**OOAD QUESTIONS**

**Q1. Differentiate an activity diagram from a state diagram.**

Ans. Both activity and state chart diagrams model the dynamic behaviour of the system. Activity diagram is essentially a flowchart showing flow of control from activity to activity. A state chart diagram shows a state machine emphasizing the flow of control from state to state.

• An activity diagram is a special case of a state chart diagram in which all or most of the states are activity states and all or most of the transitions are triggered by completion of activities in the source state (An activity is an ongoing non-atomic execution within a state machine).

• Activity diagrams may stand alone to visualize, specify, and document the dynamics of a society of objects or they may be used to model the flow of control of an operation. State chart diagrams may be attached.

**Q2. What is CRC?**

Ans. A Class Responsibility Collaborator (CRC) model ([Beck & Cunningham 1989](http://c2.com/doc/oopsla89/paper.html); Wilkinson 1995; [Ambler 1995](http://www.ambysoft.com/theObjectPrimer.html)) is a collection of standard index cards that have been divided into three sections, as depicted in [Figure 1](http://agilemodeling.com/artifacts/crcModel.htm#Figure1CRCCardLayout). A class represents a collection of similar objects, a responsibility is something that a class knows or does, and a collaborator is another class that a class interacts with to fulfil its responsibilities.

* An effective way to analyze scenarios
* First was proposed by Beck & Cunningham as a tool for teaching OO programming.
* As a development tool that facilitates brainstorming & enhances communication among developers.
* The cards are arranged to show the flow of messages among instances of each class.

**Q3. Compare and contrast a sequence and a collaboration diagram.**

Ans. (On the next page)

|  |  |
| --- | --- |
| **Sequence vs Collaboration Diagram** | |
| The sequence diagram is a UML representation to visualize the sequence of calls in a system to perform a specific functionality. | The collaboration diagram is a UML representation to visualize the organization of the objects and their interaction. |
| **Representation** | |
| The sequence diagram represents the sequence of messages flowing from one object to another. | The collaboration diagram represents the structural organization of the system and the messages sent and received. |
| **Usage** | |
| If the time sequence is important, the sequence diagram can be used. | If the object organization is important, then the collaboration diagram can be used. |
|  |  |

**Q4. What is the relationship between abstraction, information hiding, and encapsulation?**

Ans.

* An **abstraction** denotes the essential characteristics of an object that distinguish it from all other kinds of objects and thus provide crisply defined conceptual boundaries, relative to the perspective of the viewer.
* Key concepts:
* Concentrating only on essential characteristics
  + Allows complexity to be more easily managed
* Abstraction is relative to the perspective of the viewer
  + Many different views of the same object are possible.

**Information hiding** concept restricts direct exposure of data. Data is accessed indirectly using safe mechanism, methods in case of programming object. Taking bike as an example, we have no access to the piston directly, we can use 'start button' to run the piston.

* ***Encapsulation***is the practice of including in an object everything it needs hidden from other objects. The internal state is usually not accessible by other objects.

Key concepts:

* Packaging structure and behavior together in one unit
  + Makes objects more independent
* Objects exhibit an *interface* through which others can interact with it
* Hides complexity from an object’s clients

**Q5. Differentiate between activity diagrams, flow charts, and state transition diagrams.**

Ans. An **activity diagram** is a UML behaviour diagram. It represents how each activity flows one after the other. An activity is some kind of an operation of the system. Furthermore, activity diagrams help business and development teams of an organization to understand the system processes and behaviours.

A flowchart is a diagrammatic representation of an algorithm. In computer programming, the flowchart diagram helps to write down an algorithm to solve the problem.

The **main difference** **between activity diagram and flowchart** is that an **activity diagram is a UML behaviour diagram that represents the workflow of stepwise activities of the system while a flowchart is a graphical diagram that represents the sequence of steps to solve a problem.**

**Q6. Name the UML diagrams used for the following: (i) modelling requirements (ii) modelling workflows (iii) modelling behaviour of an object (iv) interaction between groups of objects.**

Ans.

(i)

1. **Object Diagram**
2. **Class Diagram**
3. **Sequence Diagram**
4. **Collaboration Diagram**
5. **State chart Diagram**
6. **Activity Diagram**
7. **Usecase Diagram**
8. **Component Diagram**
9. **Deployment Diagram**

**Q7. Why is object orientation needed?**

Ans.

