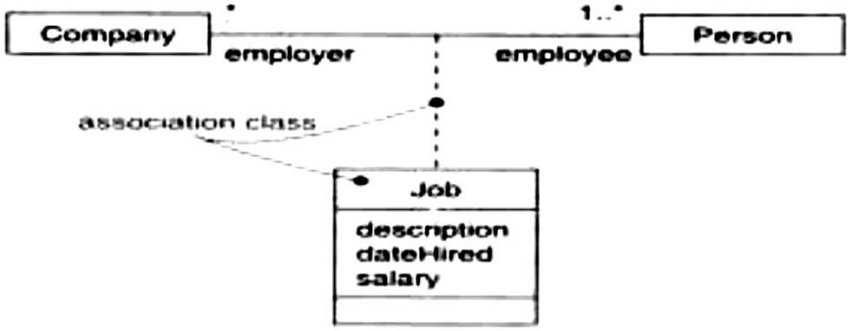
Ans.: Object Modeling Technique (OMT) by Rumbaugh includes four stages: [1 mark each]

1. **Analysis:** Starting from a statement of the problem, the analyst builds a model of the real-world situation showing its important properties. The analyst works with the requestor to understand the problem statement. The analysis model is a concise, precise abstraction of what the desired system must do, not how it will be done. A good model can be understood and criticized by application experts who are not programmers. The analysis model does not contain any implementation details.
2. **System Design:** System designer makes high level decisions about the overall architecture. During system design, the target system is organized into subsystems based on both the analysis structure and the proposed architecture. The system designer decides what performance characteristics to optimize, choose a strategy of attacking the problem and make tentative resource allocations.
3. **Object Design:** The object designer builds a design model based on the analysis model but contains implementation details. The designer adds details to the design model in accordance with the strategy established during system design. The focus of object design is the data structures and algorithms needed to implement each class.
4. **Implementation:** The object classes and relationships developed during object design are finally translated into a particular programming language, database or hardware implementation.

Q.1(b) Explain link attribute and association as a class with example [4]

Ans.: Link attribute [2 marks]

Link attribute specify properties of link/association between two classes/objects. For example: In the below example company and person has a link/association between them which has link attributes as description, datehired and salary. These attributes are placed inside the association class job. Association class is linked to association line with dashed line.



Association as a class: [2 marks]

- In an association between two classes, the association itself might have properties. Just as an object of a class is describe with attributes, the of an association can also have a description.
- For example, in an employer/ employee relationship between a Company and a Person there is a job that represents the properties of that relationship that to exactly one pairing of the Person and Company. It would not be appropriate to model this situation with a Company to job association together with a job to person association. That would not tie a specific instance of the job to the specific paring of Company and Person.

- These can be model as an association class, which is a modeling element that has both association and class properties.
  An association class can be seen as an association that also has class properties, or as a class that also has association properties.
  An association class as a class symbol attached by a dashed line to an association can be shown in figure.
- Sometimes it is the same properties for several different association classes can be used. However, we cannot attach an association class to more than one association, since an association class is the association itself. To achieve that effects, define a class© and then have each association class that needs those features inherit from C or use C as the type of An attribute.

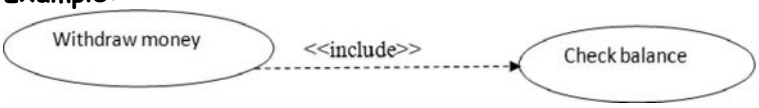
Q.1(c) Explain <<include>> and <extend>> with example **[4]**

Ans.: <<include» relationships [2 marks]

Include relationship is used to include one use case within the behavior sequence of another use case.

- An include relationship between use cases means that the base case explicitly incorporates the behavior of another use case at a location specified in the base.
- The include use case never stand alone. When an actor initiates any base use case then base use case executes included use case.
- An include relationship as a dependency can be render with stereotyped as include. To specify the location in a flow of events in which the base use case includes the behavior of another, write include followed by the name of the use case.
- Arrow is placed near the included use case. Arrow is directed from base use case to included use case.

Example:

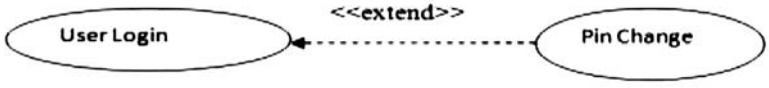


«extend» relationships [2 marks]

It adds incremental behavior of an use case.

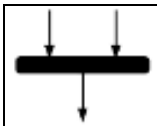
- A extend relationship between use cases means that the base use case implicitly incorporates the behavior of another use case at a location specified indirectly by the extending use case.
- The extended use case adds itself to the base use case. Most of the time, an extend relationship has a condition attached to it. The extended use case executes only when the condition is true.
- The base use case may stand alone, but under certain conditions, its behavior may be extended by behavior of another use case.
- An extend relationship as a dependency can be render with stereotyped as extend. Arrow is directed from extended use case towards base use case.

Example:

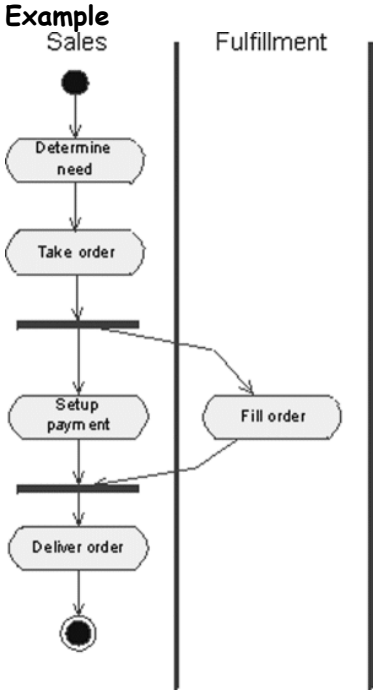


Q.1(d) Explain Synchronization Bar. **[4]**

Ans.: [Explanation - 4 marks]

	Join symbol/ Synchronization bar	Combines two concurrent activities and re-introduces them to a flow where only one activity occurs at a time. Represented with a thick vertical or horizontal line.
	Fork symbol	Splits a single activity flow into two concurrent activities. Symbolized with multiple arrowed lines from a join.

Example



Q.1(e) Explain with diagram how to create and destroy messages. **[4]**

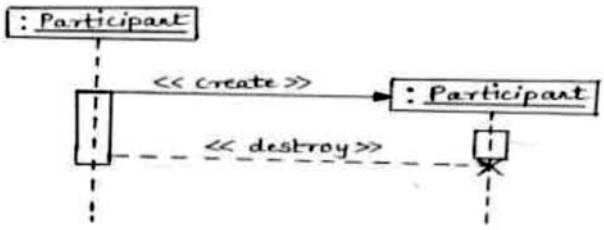
Ans.: **Create message:** [Description of create message & destroy message with diagram – 2 marks each]

1. Objects can be created according to the requirement of the system in between the processing of the system because they are not required for the entire duration of the sequence diagrams interaction.
2. If an object does not exist at the beginning of a sequence diagram then it must be created in the system.
3. The UML shows creation by placing the object notation at the head of the arrow for the message call that creates an object.

Destroy message:

1. An object can destroy itself or it can be destroyed by other objects of the sequence diagram because those objects may not further require during the system.
2. If the object is destroyed by itself then "X" is placed at the tail of the line and arrow head is towards another object to which it passes the control.
3. If the object is destroyed by another object then a destroy message is send by another object from the system. In this case the large "X" is placed at the head of the return arrow.

Example



Q.1(f) Explain uses of deployment diagram **[4]**

Ans.: **Usage of deployment diagrams can be described as follows :** [Four uses - 4 marks]

- To model the hardware topology of a system.
- To model embedded system.
- To model hardware details for a client/server system.
- To model hardware details of a distributed application.
- Forward and reverse engineering.

