SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Modeling and Design **Subject Code: 17630**

Important Instructions to examiners:

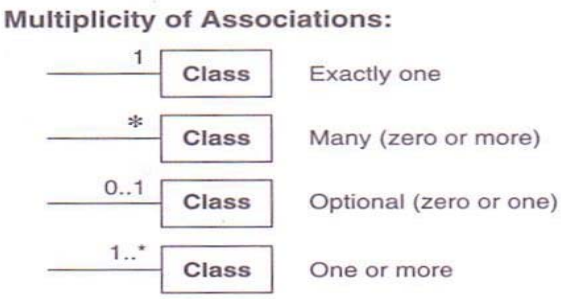
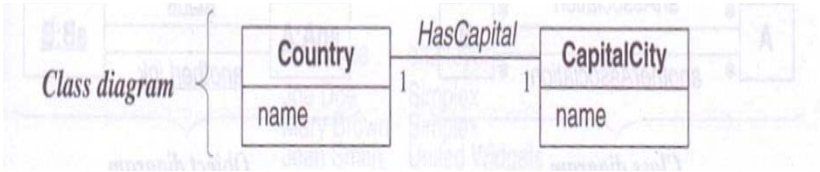
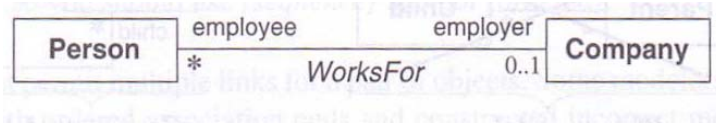
1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills).
4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
7) For programming language papers, credit may be given to any other program based on equivalent concept.

Q. No.	Sub Q.N.	Answer	Marking Scheme
1.	**a) Ans.**	**Attempt any FIVE of the following:** **State and explain benefits of OOA.** **Benefits of Object Oriented Analysis are:** **1. Object Oriented Approach** It focuses on data rather than the procedures as in structured analysis. **2. Data encapsulation and hiding** The principles of encapsulation and data hiding help the developer to develop systems that cannot be tampered by other parts of the system. **3. Modularity** It allows effective management of software complexity by the virtue of modularity. **4. Real-World Modeling:** Objects are organized into classes of objects, and objects are associated with behaviors. The model is based on objects, rather than on data and processing.	**20** **4M** ***Any four relevant benefits 1M each***

		5. High Code Reusability: When a new object is created, it will automatically inherit the data attributes and characteristics of the parent class. The new object will also inherit the data and behaviors from all super classes in which it participates. **6. Structured Analysis :** It can be upgraded from small to large systems at a greater ease than in systems following structured analysis. **7. Improved Reliability and Flexibility:** Objects are dynamically called and accessed, new objects may be created at any time. The new objects may inherit data attributes from one, or many other objects. Behaviors may be inherited from super-classes and added without effecting existing systems functions.	
	b) Ans.	**Explain dynamic modeling.** **Dynamic modeling:** 1. The dynamic model describes those aspects of a system concerned with time and the sequencing of operations, events that make changes, sequences of events, states that define the context for events and the organization of events and states. 2. The dynamic model captures control, that aspect of a system that describes the sequences of operations that occur without regard of what the operations do, what they operate on, or how they are implemented. 3. It is important for interactive systems, but insignificant for purely static data repository, such as database. Following steps are performed in constructing a dynamic model: 1. Prepare scenarios of typical interaction sequences. 2. Identify events between objects. 3. Prepare an event trace for each scenario. 4. Build a state diagram. 5. Match events between objects to verify consistency. Diagrams used for dynamic modeling: 1. Sequence diagram 2. Collaboration diagram 3. State chart diagram 4. Activity diagram	**4M** ***Correct explanation 4M***

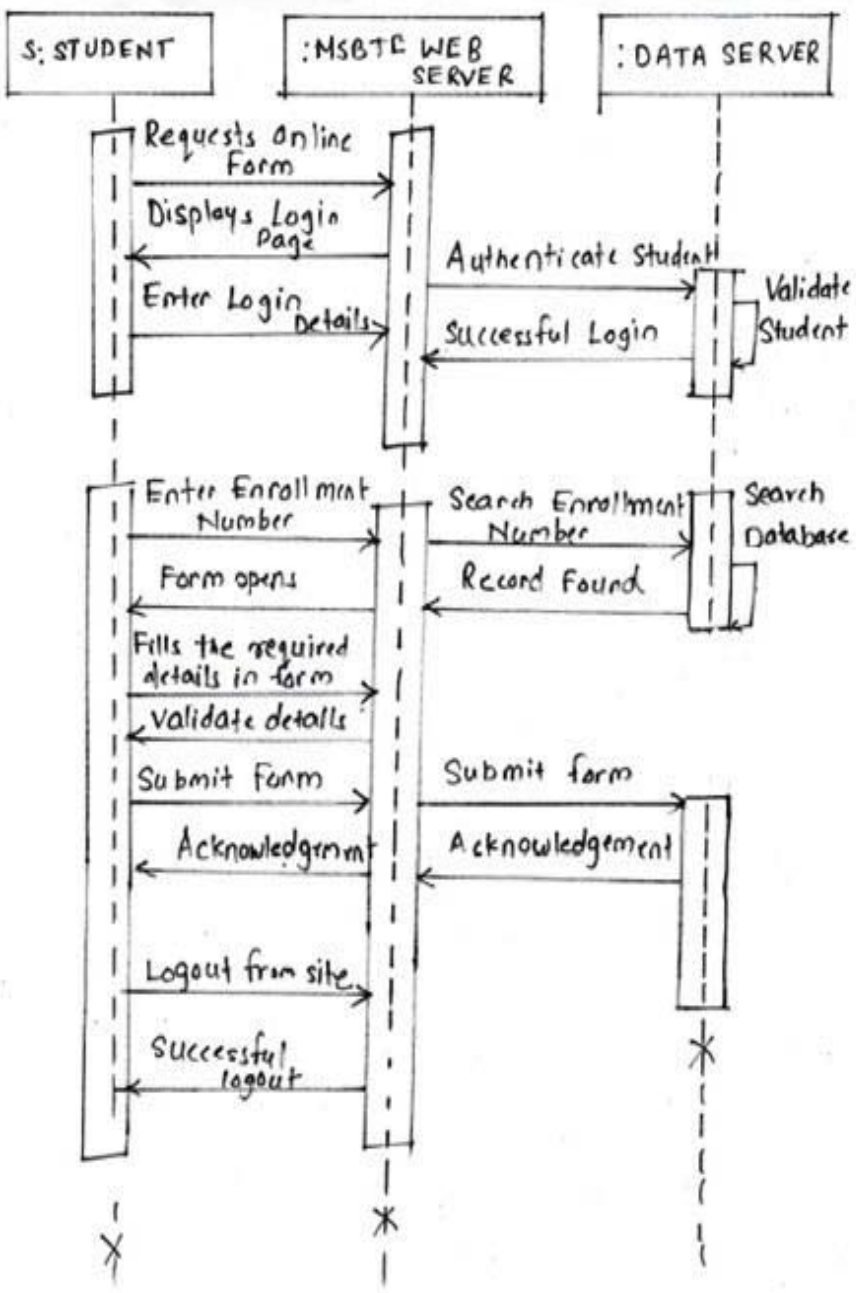
SUMMER – 2019 EXAMINATION
MODEL ANSWER

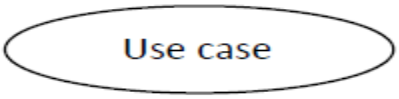
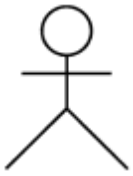
Subject: Object Oriented Modeling and Design **Subject Code: 17630**

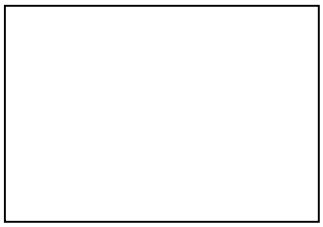
	c) Ans.	**Describe multiplicity with suitable example.** **Multiplicity:** Multiplicity specifies the number of instances of one class that may relate to a single instance of an associated class. Multiplicity decides the number of related objects. Multiplicity is generally explained as "one" or "many," but in general it is a subset of the non-negative integers. 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 :**  **Fig: one-to-one association**  **Fig: one to many association**	**4M** ***Correct description of Multiplicity 2M*** ***Suitable example 2M***