## **1. Modularity for easier troubleshooting**

That’s the beauty of encapsulation. Objects are self-contained, and each bit of functionality does its own thing while leaving the other bits alone. Also, this modality allows an IT team to work on multiple objects simultaneously while minimizing the chance that one person might duplicate someone else’s functionality.

## **2. Reuse of code through inheritance**

Suppose that in addition to your Car object, one colleague needs a RaceCar object, and another needs a Limousine object. Everyone builds their objects separately but discover commonalities between them. In fact, each object is really just a different kind of Car. This is where the inheritance technique saves time: Create one generic class (Car), and then define the subclasses (RaceCar and Limousine) that are to inherit the generic class’s traits.

Object-Oriented Programming has the following advantages:

1. OOP provides a clear modular structure for programs which makes it good for defining abstract datatypes where implementation details are hidden and the unit has a clearly defined interface.
2. OOP makes it easy to maintain and modify existing code as new objects can be created with small differences to existing ones.
3. OOP provides a good framework for code libraries where supplied software components can be easily adapted and modified by the programmer. This is particularly useful for developing graphical user interfaces.

**Q8. What is use case modelling?**

Ans. Use case modelling is a useful tool for requirements elicitation. It provides a graphical representation of the software system's requirements.

The key elements in a use case model are **actors** (external entities), and the **use cases** themselves. In outline, a use case is a unit of functionality (a requirement), or a service, in the system. A use case is not a process, or program, or function.

Because use case models are simple both in concept and appearance, it is relatively easy to discuss the correctness of a use case model with a non-technical person (such as a customer).

Use case modeling effectively became a practicable analysis technique with the publication of Ivar Jacobson's (1991) book “Object-oriented software engineering: a use case driven approach”. Jacobson has continued to promote this approach to system analysis to the present day, and it has now been formalised as part of the UML. However, use case modeling is not very different in its purpose and strategy from earlier techniques, such as structured viewpoint analysis.

**Q9.** Explain 5 views of system in OOAD.

Ans.

***Use Case View***

* Use Case Analysis is a technique to capture business process from user’s perspective.
* Encompasses the behavior as seen by users, analysts and testers.
* Specifies forces that shape the architecture.
* Static aspects in use case diagrams; Dynamic aspects in interaction (statechart and activity) diagrams.

***Design View***

Encompasses classes, interfaces, and collaborations that define the vocabulary of a system.

Supports functional requirements of the system.

Static aspects in class and object diagrams; Dynamic aspects in interaction diagrams.

***Process View***

Encompasses the threads and processes defining concurrency and synchronization.

Addresses performance, scalability, and throughput.

Static and dynamic aspects captured as in design view; emphasis on active classes.

***Implementation View***

Encompasses components and files used to assemble and release a physical system.

Addresses configuration management.

Static aspects in component diagrams; Dynamic aspects in interaction diagrams.

***Deployment View***

Encompasses the nodes that form the system hardware topology.

Addresses distribution, delivery, and installation.

Static aspects in deployment diagrams; Dynamic aspects in interaction diagrams.

**Q10. Differentiate static and dynamic models**

Ans. The most notable difference between static and dynamic models of a system is that while a dynamic model refers to runtime model of the system, static model is the model of the system not during runtime. Another difference lies in the use of differential equations in dynamic model which are conspicuous by their absence in static model. Dynamic models keep changing with reference to time whereas static models are at equilibrium of in a steady state.

Static model is more structural than behavioral while dynamic model is a representation of the behavior of the static components of the system. Static modelling includes class diagram and object diagrams and help in depicting static constituents of the system. Dynamic modelling on the other hand consists of sequence of operations, state changes, activities, interactions and memory.

Static modelling is more rigid than dynamic modelling as it is a time independent view of a system. It cannot be changed in real time and this is why it is referred to as static modelling. Dynamic modelling is flexible as it can change with time as it shows what an object does wwith many possibilities that might arise in time.

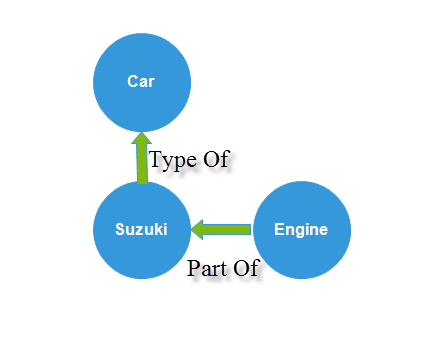
**Q11. Why unified modelling language is needed?**

Ans. Modeling is a proven & well accepted engineering techniques. In building architecture, we develop architectural models of houses & high rises to help visualize the final products. In Unified Modeling Language (UML), a model may be structural, emphasizing the organization of the system or it may be behavioral, emphasizing the dynamics of the system. A model is a simplification of reality, providing blueprints of a system. UML, in specific:

* Permits you to specify the structure or behavior of a system.
* Helps you visualize a system.
* Provides template that guides you in constructing a system.
* Helps to understand complex system part by part.
* Document the decisions that you have made.