Q.2 Attempt any FOUR of the following : [16]

Q.2(a) Explain unified software development life cycle. [4]

Ans.: [Explanation - 4 marks]

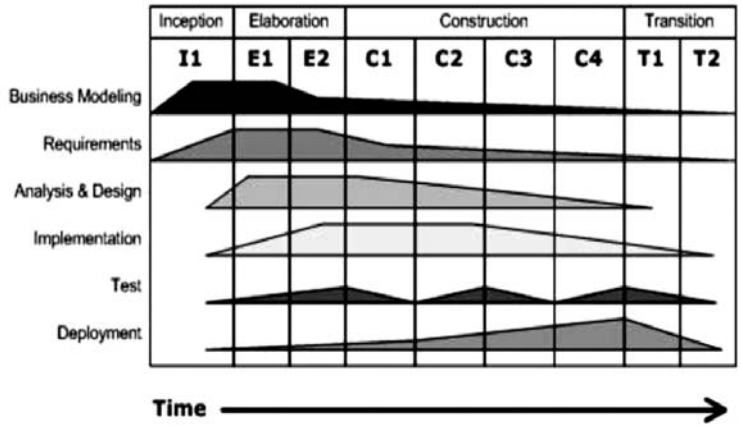


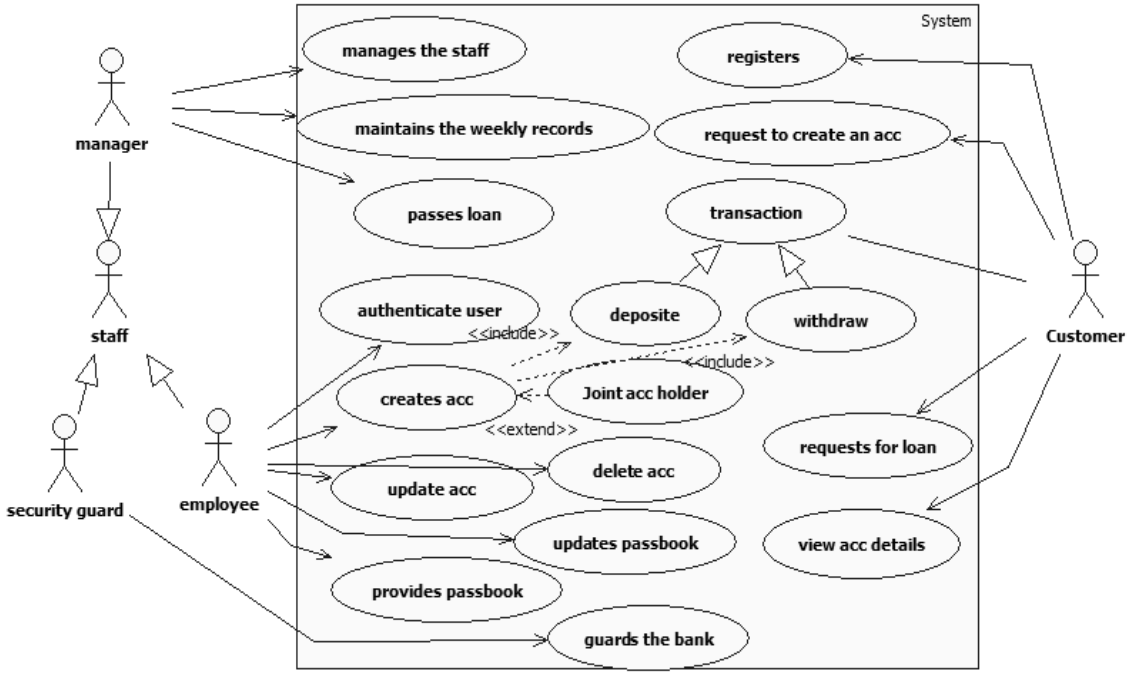
Fig: Software development lifecycle (SDLC)

There are four phases of SDLC:

1) **Inception:** It is first phase of SDLC where the idea for the development of a system is finalized.
2) **Elaboration:** It is second phase of process where the product vision and its architecture is defined. In this phase the systems requirements are collected with respect to vision statement, evaluation criteria, functional and non-functional behavior and testing of the processes.
3) **Construction:** It is a third phase of the SDLC where the project is implemented to handover to the user community the systems requirements and its evaluation criteria are constantly reexamined against the project needs in this phase the resources are allocated to reduce risk in the project after implementation.
4) **Transition:** It is a fourth phase of process where the software is handed over to user community in this phase the software is continuously in improvement the bugs are detected and solved and if required new released of the project are developed.

Q.2(b) Draw use case diagram for banking application. [4]

Ans.: [Diagram - 4 marks]

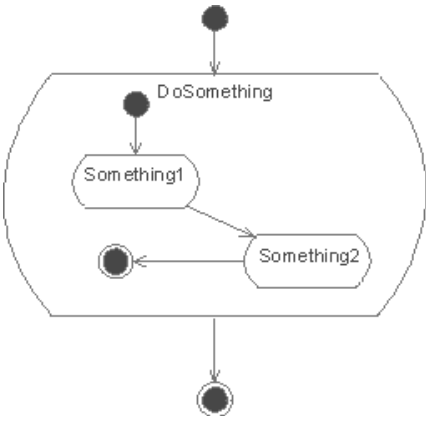


Q.2(c) What is importance of nested activity diagram? [4]

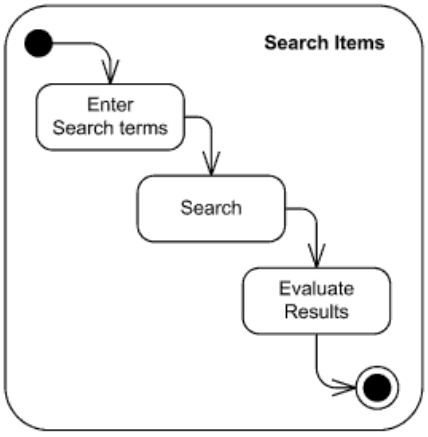
Ans.: Nested Activity diagram [Explanation - 2 marks, Suitable example – 2 marks]

1. Nested activity diagram contains an activity diagram inside an activity.
2. Nested activity state may reference another activity diagram that shows the internal structure of the activity state.
3. It shows the subgraph (subactivity) inside of the activity state which refers to another diagram. Subactivity can be reused as independent activity.
4. Nested activity diagram gives the result of its activities to the activity in which it resides.

Structure:



Example:



Q.2(d) Define constraints. How they are applied? Give example. [4]

Ans.: Definition [Definition - 2 marks]

Constraint is a Boolean condition involving model elements Definition such as objects, classes, attributes, links and associations. A constraint restricts the values that entities/elements can assume.

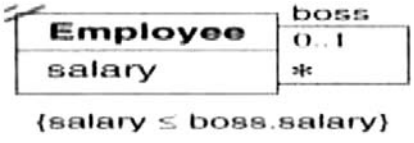
[Description any one application with example - 2 marks]

Notation:

Constraint is shown as text written inside curly bracket.