SUMMER – 2019 EXAMINATION
MODEL ANSWER

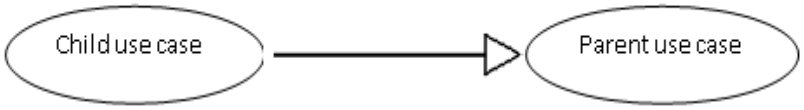
Subject: Object Oriented Modeling and Design **Subject Code: 17630**

	d) Ans.	**Draw the sequence diagram for msbte online exam form filling.** 	**4M** ***Correct sequence diagram with proper notations 4M***

	e) Ans.	**Give notations used for use case diagram along with its pictorial representations.** **Notations used in use case diagram are:** **1. Use case:** Use case is the description of set of sequences of actions. It is graphically represented as an ellipse and labeled with the name of the use case. Use case represents an action performed by a system. **Notation:**  **2. Actor:** An actor represents a coherent set of roles that users of use case can play while interacting with use cases. An actor represents a role that a human, hardware device or another system plays when it communicates with the system. It is represented with the stickman notation. **Notation:**  **3. Communication Line:** A Communication line is a connection between an actor and use case. It indicates that both are communicating with each other. Communication line is represented with a solid line. **Notation:** **4. System Boundary:** System boundary specifies the scope of an application in order to specify functionality. It indicates what the system includes and what it omits. System boundary groups together logically related things. It separates use cases and actors involved in the system. System boundary is shown with a box in a use case diagram. **Notation:**	**4M** ***Any four correct notations 1M each***

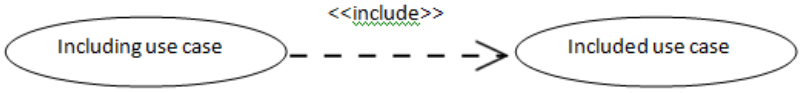


5. Generalization: Generalization is used to show the relationship between two use cases. In this relationship the child use case inherits the behavior and meaning of parent use case. It is represented with the solid line with a large hollow triangle as an arrowhead. Arrow head indicates direction of generalization.



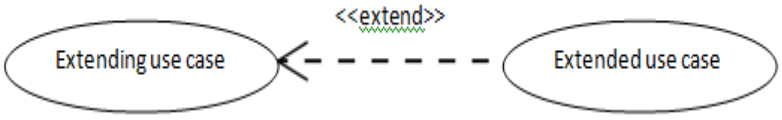
6. Include Relationship: An include relationship is the directed relationship between two use cases. Including a use case requires forceful execution of included use case.

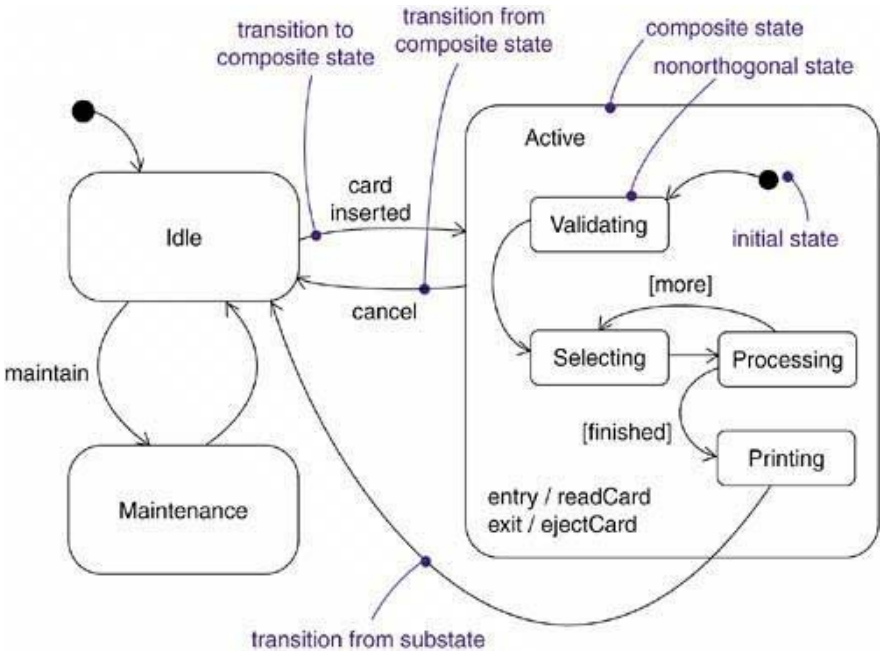
Notation:



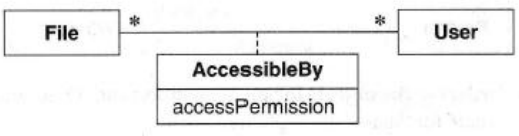
7. Extend Relationship: An extend relationship is a directed relationship between two use cases that specifies extra actions in a system. Extend relationship specifies optional behavior for extending use case.

Notation:

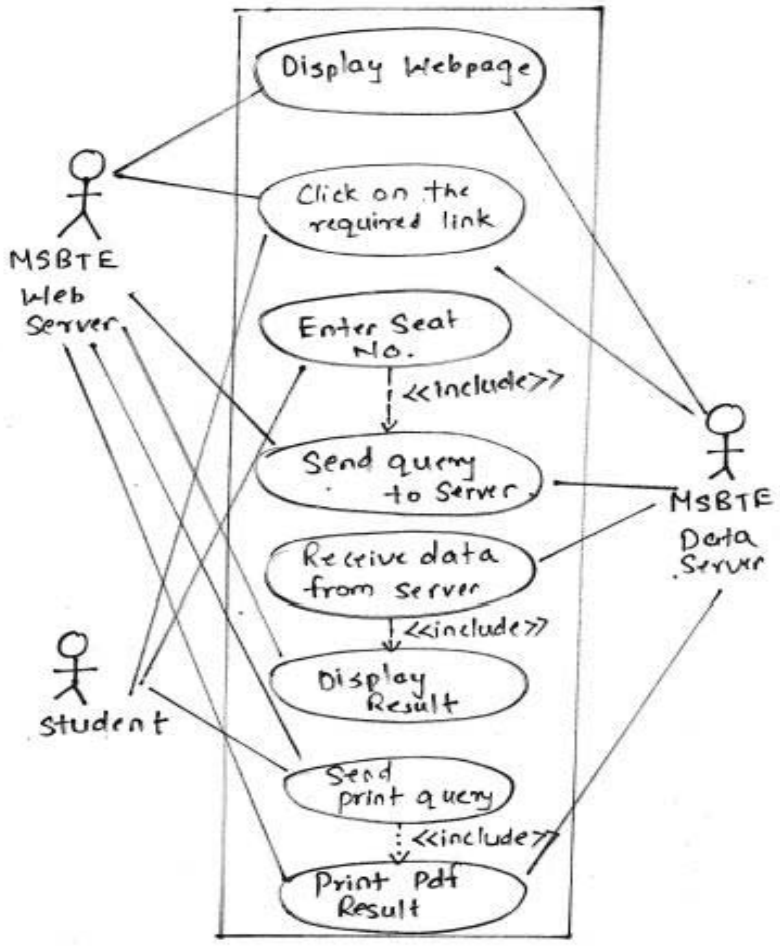


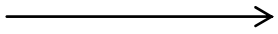
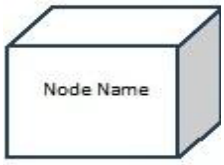
	f) Ans.	**Explain sequential substrate with example.** **Sequential Substates:** 1. Sequential sub states are those sub states in which an event common to the composite states can easily be exercised by each states inside it at any time. 2. Sequential sub states partition the state space of the composite state into disjoint states. 3. A nested sequential state machine may have at most one initial state and one final state. **Example:**  Consider the problem of modeling the behavior of an ATM. This system might be in one of three basic states: Idle (waiting for customer interaction), Active (handling a customer's transaction), and Maintenance (perhaps having its cash store replenished). While Active, the behavior of the ATM follows a simple path: Validate the customer, select a transaction, process the transaction, and then print a receipt. After printing, the ATM returns to the idle state. It might represent these stages of behavior as the states Validating, Selecting, Processing, and Printing. It would even be desirable to let the	**4M** ***Correct explanation 2M*** ***Relevant example 2M***