We build model so that we can better understand the system we are developing. A model may encompass an overview of the system under consideration, as well as a detailed planning for system design, implementation and testing.

**Q12. What is a part of relationship?**

Ans. Composition is a "part-of" relationship. Simply composition means mean use of instance variables that are references to other objects. In composition relationship both entities are interdependent of each other for example “engine is part of car”, “heart is part of body”.  
  
Let us take an example of car and engine. Engine is a part of each car and both are dependent on each other.  


**Q13. Differentiate between an action and use case.**

* Ans. ***use case*** - a description of set of sequence of actions that a system performs that yields an observable result of value to a particular actor. A use case is used to structure the behavioral things in a model. A use case is realized by a collaboration.

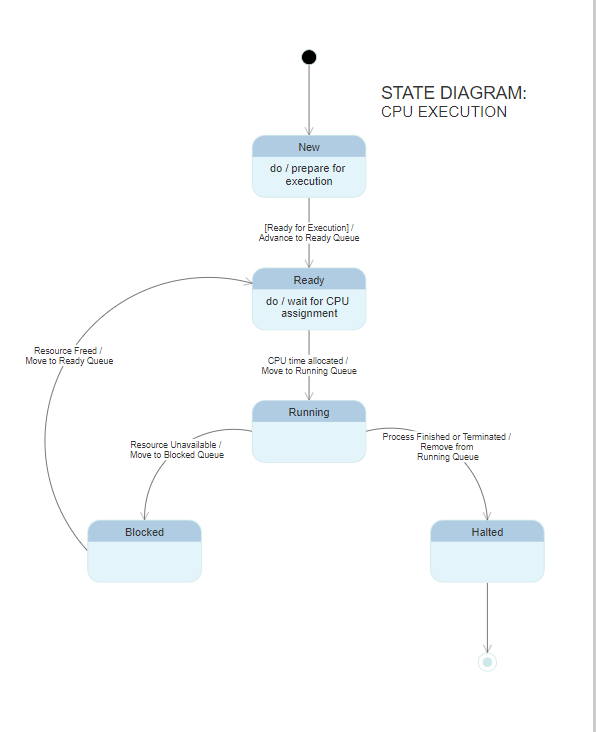
“In the description of a use case, there are descriptions of what happens in the system. The use case description does not define how tasks are performed in the system.” This is a clear confirmation that a use case includes actions of changing the system’s state, and that use case specifications describe them

**Q14. Compare sequence versus collaboration diagram.**

Ans.

| **Sr. No.** | **Key** | **Sequence Diagram** | **Collaboration diagram** |
| --- | --- | --- | --- |
| 1 | Definition | Sequence diagram is the diagram in which main representation is of the sequence of messages flowing from one object to another; also main emphasis is on representing that how the messages/events are exchanged between objects and in what time-order. | On other hand, Collaboration diagram is a diagram in which main representation is of how one object is connected to another implementing the logic behind these objects with the use of conditional structures, loops, concurrency, etc. |
| 2 | Main focus | Sequence diagram mainly focuses to represent interaction between different objects by pictorial representation of the message flow from one object to another object. It is time ordered that means exact interactions between objects is represented step by step. | On other hand Collaboration diagram focus to represent the structural organization of the system and the messages that are sent and received. |
| 3 | Type | As Sequence diagram models the sequential logic, ordering of messages with respect to time so it is categorised as Dynamic modelling diagram. | On other hand Collaboration diagram mainly represent organization of system so it is not classified as Dynamic modelling diagram. |
| 4 | UseCase | Sequence diagram as already mentioned is used to describe the behaviour of several objects in a particular single use case with implementation of all possible logical conditions and flows. | However on other hand Collaboration diagrams is used to describe the general organization of system for several objects in several use cases. |

**Q15. Draw a state diagram to depict the states of the CPU.**

Ans. 