{Constraint}

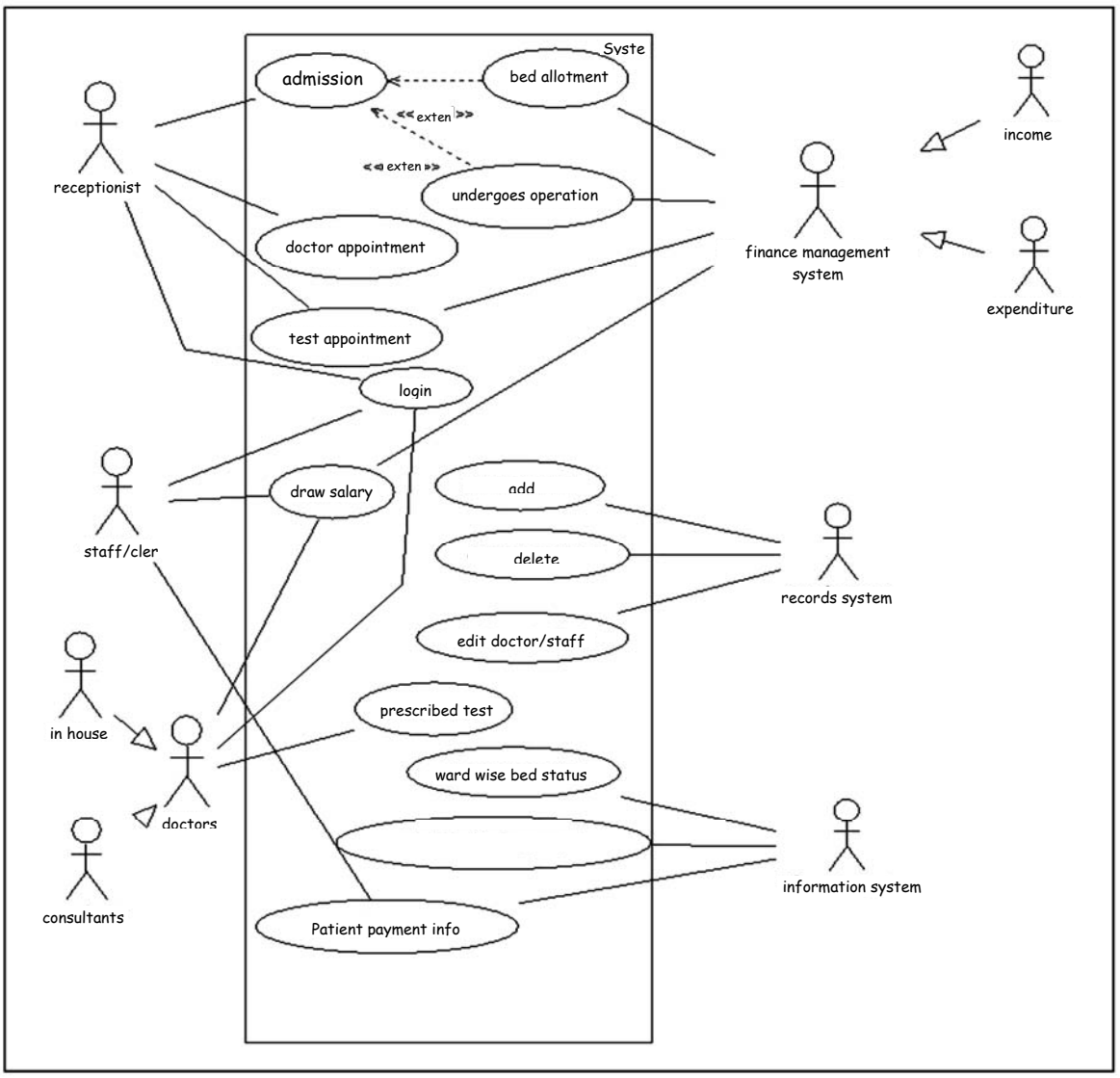
Example: Constraints on Objects:



In the above example a constraint is specified as salary<=boss.salary inside curly brackets. It restricts the value of attribute salary of employees with respect salary of boss.

Q.2(e) Draw class diagram for hospital management system **[4]**

Ans.: [Diagram - 4 marks]



Q.2(f) What are the principles of Modeling? **[4]**

Ans.: Four basic principles of modeling : [1 mark each]

(i) The choice of what models to create has a profound influence on how a problem is attacked and how a solution is shaped.
In other words, choose your models well. The right models will brilliantly illuminate the most wicked development problems, offering insight that you simply could not gain otherwise; the wrong models will mislead you, causing you to focus on irrelevant issues.

(ii) Every model may be expressed at different levels of precision.
If you are building a high rise, sometimes you need a 30,000-foot view• for instance, to help your investors visualize its look and feel.

(iii) The best models are connected to reality.
A physical model of a building that doesn't respond in the same way as do real materials has only limited value; a mathematical model of an aircraft that assumes only ideal conditions and perfect manufacturing can mask some potentially fatal characteristics.

(iv) No single model is sufficient. Every nontrivial system is best approached through a small set of nearly independent models.
If you are constructing a building, there is no single set of blueprints that reveal all its details. At the very least, you'll need floor plans, elevations, electrical plans, heating plans, and plumbing plans.

Q.3 Attempt any FOUR of the following : [16]

Q.3(a) Explain following terms : [4]

(i) Class **(ii) Role name**

(iii) Qualified association **(iv) Order association**

Ans.: (i) Class: [1 mark]

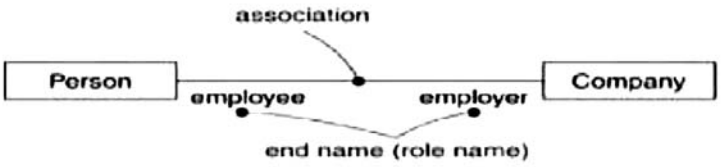
A Class is a group of objects with similar properties (attributes), common behavior (operation), common relationship to other objects and common semantics.

Example: fruit, student, employee etc.

ClassName
attribute attribute : DataType[attMult] attribute : DataType[attMult) = defaultValue ...
operation operation (arg1: Name1,. ..) : ResultType ...

(ii) Role names: [1 mark]

A role name is a name that uniquely identifies one end of an association. It specifies a role of an object of a class which it plays in the association. A role name is written next to the association line near the class that plays the role. Example:-



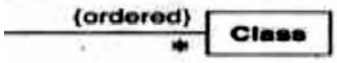
(iii) Qualified association: [1 mark]

Qualified association specifies relation between two object classes and a qualifier. The qualifier is a special attribute that reduces the effective multiplicity of an association. The qualifier distinguishes among the set of objects at the many end of an association. A qualifier is drawn as a small box on the end of the association line near the class it qualifies. Notation:

Qualified Association:

(iv) Ordered association: [1 mark]

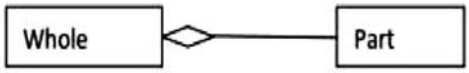
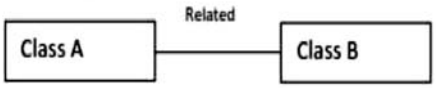
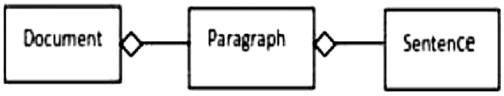
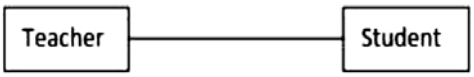
Usually the objects on the "many" side of an association have no explicit order, and can be regarded as a set. Sometimes the objects on the many side of an association have order. Writing {ordered} next to the multiplicity dot indicates an ordered set of objects of an association. Notation:



Q.3(b) Differentiate between association and aggregation. [4]