		customer select and process multiple transactions after validating the account and before printing a final receipt.	
	g) Ans.	**List and explain types of transitions.** **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. **Notation:** ⟶ **Types of transition:** 1. Automatic/Lambda/Trigger less transition (without condition) 2. Guarded / triggered transition (with condition) 1. Automatic/Lambda/Trigger less transition (without condition) Transition without an event name indicates an automatic transition. It fires when the activity associated with the source state is completed. If there is no activity, the unlabeled transition fires as soon as the state is entered. 2. Guarded / triggered transition (with condition) Guarded transition fires when its event occurs, but only if the guard condition is true.	**4M** ***List of transitions 2M*** ***Explanation 2M***
2.	**a) Ans.**	**Attempt any <u>FOUR</u> of the following:** **State the characteristics of UML.** **Unified Modeling language** is a standard language for writing software blueprints. It is very expressive language addressing all the views needed to develop &deploy. **Characteristics of UML:** 1) Visualization Thoughts are brought into code by a programmer. Text is a minimal and direct way to write expressions and algorithms. 2) Specification Specification means building models that are precise, unambiguous and complete. UML addresses the specification of all the important analysis, design and implementation decisions that must be made in developing and deploying a software	**16** **4M** ***Any four characteristics 1M each***

		intensive system. 3) Construction In UML, models can be directly connected to a variety of programming languages. It is possible to map from a model in the UML to a programming language such as JAVA, C++. Visual Basic or object-oriented databases. 4) Documentation A healthy software organization produces all sorts of artifacts in addition to raw executable code. These artifacts include • Requirements • Architectures • Design • Source code • Project plans • Tests • Prototypes • Releases	
	b) Ans.	**How can you use associations as a class.** Use of association as a class: 1. Association Class is an association that is also a class. 2. Like the links of an association, the instances of an association class derive identity from instances of the constituent classes. 3. Like a class, an association class can have attributes and operations and participate in associations. **Notation:** The UML notation for an association class is a box attached to the association to the association by a dashed line. **Example:**  In above example, 'access Permission' is an attribute of class 'Accessible By'. The sample data at the bottom of the figure shows the value for each link.	**4M** ***Association class with any relevant example 4M***

	c)	**Give two advantages and disadvantages of** **(i) Class** **(ii) Advanced class diagram** ***(Note: Advantages and Disadvantages stated below are for Class Diagram and Advanced Class diagram. Description of class with example shall be given marks.)***	**4M**
	Ans.	**(i) Class** A class describes a group of objects with the same properties (attributes), behavior (operations), kinds of relationships and semantics. **Class Example:** **STUDENT** Name Rollno Grade getName() printGrade() **OR** **(i) Class Diagram:** **Advantages:** 1. It is used to model vocabulary of a system. 2. It is used to model simple collaborations. 3. It is used to model a logical database schema. **Disadvantages/limitations:** **1.** Class diagram represents only structure of object , it doesn't explain the flow of object 2. Class diagram doesn't specify conditions applied to object. **(ii) Advanced class diagram:** **Advantages:** 1. Analysis and design of the static view of an application. 2. It provides a blueprint for maintenance programmers to get an overview of how the application is structured before examining the actual code which may reduce maintenance time. **Disadvantages:** 1. The programmer may need to learn UML to build the class	***Any two advantages and disadvantages 1M each***

		diagram in the first place. 2. The time spent building the class diagram may add to overall development time.	
	d) **Ans.**	**Draw the use case diagram for printing result from MSBTE website.** ***(Note: Any other relevant diagram with correct notation shall be considered).***  	**4M** ***Use case diagram with correct notations 4M***
	e) **Ans.**	**Explain transitions and their types.** **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	**4M**

		state when a specific event occurs. Transition is represented with a directed line. Notation:  **Types of transition:** 1. Automatic transition Transition without an event name indicates an automatic transition that fires when the activity associated with the source state is completed. If there is no activity, the unlabeled transition fires as soon as the state is entered. Such unlabeled transitions are sometimes called as lambda transitions. 2. Triggerless transition Trigger less transition will fire only if that condition is met. It may have guard condition.	***Explanation of transition 2M*** ***Types of transition 2M***
	f) Ans.	**List and draw the notations used in deployment diagram.** Notations used in deployment diagram are: 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 anode or to the component. It is shown with a solid line.	**4M** ***Any four correct notations 1M each***

3. Communication line-dependency

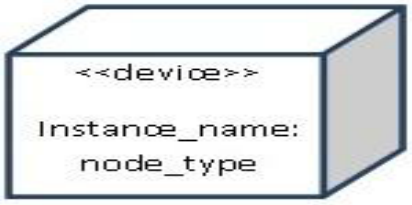
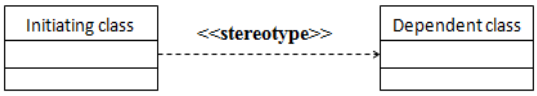
It is used to show relationship between node and a component. Dependency is shown with dashed line and arrow head. It connects node with the component arrow head points towards component.

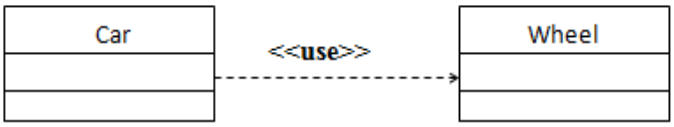
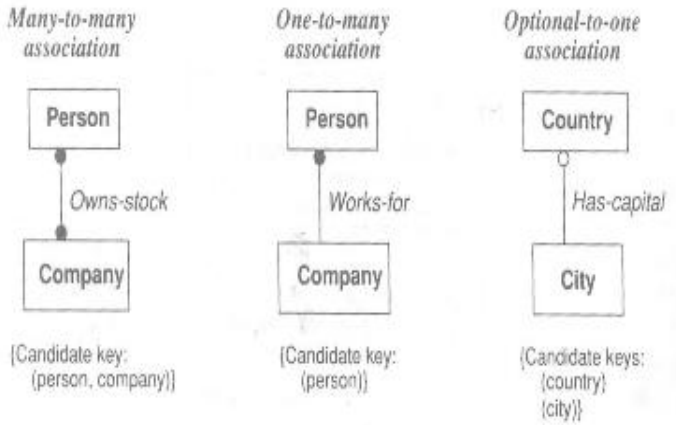
4. Artifact

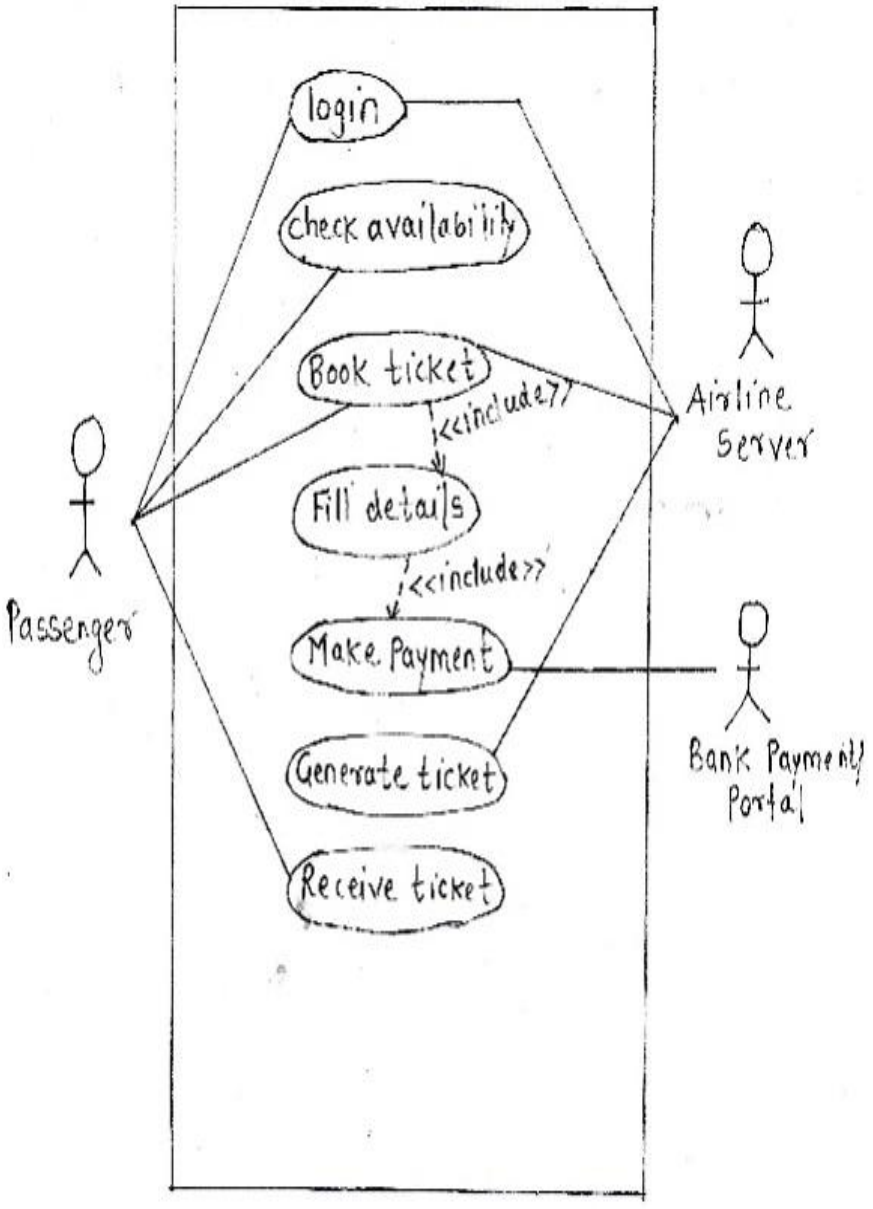
Artifacts are physical file that execute or are used by software of the system.