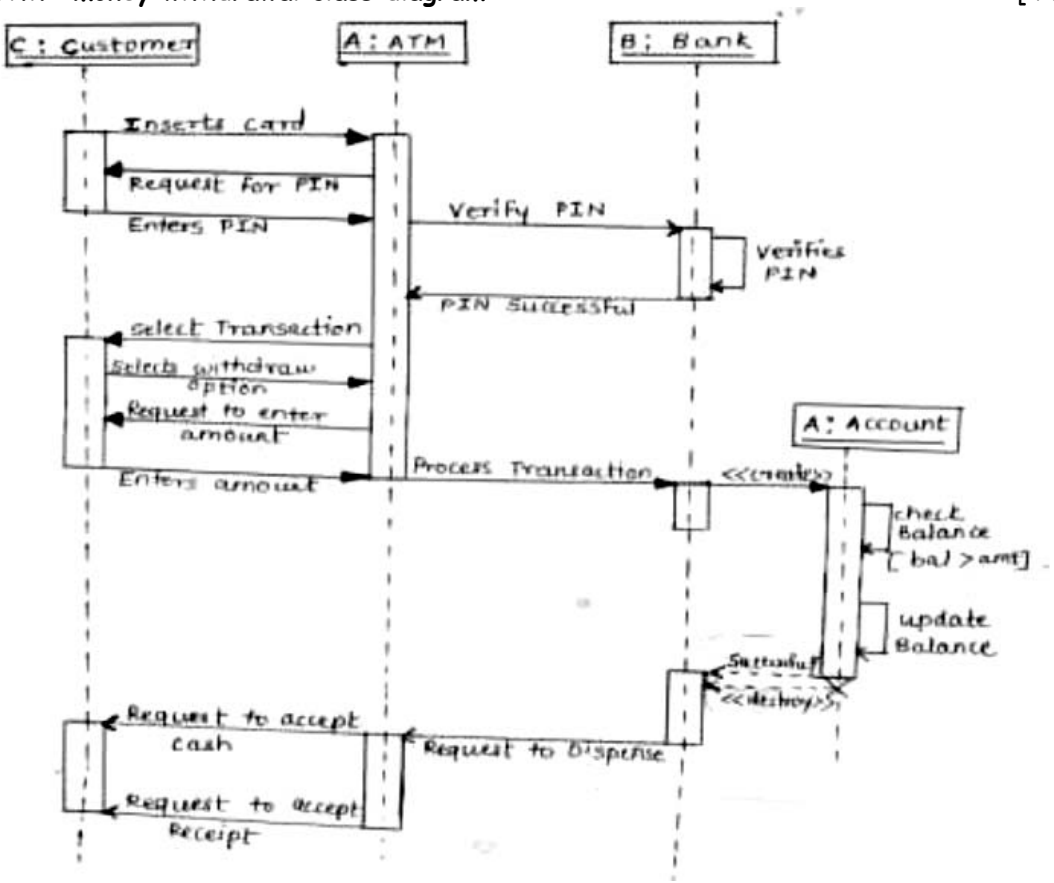
Ans.: [Any four point - 4 marks]

	Aggregation	**Association**
1.	Aggregation is the "Part-whole" or "a-part-of" relationship in which objects representing the components.	Association describe q group of links with common structure & common semantics.
2.	Aggregation is drawn like association, except a small diamond indicates the assembly end of the relationship.	A link is an instance of an association.

3.	**Notation:** 	**Notation:** 
4.	An Aggregation is a specialized association.	An Association defines a relationship between two or more classes.
5.	Aggregation is tightly looped from of association with same extra semantics.	Association represents static relationship between classes.
6.	Two types of Aggregation: aggregation & composition	Two types of association Binary & n-ary.
7.	**Example:** 	**Example:** 
8.	Example shows that a document consists of many sentences.	Example shows the object model for Teacher & student associations.

Q.3(c) Draw at Sequence diagram for ATM session **[4]**

Ans.: ATM -money withdrawal class diagram [4 marks]

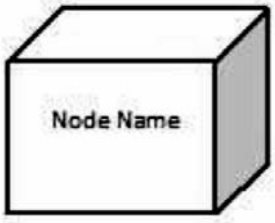


Q.3(d) Describe notations used in deployment diagram. **[4]**

Ans.: Notations: [Any four notations - 1 mark each]

1. **Node**

   A node is physical element that exists at runtime & represents a computation resource with some memory and processing capability nodes can be a server, printer, cash dispenser etc.

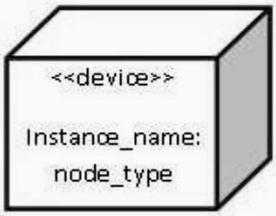


2. **Communication line-Association**
   Communication line is used to connect 2 nodes or nodes with other devices. Communication lines specify 2 types of relationship for connecting to either a node or to the component.
   Association is used to show relationship between 2 nodes. It is shown with a solid line.

3. **Communication line-dependency**
   It is used to show relationship between node and a component. A component is placed inside the node to provide processing capability to the node. A node depends on the component. Dependency is shown with dashed line and a arrow head. It connects node with the component arrow head points towards component.

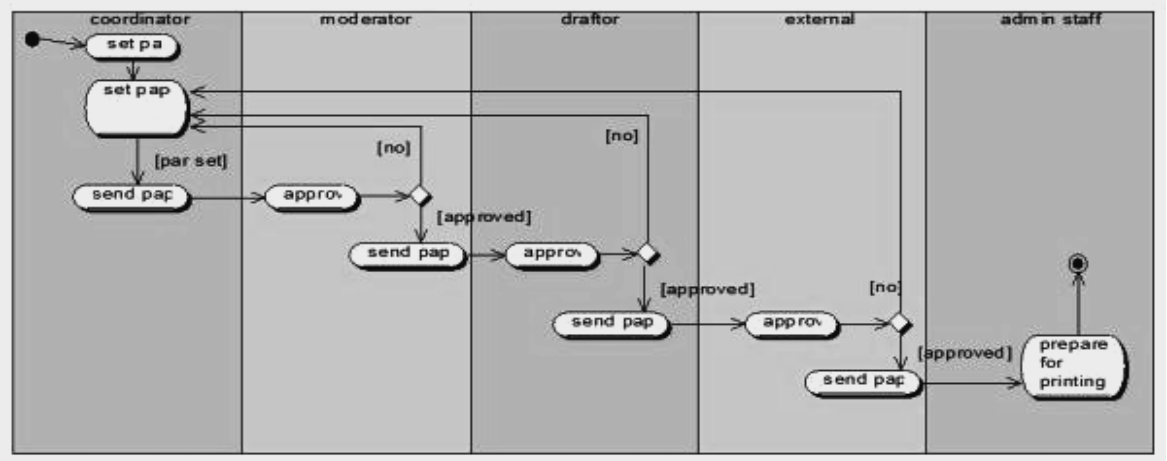
4. Artifact
   Artifacts are physical file that execute or are used by software of the system.
   Artifacts includes:
   (i) Executable files such as .exe or .jar files
   (ii) Library files such as .dll files
   (iii) Source files such as .java or .cpp files
   (iv) Configuration files that are used by software at runtime in specific format such as .xml or .txt

5. **Node instance**
   Instance of a node means two or more nodes of similar node type. In diagram there can be more than one nodes with same properties and structure each node with similar structure is referred as instance of a node. Each instance has its unique identity.



Q.3(e) Explain Swimlane in activity diagram with example. **[4]**

Ans.: [Diagram - 4 marks]



Q.3(f) Explain different types of relationships in UML **[4]**

Ans.: Relationships in the UML: [Any Two - 2 marks each]

1. **Dependency:**
   A dependency relationship specifies that a change in the specification of one thing may affect another thing that uses it. Graphically, a dependency is rendered as a dashed line, directed to the thing that is dependent on.
   Notation: ----------->

2. **Association:**
   An association is a structural relationship, specifying that objects of one thing are connected to objects of another. When an association connects two classes you can navigate from one object of one class to an object of another class and vice versa. Graphically an association is represented as a solid line connecting more than one class.
   Notation: ———————

3. **Generalization:**
   It is a relationship between a general thing (called the super class or parent) and a more specific kind of that thing (called the subclass or child). With generalization relationship a child can inherit all the structure and behavior of parent. A child may add new structure or behavior or it may modify the behavior of the parent. Graphically it is represented as a solid line with hollow triangle placed near to the super
   Notation: ——————▷

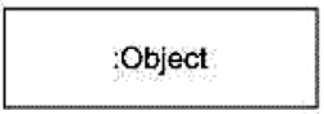
Q.4(a) Attempt any TWO of the following : **[16]**

Q.4(a) State and explain notations used to draw sequence diagram. Draw sequence diagram for library management system: Book issue **[8]**

Ans.: Notations [1 mark each]

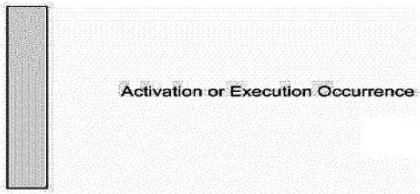
(1) Object

Class roles describe the way an object will behave in context. Use the UML object symbol to illustrate class roles, but don't list object attributes.



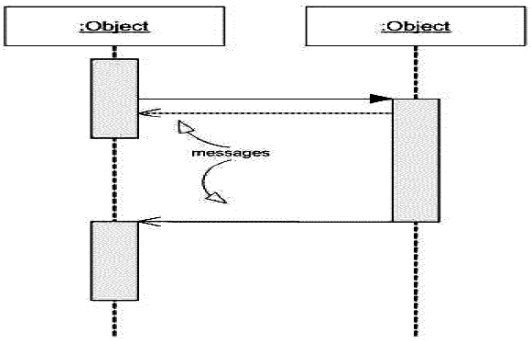
(2) Activation or Execution Occurrence

Activation boxes represent the time an object needs to complete a task. When an object is busy executing a process or waiting for a reply message, use a thin gray rectangle placed vertically on its lifeline.



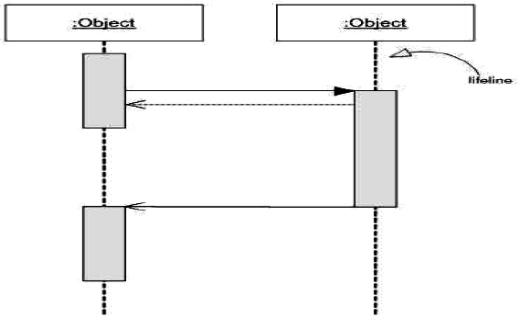
(3) Messages

Messages are arrows that represent communication between objects. Use half-arrowed lines to represent asynchronous messages. Asynchronous messages are sent from an object that will not wait for a response from the receiver before continuing its tasks.



(4) Lifelines

Lifelines are vertical dashed lines that indicate the object's presence over time.



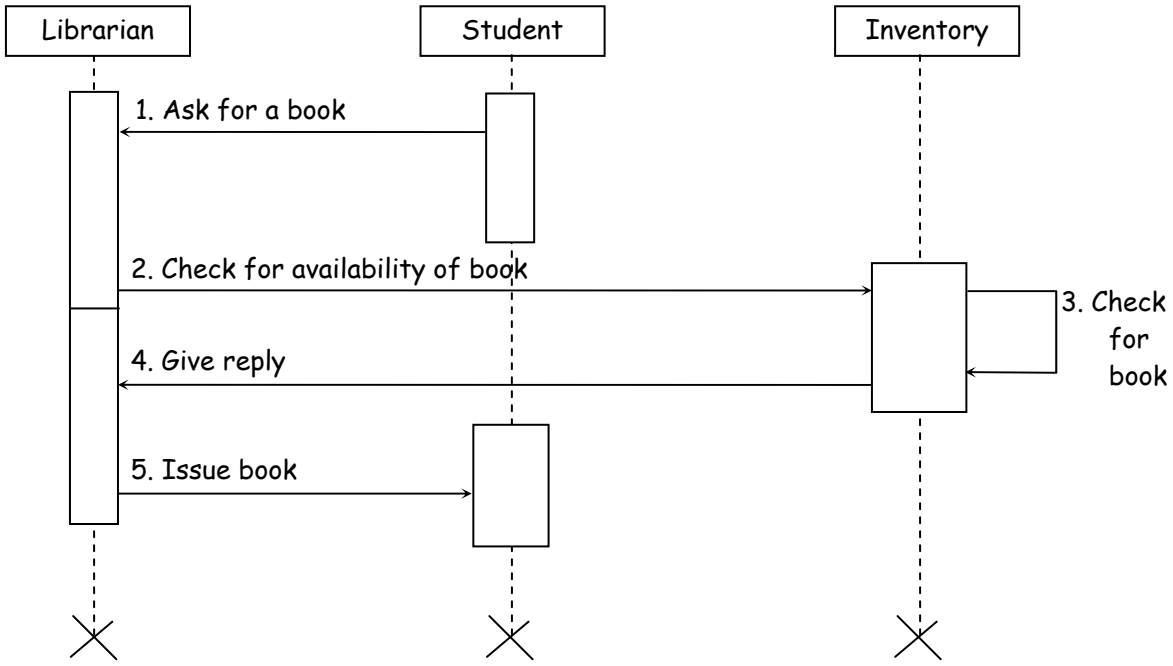
(5) Destroying objects

Objects can be terminated early using an arrow labeled "<<destroy>>" that points to an X. This object is removed from memory. When that object's lifeline ends, you can place an X at the end of its lifeline to denote a destruction occurrence.

(6) Loops

A repetition or loop within a sequence diagram is depicted as a rectangle. Place the condition for exiting the loop at the bottom left corner in square brackets [ ].

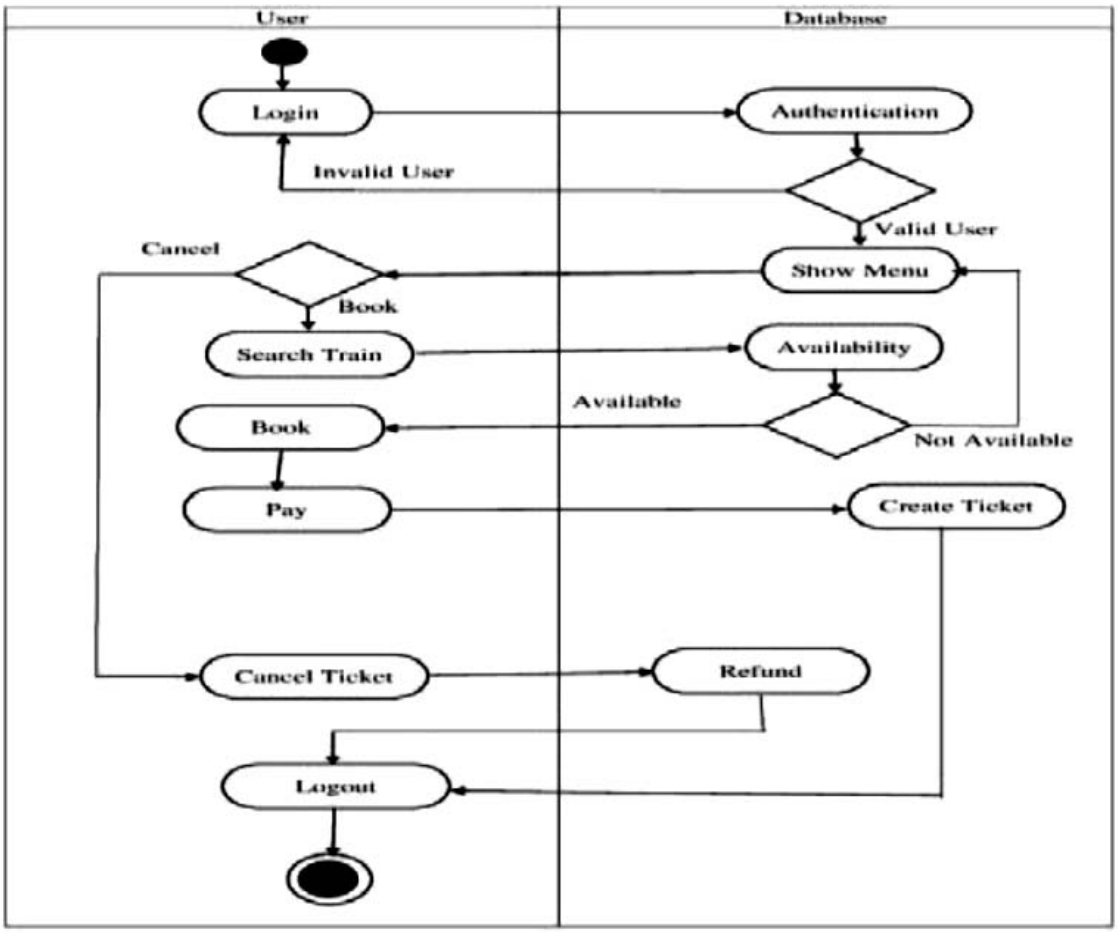
Library management system - Book Issue sequence diagram [2 marks]



Q.4(b) Draw activity and state diagram for online railway reservation system. [8]

Ans.: [Activity Diagram - 4 marks; State Diagram - 4 marks]

Any relevant diagram shall be considered



Q.4(c) State any two types of actors used in use case diagram. Draw use case diagram for online shopping. [8]

Ans.: Types of actors used in use case diagram: [Any Two types of actors - 2 marks each]

(i) Primary/principle Actor

People who use the main system functions are refereed as primary or principle actors.
Example: in ATM system primary actor is customer.