Artifacts includes:

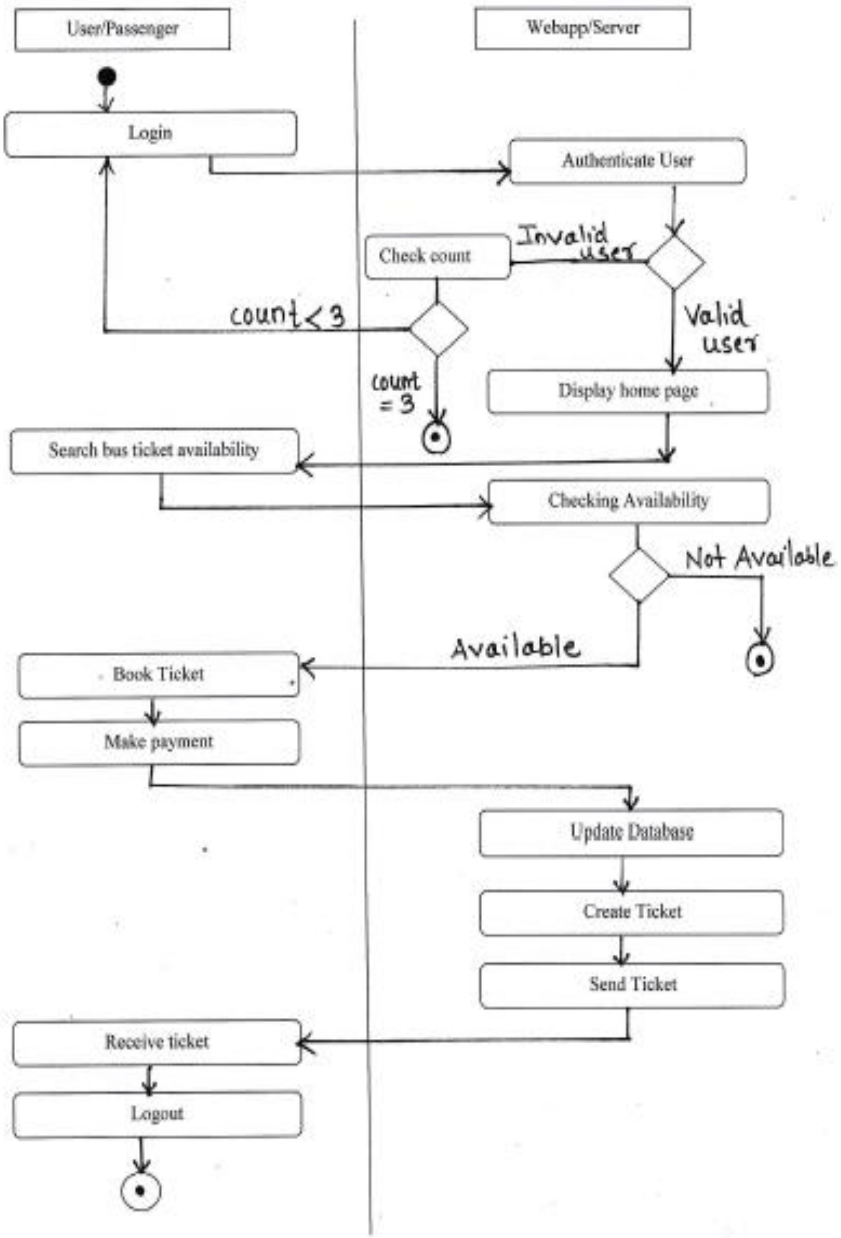
1. Executable files such as .exe or .jar files
2. Library files such as .dll files
3. Source files such as .java or .cpp files
4. Configuration files that are used by software at runtime in specific format such as .xml or .txt

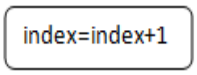
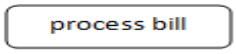
		5. Node instance Instance of a node means two or more nodes of similar node type. In diagram there can be more than one nodes with same properties and structure each node with similar structure is referred as instance of a node. Each instance has its unique identity  	
3.	**a)** **Ans.**	**Attempt any <u>FOUR</u> of the following:** **Describe the term dependencies w. r. t to class diagram.** Dependency relationship specify that a change in the specification of one thing (class) may affect another thing(class) that uses it. **Notation:** Dependency is shown as a dashed line directed towards the class that depends on the initiating class.  Stereotypes applicable to dependency relationship among classes: • **bind**: Specifies that the source instantiates the target template using given parameters. • **derive**: Specifies that the source may be computed from the target. • **friend**: Specifies that the source is given special visibility into the target. • **instanceOf**: Specifies that the source object is an instance of the target classifier. • **instantiate**: specifies that the source creates instances of the target. • **use:** Specifies that the semantics of the source element depends on the semantics of the public part of the target.	**16** **4M** ***Correct descripti on 4M***

		Example:  	
	b) Ans.	**Explain candidate key with example and diagram.** It is a minimal set of attributes that uniquely identifies an object or link. It means you cannot discard an attribute from the candidate key and still distinguish all objects and links. A class or association may have one or more candidate keys, each of which may have different combinations and numbers of attributes. The object id is always a candidate key for a class. One or more combinations of related objects are candidate keys for associations. Notation: -A candidate key is delimited with braces in an object model. Example:  A many-to-many association requires both related objects to uniquely identify each link. A one-to-many association has a single candidate key: the object on the many side. A one-to-one association has two candidate keys: either of the objects.	**4M** ***Explanation 2M*** ***Example with diagram 2M***

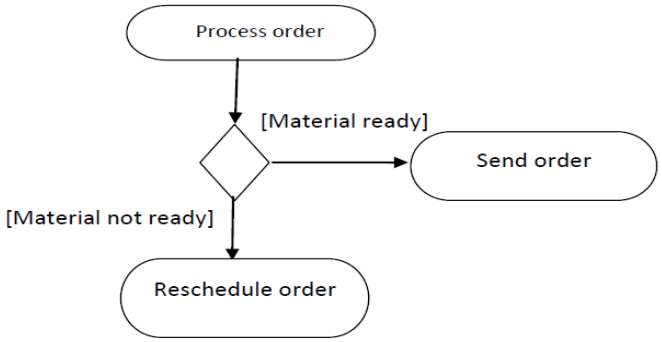
	c) **Ans.**	**Draw the use case diagram for online Airline reservation system.** ***(Note: Any other relevant diagram with correct notation shall be considered).***  	**4M** ***Correct use case diagram for online Airline reservation 4M***

	d) **Ans.**	**Draw the activity diagram for e-ticket booking for bus.** ***(Note: Any other relevant diagram with correct notation shall be considered).***	**4M** ***Correct activity diagram with proper notation 4M***



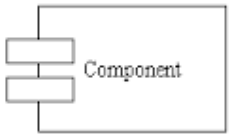
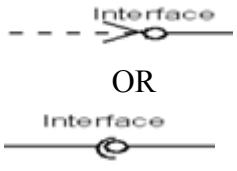
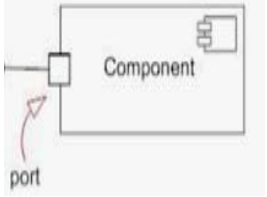
	e) **Ans.**	**Explain the following term under activity diagram.** **(i) States** **(ii) Branching** **(i) States:** Action state: The executable, atomic computations such as sending a signal to an object, creating or destroying object are called as action state. Action state cannot be decomposed. For example: an expression for calculating gross salary, entering amount for withdrawal cannot be decomposed. Example:  **OR** Activity state: an activity is an ongoing non-atomic execution within an activity diagram. Activity results in action. Activity state can be further decomposed in multiple activities. Activity states are not atomic that means they may be interrupted and they may take some time duration to complete. Activity state is a composite of flow control made up of other activity states and action states. Example:  **(ii) Branching:** In an activity diagram, branching is used to show alternate path depending on the result of Boolean expression. In a system, some application processing may require flow of control based on Boolean expression. A branch may have one incoming transition and two or more outgoing transitions. The outgoing transitions are evaluated only when a branch is executed. Branching contains a decision box that holds Boolean expression. Depending on result of expression one of the branches is executed. Notation:-	**4M** ***Explanation of each term 2M***

Diamond Shape is used for Decision and branches are represented by lines. The condition written in diamond is the decision criteria. Lines representing branches has guard condition with it.

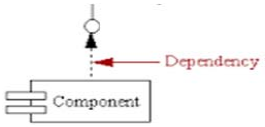
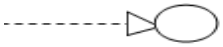
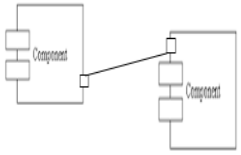


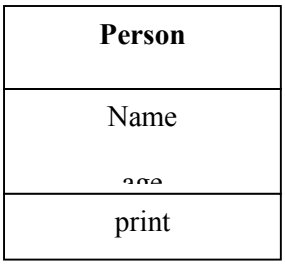
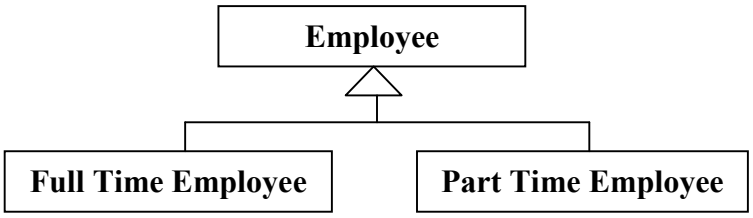
f) Ans. **Draw and list the notations used for component diagram.** **4M**

Following are the notations for component diagram :

Sr. No.	Notation	Use
1		**Component-** A component is a physical and replaceable part of the system that provides or uses set of interfaces.
2		**Interface-** An interface is a collection of operations that are used to specify services of components.
3		**Port:** A port specifies an interaction point through which a component can communicate with its environment, other components or with its internal parts.

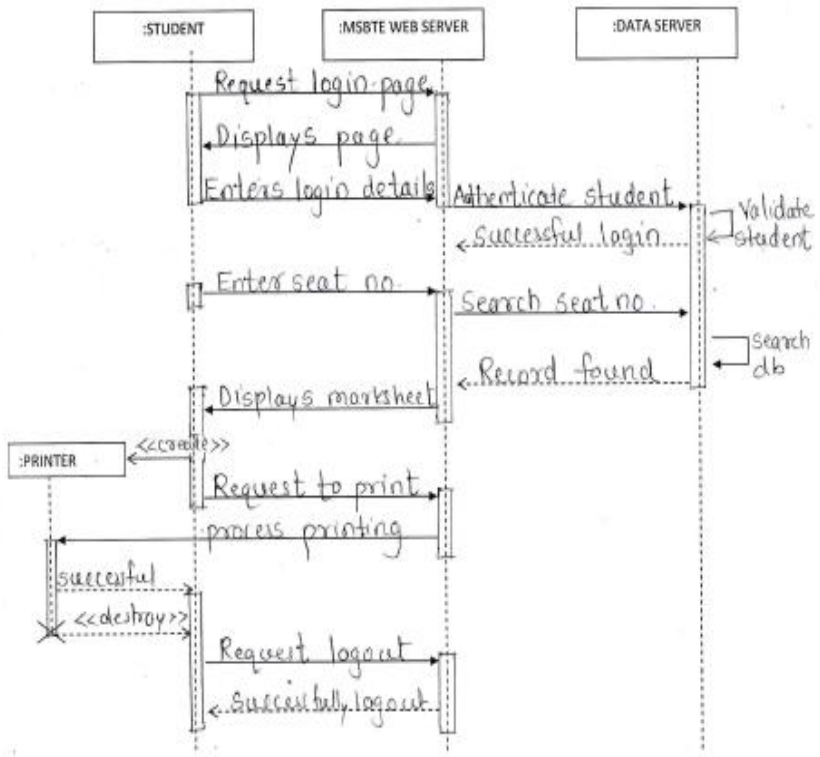
Any four notation 1M each

		4		**Dependency:** A dependency exists between two elements. Changes to the definition of one element may cause changes to the other.
		5		**Realization:** A component realizes an interface by providing service through interface.
		6		**Connector:** It is a link that specifies communication between two or more classifiers. • Delegation connector • Assembly connector
4.	**a)**	**Attempt any TWO of the following:** **Explain characteristic of object with respect to** **(i) Identity.** **(ii) Classification.** **(iii) Polymorphism.** **(iv) Inheritance.**		**16** **8M**
	Ans.	**(i) Identity:** It means that data is quantized into discrete, distinguishable entities called object. Objects can be concrete, such as a file system, or conceptual such as scheduling policy in multiprocessing operating system. Each object has its inherent identity. Two objects are distinct even if all their attribute values are identical. Example: (Person) ABC **(ii) Classification:** It means that objects with the same data structure (attributes) and behavior (operations) are grouped into a class. A class is an abstraction that describes properties important to an application and ignores the rest. Each class describes a possible infinite set of individual objects. Each object is said to be an instance of a its class.		***Explanation of each term 2M***

		Example:  **(iii) Polymorphism:** It means that the same operation may behave differently for different classes. The move operation behaves differently for the pawn than for the queen in a chess game. An operation is an action or transformation that an object performs or is subject to. A specific implementation of an operation by a certain class is called a method. Example: Draw operation will draw a circle if radius and x,y coordinates are given and will draw rectangle if x1,y1,x2,y2 values are given. **(iv) Inheritance:** It is sharing of attributes and operations (features) among classes based on hierarchical relationships. a class can be defined broadly and then refined into successively finer sub classes. Each sub class inherits all of the properties of its super class and adds its own unique properties.  	
	b) **Ans.**	**Draw sequence diagram for printing semester end exam marksheet by student from MSBTE website.** ***(Note: Any other relevant diagram with correct notation shall be considered).***	**8M**

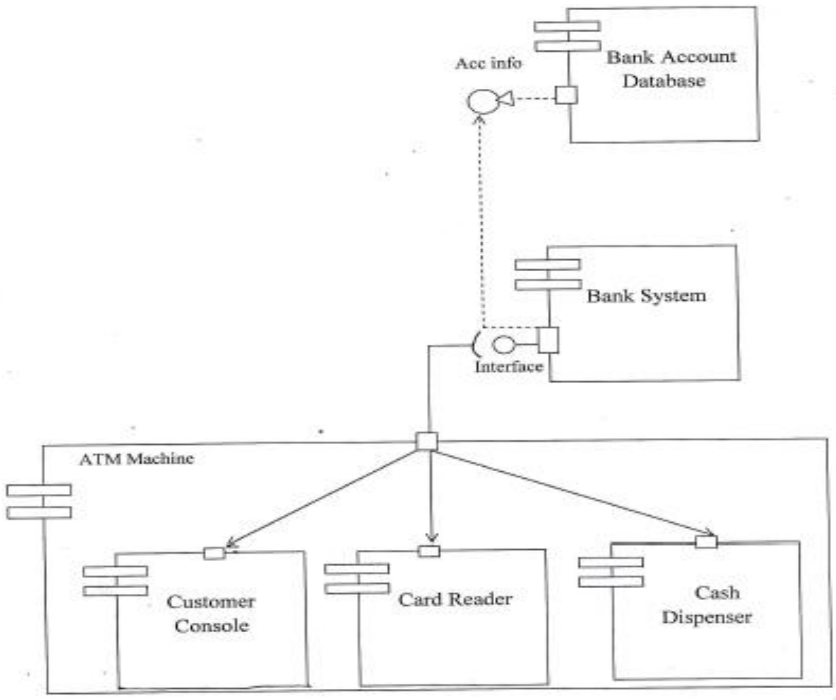
SUMMER – 2019 EXAMINATION
MODEL ANSWER

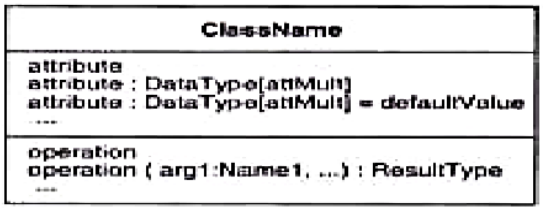
Subject: Object Oriented Modeling and Design **Subject Code: 17630**

			***Correct sequence diagram with proper notations 8M***
	c) Ans.	**Draw and explain the component diagram for ATM system.** A component diagram shows the organization and dependencies among a set of components. Graphically, a component diagram is a collection of vertices and arcs. It is used to model the static implementation view of a system. This involves modeling the physical things that reside on a node, such as executables, libraries, tables, files, and documents. Component diagrams are essentially class diagrams that focus on a system's components.	**8M**