(ii) Secondary Actor

People who perform administrative or maintenance task are referred as secondary actors. It provides a service to the system under design.
Example: in ATM system a person in charge of loading money into the system is a secondary actor.

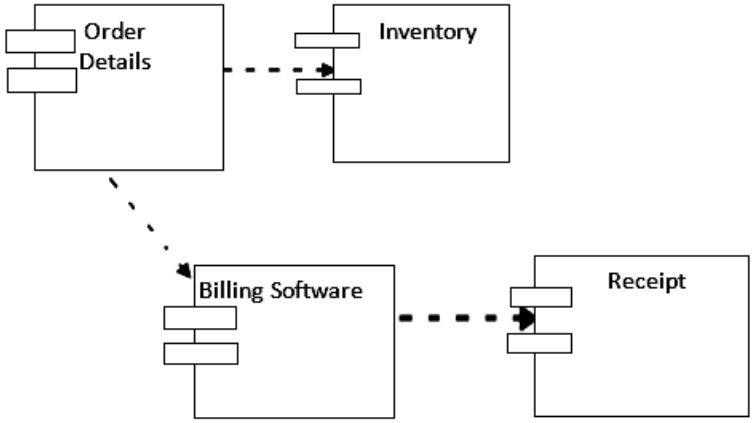
(iii) External actor

The hardware devices which are required as apart of application domain and must be used are referred as external actors.
Example: in ATM system printer is an external actor.

(iv) Other system actor

The other system with which the system must interact referred as other system actors.
Example: in ATM system bank network system is another system actor.

[Case Diagram - 4 marks]



Q.5 Attempt any FOUR of the following : [16]

Q.5(a) List and classify UML diagrams [4]

Ans.: UML Diagrams are classified into two major category as follows :

1. **Structure Diagram** [2 marks]
   a. Class Diagram
   b. Object Diagram
   c. Deployment Diagram
   d. Component Diagram
2. **Behavior Diagram** [2 marks]
   a. Activity Diagram
   b. Use case Diagram
   c. State Machine Diagram
   d. Interaction Diagram

Q.5(b) Explain aggregation with example. [4]

Ans.: Aggregation: [2 marks]

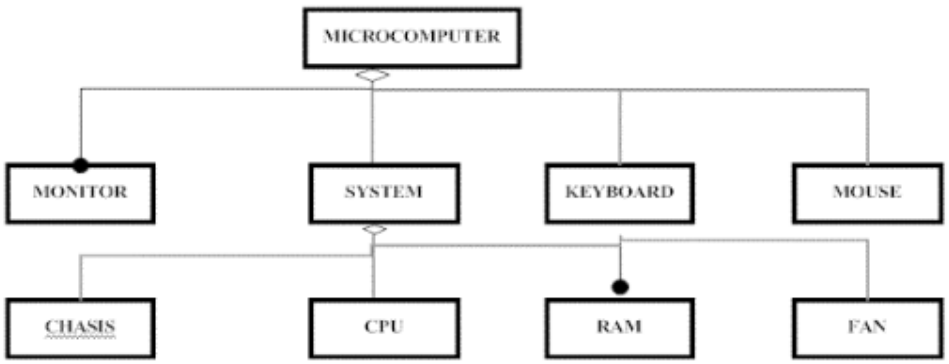
Aggregation is a strong form of association in which an aggregate object is made of components.

Components are part of aggregate. Aggregation is the „part-whole" or a-part-of" relationship in which objects representing the components of something are associated with an object representing the entire assembly.

Properties:

- Aggregation is the "Part-whole" or "a-part-of" relationship in which objects representing the components.
- An Aggregation is a specialized association.
- Aggregation is tightly looped from of association with same extra semantics.
- Example shows that a document consists of many sentences.
- Aggregation is drawn like association, except a small diamond indicates the assembly end of the relationship.

Example [2 marks]

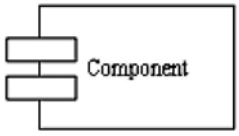


Q.5(c) Explain notations used to draw component diagram. **[4]**

Ans.: Notations used to draw component diagram [Any four - 4 marks]

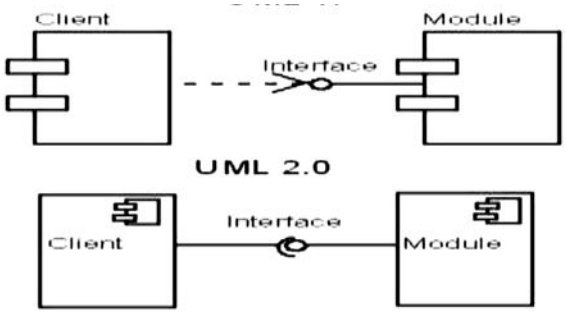
1. **Component:**

   A component is a physical and replaceable part of the system that provides or uses set of interfaces. A component is shown as a rectangle with tabs. A component has name that distinguish it from other components. Name of the component is written as a text inside the rectangle.



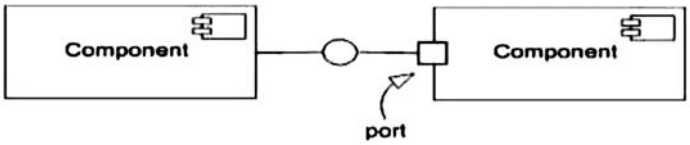
2. **Interfaces:**

   A component can be connected with other components through interfaces. An interface is a collection of operations that are used to specify services of components. A component can provide an interface or can use services of a component. A full circle represents an interface created or provided by the component. A semi-circle represents a required interface.



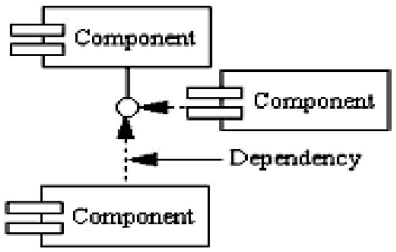
3. **Port:**

   A port specifies an interaction point through which a component can communicate with its environment, other components or with its internal parts. Ports are represented using a square along the edge of a component. A port is often used to help expose required and provided interfaces of a component.



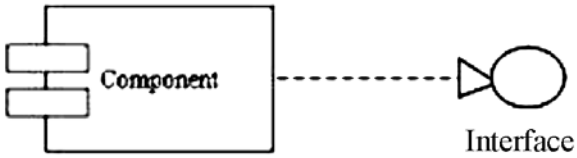
4. **Dependencies**

   A dependency exists between two elements. Changes to the definition of one element may cause changes to the other. It is represented as dashed line with an arrow.



5. **Realization**

   A component realizes an interface by providing service through interface. It is indicated with a dashed line and a hollow arrow head.

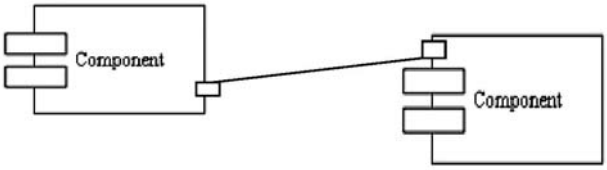


6. **Connector**

   It is a link that specifies communication between two or more classifiers.

Connectors are of two types :

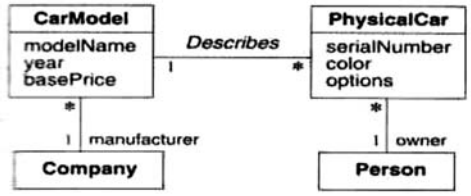
1. Delegation connector: a component realizes or uses an interface. A component can have internal parts. A part of a component can realize or use an interface. To show a connection among internal parts of a component and interface delegation connector is used.
2. Assembly connector: it is a connector between two or more parts or ports on parts that defines services provided by parts for other parts.