SUMMER – 2019 EXAMINATION
MODEL ANSWER

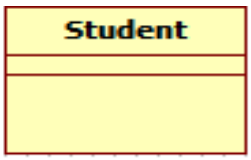
Subject: Object Oriented Modeling and Design **Subject Code: 17630**

		 The above component diagram includes three components for ATM system that includes ATM machine, Bank system and Bank account database. ATM machine contains sub components such as customer Console, Card Reader and Cash Dispenser. Bank account database provide account info to bank system. Bank account database component realizes interface ACC INFO and is used by bank system. Bank system provides interface that is used by ATM machine. Relationship between Bank system and ATM machine components is shown with ball and socket notation.	***Correct component diagram with proper notations 6M*** ***Explanation 2M***
5.	**a) Ans.**	**Attempt any <u>FOUR</u> of the following:** **List and explain four principles of modeling.** **Modelling principles are as follows:** 1. **The choice of what models to create has a profound influence on how a problem is attacked and how a solution is shaped**: This means choose your model well. The right models will brilliantly illuminate the most wicked development problems. The wrong models will mislead you, causing you to focus on irrelevant issues.	**16** **4M**

		2. **Every model may be expressed at different levels of precision**: This means all the users and developers both may visualize a system at different levels of details at different times. 3. **The best models are connected to reality**: in object oriented systems, it is possible to connect all the nearly independent views of a system into one semantic whole. 4. **No single model is sufficient. Every nontrivial system is best approached through a small set of nearly independent models**: to understand the architecture of a system, you need multiple interlocking views such as use case view, design view, process view, implementation view and deployment view. Each of these views may have structural as well as behavioral aspects. Together these views represent a system.	***Each principle 1M***
	b) **Ans.**	**Define class w. r. t to OO and which symbols are used in class diagrams.** **Class:**A class is a group of objects with similar properties(attributes), common behaviour(operation), common relationship to other objects and common semantics. A class is a collection of objects of similar types. **e.g Student,Fruit,Person etc.**  **Example:** **STUDENT** Name Rollno Grade getName() printGrade() Here student class having Name,Rollno,Grade are attributes of class and getName( ) and printGrade( ) are operation on student class.	**4M** ***Definition of class 1M***

Symbols used in class diagram

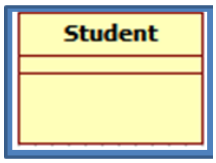
1) Class



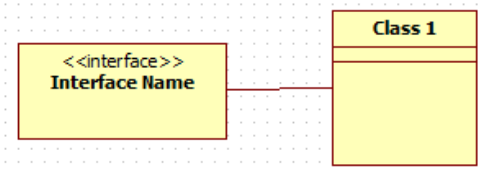
Describe set of objects that share same specification, feature, constraints and semantics.

2) Active Class

- Initiate and control the flow of activity , while passive classes store data and serve other classes,
- Illustrate classes with thicker border.

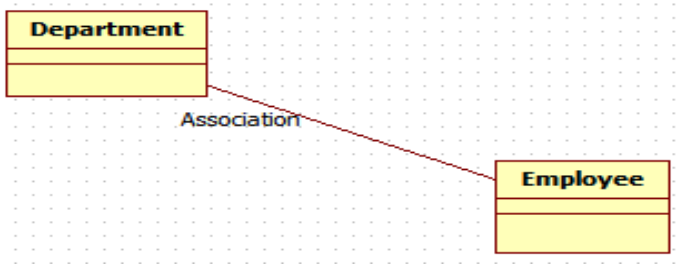


3) Interface: is an abstract class that defines a set of operations that the object of the class associated with this interface provides to other objects.



4) Associations

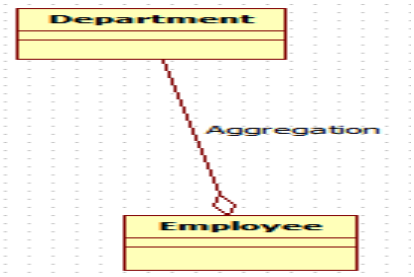
Is a relationship that connects two classes.



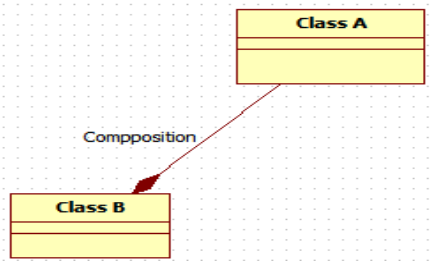
Any 3 symbols 1M each

5) Aggregation

- Is a association with the relation between the whole and its parts
- One class is certain entity that includes other entities
- N-ary Association, Represet two or more aggregations

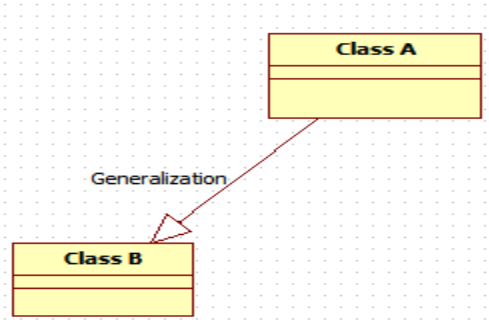


6) Composition: strong variant of aggregation when part cannot be separately of the whole.

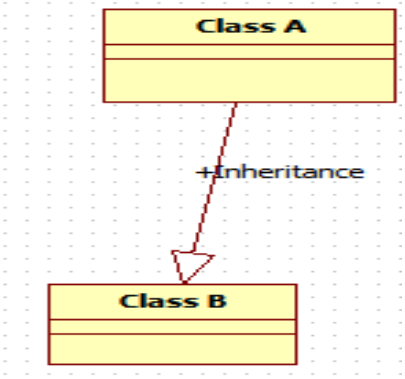


7) Generalization:

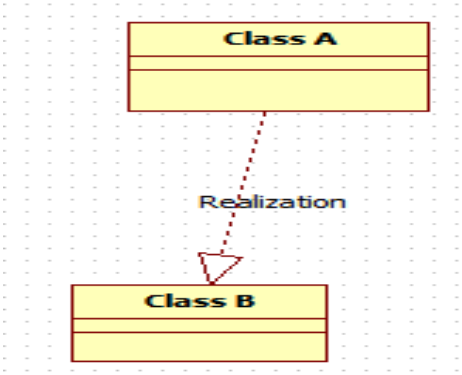
Is a association between the more general classifier and the more special classifier



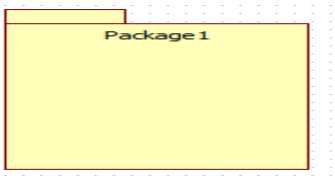
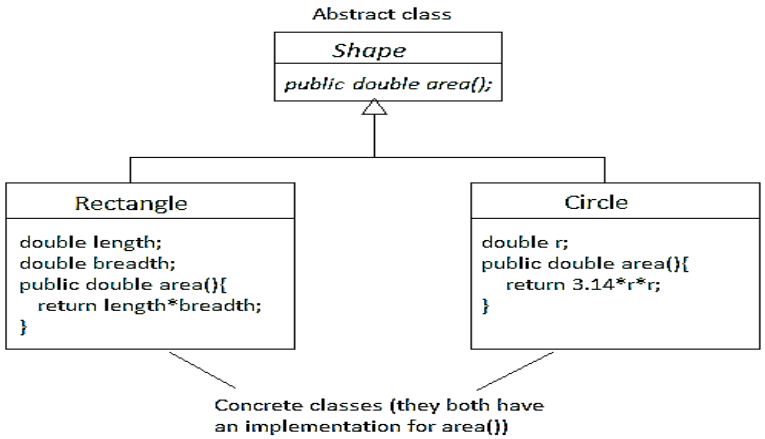
8) Inheritance
Is a relationship when a child object or class assumes all properties of this parent object or class.

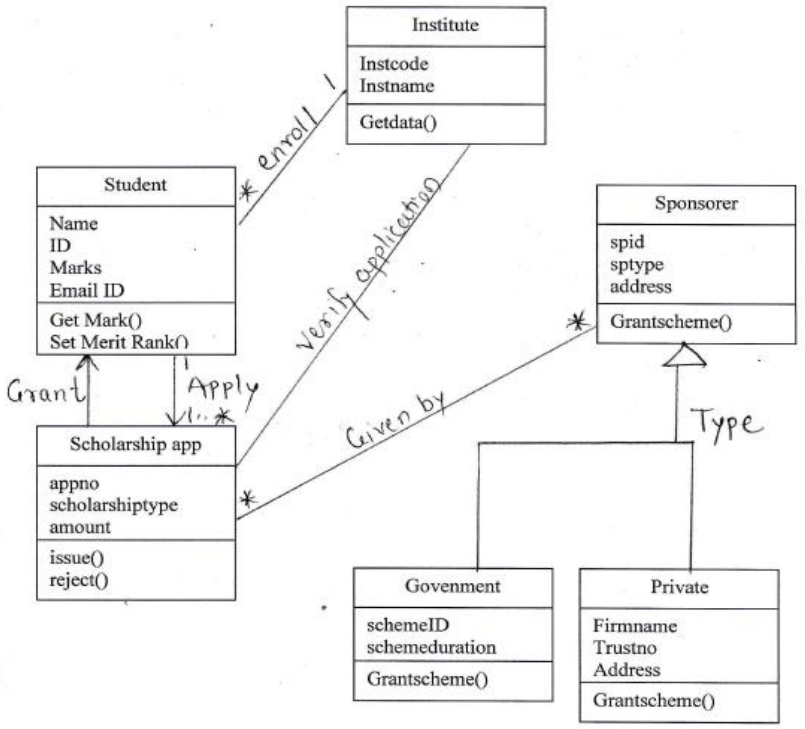


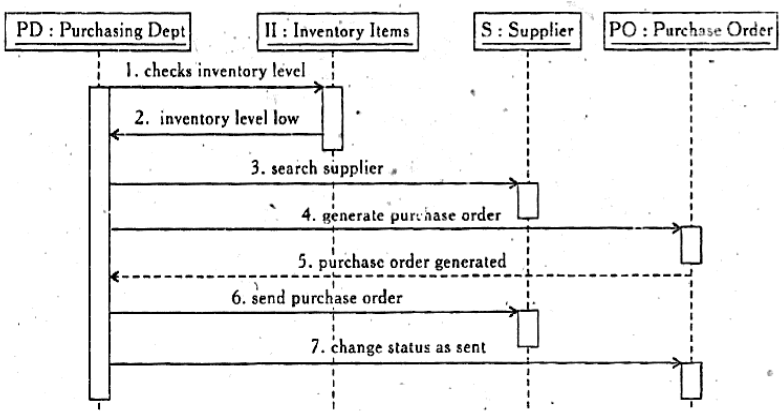
9) Realization
Is a relationship between interfaces and classes or components that realize them.



10) Multiplicity: Represent quantity of instances of one class that are linked to one instance of the other class.

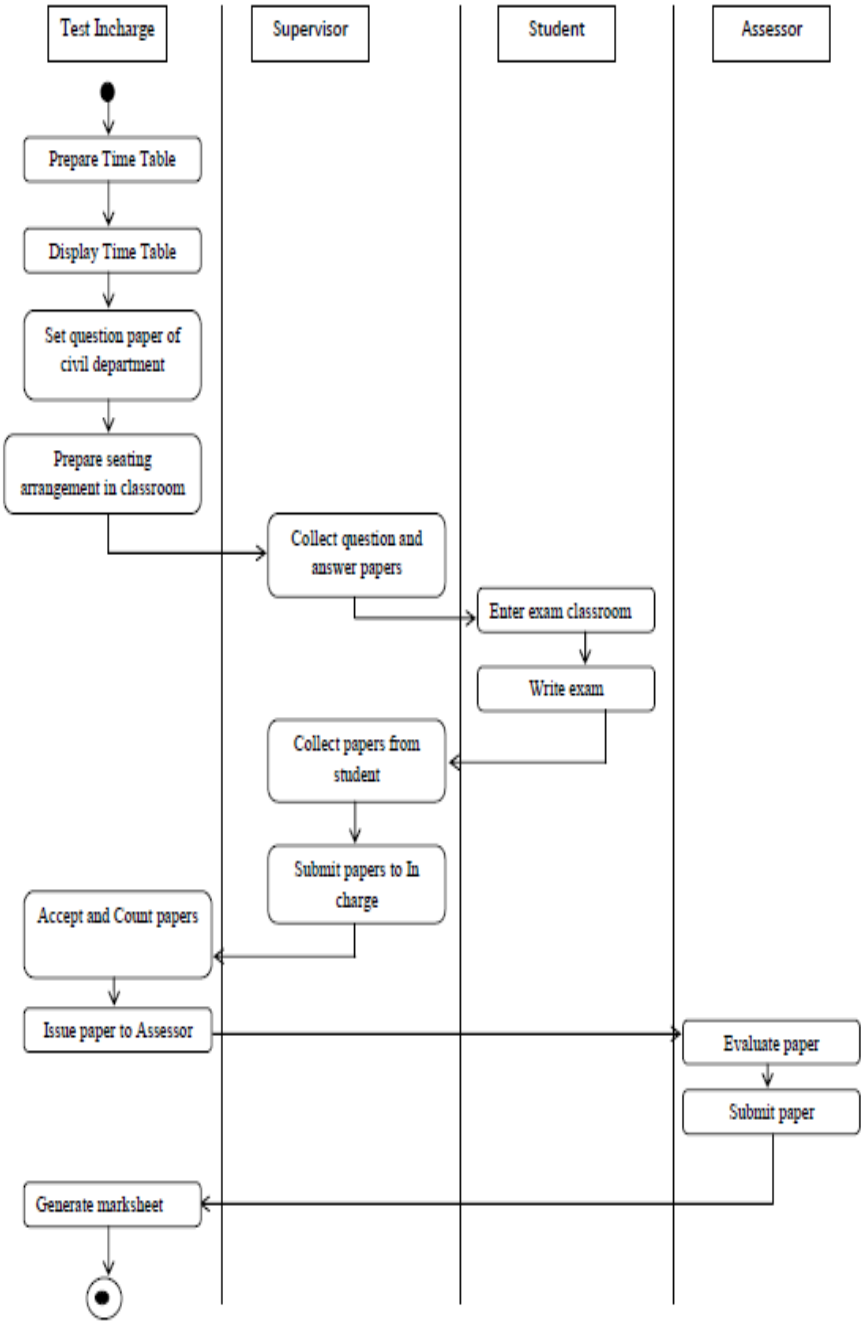
		11) Package: group classes and other packages. 	
	c) Ans.	**Describe Abstract class and Concrete class with diagram.** **Abstract class**: Classes that has no direct instances, but whose descendant classes have direct instances are known as abstract class. Abstract classes are incompletely implemented i.e. the class has methods without implementations. Abstract classes exist purely to generalize common behavior that would otherwise be duplicated across (sub) classes. **Concrete class**: It is a class have direct instances. A concrete class may have abstract subclasses. It can be a leaf class in the inheritance tree. **Example:** The figure shows three classes Shape, Rectangle and Circle. Shape class has one method area which is not implemented. An abstract class contains one or more abstract methods. Therefore, Shape is abstract class where as Rectangle and Circle are concrete classes and inherit Shape. Here concrete classes Rectangle and Circle implement area() method. 	**4M** ***Each description 1M*** ***Example /Diagram 2M***

	d) **Ans.**	**Draw the class diagram for getting scholarship of student.** ***(Note: Any other relevant diagram with correct notation shall be considered).***  	**4M** ***Correct class diagram with proper notations 4M***
	e) **Ans.**	**Explain sequence diagram with suitable example.** ***(Note: Any other example shall be considered)*** 1. A sequence diagram in a Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. 2. It is a construct of a Message Sequence Chart. 3. A sequence diagram shows object interactions arranged in time sequence. 4. A sequence diagram represents the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. 5. A sequence diagram represent the sequence of actions that occurs	**4M** ***Explanation 2M***

		in system. 6. A sequence diagram represents that dynamic behavior of a system. 7. A sequence diagram are read left to right and descending. 8. A sequence diagram is 2-Dimensional in nature. On the horizontal axis, it shows the life of the object that it represents, while on the vertical axis, it shows the sequence of the creation or invocation of these objects. **Example of sequence diagram:** **Placing purchase order: Purchasing** department checks the inventory level of all the items. If stock is less than the department searches the supplier list, generates the purchase order and sends it to the supplier. Purchasing department also maintains the copy of purchase order.  Sequence diagram for placing purchase order	***Example 2M***
	f) **Ans.**	**Draw the activity diagram for conducting unit test of civil department.** ***(Note: Any other relevant diagram with correct notation shall be considered).***	**4M**

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Modeling and Design **Subject Code: 17630**