Q.5(d) Explain metadata with example. **[4]**

Ans.: Metadata [Explanation with example - 4 marks]

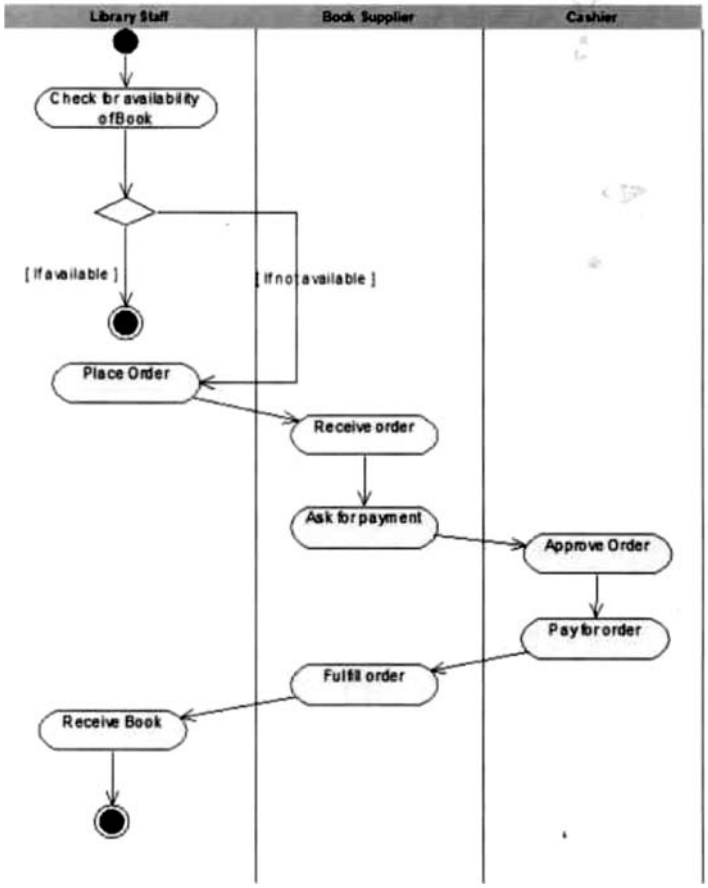
It is a data that describes other data for example a class definition is a metadata. UML models are also referred as metadata as they describe the things required for the application many real world applications have metadata such s parts, catalogues, blue-prints and dictionaries.



In above example car model has a model name, year, base price and manufacturer a physical car has a serial no,, color, options and owner. A car model describes many physical car and stores common data about them. A car model is referred as metadata which relates to the data of physical care. A class descriptor object contains feature and they can have their own classes which are known as meta classes.

Q.5(e) Draw activity diagram for product purchase order. **[4]**

Ans.: [Relevant diagram - 4 marks]

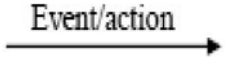


Q.5(f) Draw and explain notations used for state diagram. **[4]**

Ans.: State chart diagram: [Explanation with diagram (any four) - 4 marks]

A state chart diagram shows flow of control from one state to another state.

Notations:

	Name	Symbol	Description
1.	State		A state is a condition or a situation in the life of an object during which it satisfies some conditions, performs some activity or waits for some events. It is represented with a rounded rectangle. Name of the state is written inside the rectangle.
2.	Initial State		It indicates the default starting place of the state diagram. An initial state is represented as filled circle.
3.	Final State		Final state indicates end of the execution of the system. It is represented as a filled black circle surrounded by an unfilled circle.
4.	Transition		A transition is a relationship between two states. It indicates that an object in the first state performs some action and enters in the second state when a specific event occurs. Transition is represented with a directed line.
5.	Event		An event is the specification of a significant occurrence that has location in time and space. An event can be a signal or a call to a function. An event is indicated with text written above or below transition line.
6.	Action		An action is an executable computation. Action may include operation calls, the creation and destruction of another object or sending of a signal to an object. It is indicated with text written below or above the transition line associated with an event separated by slash.

Q.6 Attempt any FOUR of the following : **[16]**

Q.6(a) Explain multiplicity with example. **[4]**

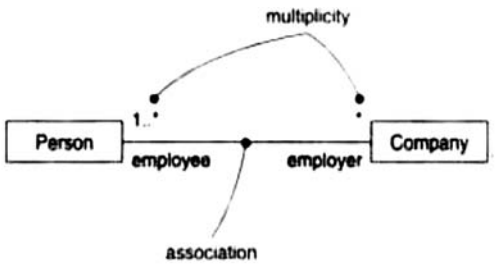
Ans.: Multiplicity [Explanation with example - 4 marks]

Multiplicity specifies the number of instances of one class that may relate to a single instance of an associated class. The UML specifies multiplicity as follows:

Notations:

1. ""1" exactly one
2. "1…*" One or more
3. "3-5" three to five
4. 0..1 zero to one
5. "2,4,18" two, four or eighteen
6. Symbol * denotes "many".

Example

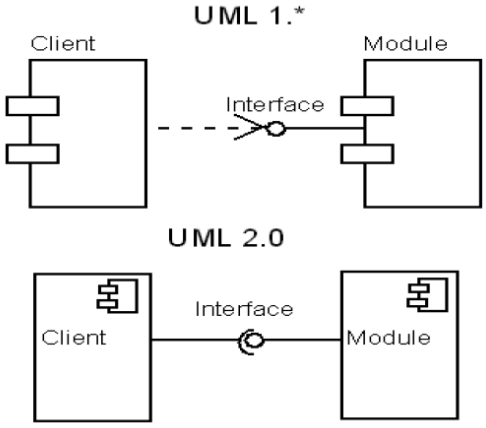


In the above example one or more persons may work in many companies.

Q.6(b) Explain concepts of interface and ports. **[4]**

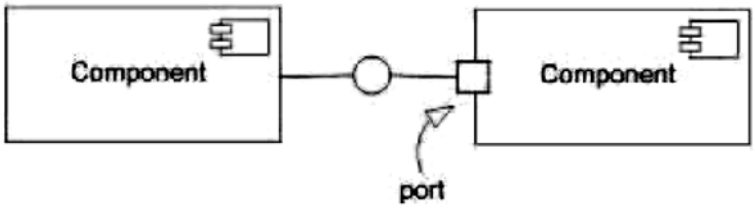
Ans.: Interface [Relevant explanation of interfaces - 2 marks, ports - 2 marks]

A component can be connected with other components through interfaces. An interface i.e. small circle or semi-circle on a stick describes a group of operations used (required) or created (provided) by components. A full circle represents an interface created or provided by the component. A semi-circle represents a required interface, like a person's input.



Ports:

Ports are represented using a square along the edge of the system or a component. A port is often used to help expose required and provided interfaces of a component.



Q.6(c) Explain propagation of operations with example. **[4]**

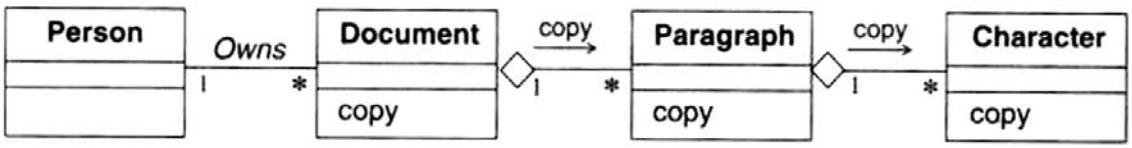
Ans.: Propagation of Operation: [Explanation with example - 4 marks]

Propagation is also called as triggering. It is the automatic application of an operation to a network of objects when the operation is applied to some starting object.

Example

Moving an aggregate object moves its parts along with it. So the operation move applied to aggregate object propagates move operation to the parts.