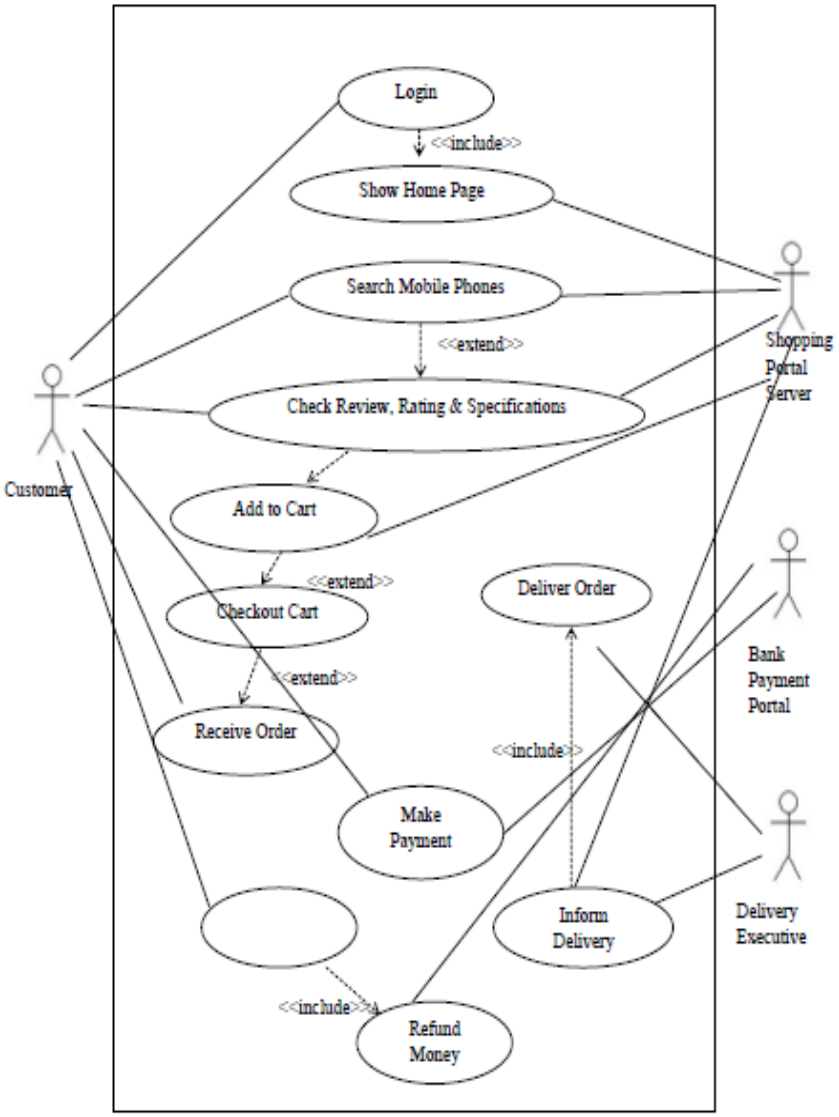


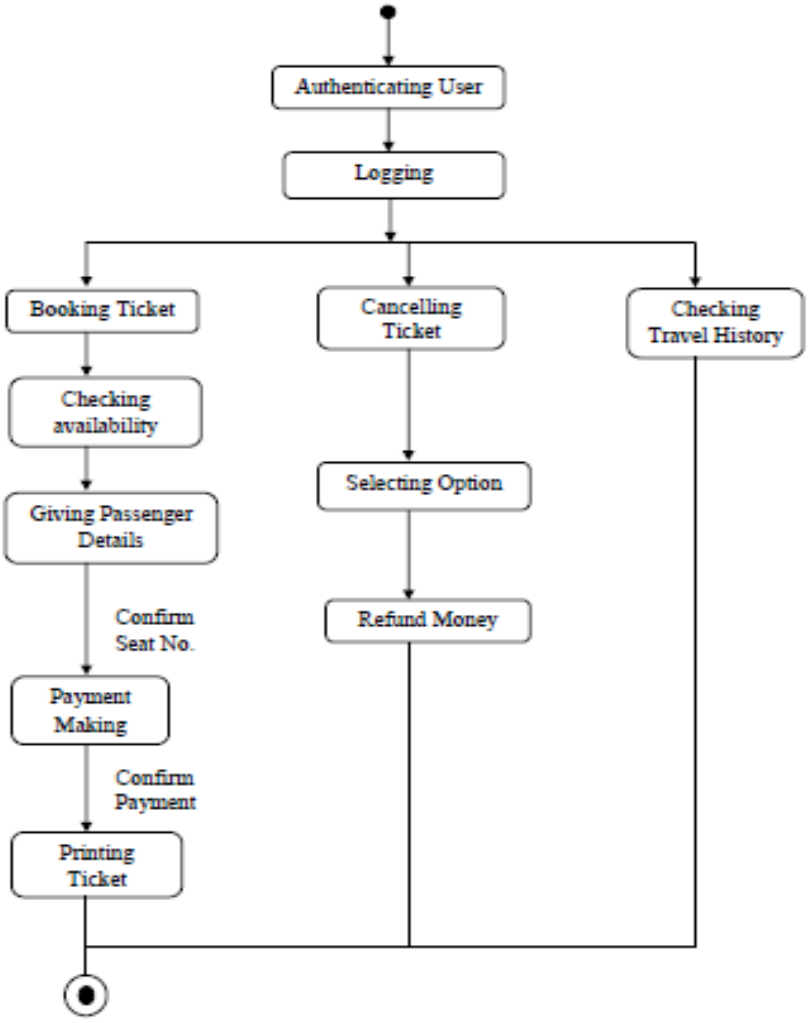
Correct activity diagram with proper notations 4M

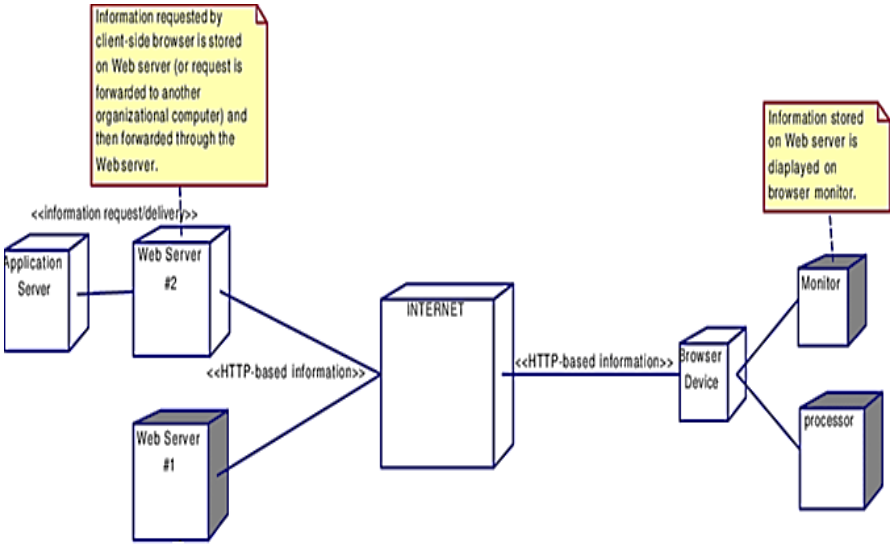
SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Modeling and Design **Subject Code: 17630**

6.	**a)** **Ans.**	**Attempt any TWO of the following:** **Draw the use case diagram for "online mobile purchasing".** ***(Note: Any other relevant diagram with correct notation shall be considered).***	**16** **8M** ***Correct diagram With proper notation s 8M***



	b) **Ans.**	**Draw and explain state chart diagram for online railway reservation system.** ***(Note: Any other relevant diagram with correct notation shall be considered).***  User must register to use online railway reservation facility. Only authenticated user get logging into system. User have 3 possibilities 1) Booking ticket: Check availability of train by providing date, time, train no, name other details. For selected train, passenger details are entered, then user makes required payment and seat get confirmed.	**8M** ***Correct diagram with proper notations 6M*** ***Relevant Explanation of diagram 2M***

		Lastly ticket generated is printed. 2) To cancel ticket select cancelling option and get refund money. 3) Check previous booking from history if any.	
	c) **Ans.**	**Draw and explain deployment diagram for internet and network connection.** ***(Note: Any other relevant diagram with correct notation shall be considered).***  Deployment Diagram provides concise modeling for the physical structure of a Web site. Web server will interact to multiple web servers through HTTP based information. Information stored on web server is displayed on browser monitor Nodes: Application serve Web server Internet	**8M** ***Correct diagram with proper notations 6M*** ***Relevant Explanation of diagram 2M***