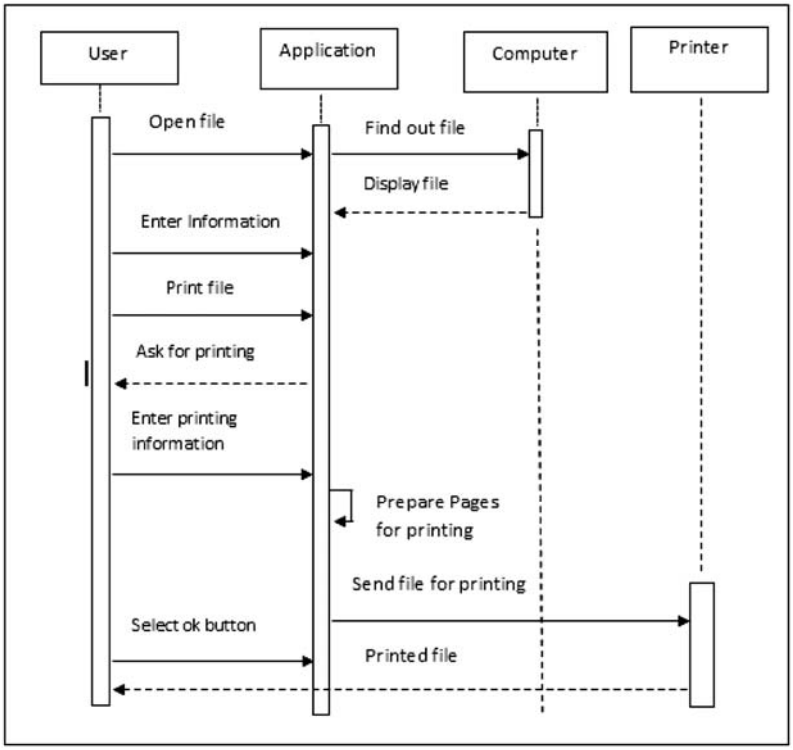
Propagation of operation is shown with a small arrow indicating direction of propagation above association line.



Above diagram shows an example of propagation. A person owns multiple documents. Each document consists of paragraph that in turn consists of characters. The copy operation propagates from documents to paragraphs to characters. Copying a paragraph copies all the characters in it. The operation does not propagate in the reverse direction. A paragraph can be copied without copying the whole document.

Q.6(d) Draw a sequence diagram for printing a file. **[4]**

Ans.: [Any relevant sequence diagram with proper notation – 8 marks]



Q.6(e) Explain structural controls in sequence diagram with example. **[4]**

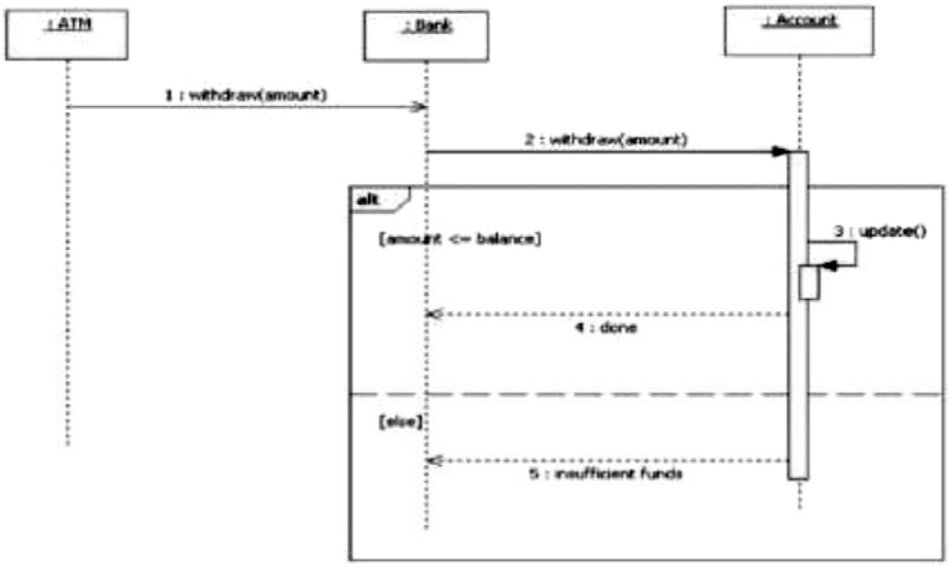
Ans.: **Structure control in sequence diagram:** [Explanation with example - 4 marks]

It defines statements or group of statements in a diagram which determines the sequence of execution of other instructions or statements.

Conditional execution:

1. In sequence diagram conditional statements are used to check the condition.
2. Alt operator is used to indicate the condition in sequence diagram. The body of the control operator is divided into multiple sub regions by horizontal dashed lines.
3. Each sub region represents one branch of a condition. Each sub region has a guard condition. If the guard condition for a sub region is true, then that sub region is executed.
4. At most one sub region may be executed.
5. One sub region may have the special guard condition [else]. This sub region is executed if none of the other guard conditions are true.

Example:

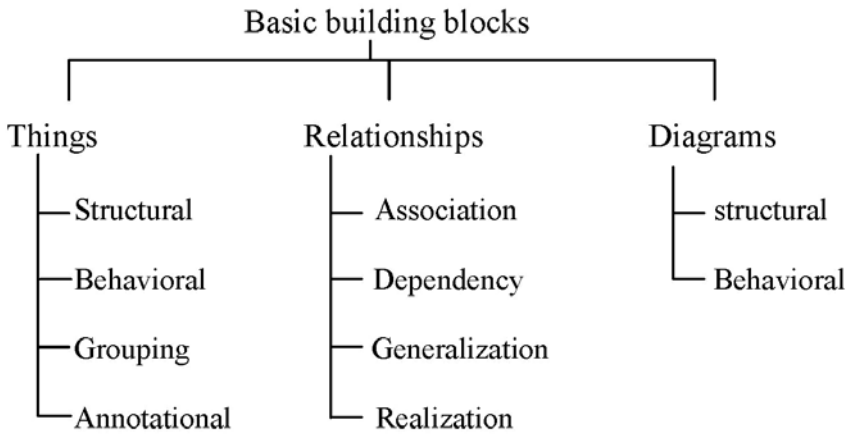


In the above example alt operator is used to show condition as 'amount >=balance'. If the condition is true then, update operation will be perform on account object. If the condition is false then, else region executes if a message as insufficient funds from account object to bank object.

Q.6(f) Explain conceptual model of UML with neat diagram. [4]

Ans.: Conceptual model of UML with neat diagram [Explanation with diagram - 4 marks]

Conceptual model of UML consist of basic building blocks, the rules that dictate how those building blocks may put together and some common mechanisms that apply throughout the UML.



Things: things are the abstractions that are first citizens in a model.

1. **Structural things:** They are nouns of UML model. These are the static parts that represent elements that are either conceptual or physical. It includes class, interface, collaboration, use case, active class, component, and node.
2. **Behavioral things:** They are dynamic parts of UML model. It includes interaction, state machine.
3. **Grouping things:** They are the organizing parts of UML model. It includes package.
4. **Annotational things:** They are the explanatory parts of UML model. It includes notes.

Relationships: The relationships are basic building blocks of the UML.

1. **Association:** It is structural relationship that describes a set of links among objects.
2. **Dependency:** It is a semantic relationship between two things in which change to one thing may affect the semantics of the other thing.
3. **Generalization:** It is a relationship in which objects of the specialized element are substitutable for objects of the generalized element.
4. **Realization:** It is a semantic relationship between classifiers, wherein one classifier specifies a contract that another classifier guarantees to carry out.

Diagrams:

1. Structural diagrams: It is used to visualize, specify, construct and document the static aspects of a system. It includes class, object, and component and deployment diagram.
2. Behavioral diagrams: It is used to visualize, specify, construct and document the static aspects of a system. It includes use case, sequence, collaboration, state chart and activity diagram.

❑ ❑ ❑ ❑ ❑