**Q16. Explain about different types of class visibilities.**

Ans. To specify the visibility of a class member (i.e., any attribute or method) these are the following notations that must be placed before the member's name:

"+" **Public**

"-" **Private**

"#" **Protected**

"/" **Derived** (can be combined with one of the

others)

“~" **Static**

**Public Visibility mode:** If we derive a subclass from a public base class. Then the public member of the base class will become public in the derived class and protected members of the base class will become protected in the derived class.

**Protected Visibility mode:** If we derive a subclass from a Protected base class. Then both public member and protected members of the base class will become protected in the derived class.

**Private Visibility mode:** If we derive a subclass from a Private base class. Then both public member and protected members of the base class will become Private in the derived class.

**Q17. What is the purpose of class diagram? Explain.**

Ans. The purpose of class diagram is to model the static view of an application. Class diagrams are the only diagrams which can be directly mapped with object-oriented languages and thus widely used at the time of construction.

UML diagrams like activity diagram, sequence diagram can only give the sequence flow of the application, however class diagram is a bit different. It is the most popular UML diagram in the coder community.

The purpose of the class diagram can be summarized as −

* Analysis and design of the static view of an application.
* Describe responsibilities of a system.
* Base for component and deployment diagrams.
* Forward and reverse engineering.

**Q18. Differentiate between UML and OOAD.**

Ans. Object-oriented analysis and design (OOAD) is a software engineering approach that models a system as a group of interacting objects. Each object represents some entity of interest in the system being modeled, and is characterised by its class, its state (data elements), and its behavior. Various models can be created to show the static structure, dynamic behavior, and run-time deployment of these collaborating objects. There are a number of different notations for representing these models, such as the Unified Modeling Language (UML).  
  
Object-oriented analysis (OOA) applies object-modeling techniques to analyze the functional requirements for a system. Object-oriented design (OOD) elaborates the analysis models to produce implementation specifications. OOA focuses on what the system does, OOD on how the system does it.

Unified Modeling Language (UML) is a standardized specification language for object modeling. UML is a general-purpose modeling language that includes a graphical notation used to create an abstract model of a system, referred to as a UML model.

**Q19. Describe about 7 phases of SDLC.**

### Ans. 1. Planning

This is the first phase in the systems development process. It identifies whether or not there is the need for a new system to achieve a business"s strategic objectives. This is a preliminary plan (or a feasibility study) for a company"s business initiative to acquire the resources to build on an infrastructure to modify or improve a service. The company might be trying to meet or exceed expectations for their employees, customers and stakeholders too. The purpose of this step is to find out the scope of the problem and determine solutions. Resources, costs, time, benefits and other items should be considered at this stage.

**2. Systems Analysis and Requirements**

The second phase is where businesses will work on the source of their problem or the need for a change. In the event of a problem, possible solutions are submitted and analyzed to identify the best fit for the ultimate goal(s) of the project. This is where teams consider the functional requirements of the project or solution. It is also where system analysis takes place—or analyzing the needs of the end users to ensure the new system can meet their expectations. Systems analysis is vital in determining what a business"s needs are, as well as how they can be met, who will be responsible for individual pieces of the project, and what sort of timeline should be expected.

There are several tools businesses can use that are specific to the second phase. They include:

* CASE (Computer Aided Systems/Software Engineering)
* Requirements gathering
* Structured analysis

**3. Systems Design**

The third phase describes, in detail, the necessary specifications, features and operations that will satisfy the functional requirements of the proposed system which will be in place. This is the step for end users to discuss and determine their specific business information needs for the proposed system. It"s during this phase that they will consider the essential components (hardware and/or software) structure (networking capabilities), processing and procedures for the system to accomplish its objectives.

**4. Development**

The fourth phase is when the real work begins—in particular, when a programmer, network engineer and/or database developer are brought on to do the major work on the project. This work includes using a flow chart to ensure that the process of the system is properly organized. The development phase marks the end of the initial section of the process. Additionally, this phase signifies the start of production. The development stage is also characterized by instillation and change. Focusing on training can be a huge benefit during this phase.

**5. Integration and Testing**

The fifth phase involves systems integration and system testing (of programs and procedures)—normally carried out by a Quality Assurance (QA) professional—to determine if the proposed design meets the initial set of business goals. Testing may be repeated, specifically to check for errors, bugs and interoperability. This testing will be performed until the end user finds it acceptable. Another part of this phase is verification and validation, both of which will help ensure the program"s successful completion.

**6. Implementation**

The sixth phase is when the majority of the code for the program is written. Additionally, this phase involves the actual installation of the newly-developed system. This step puts the project into production by moving the data and components from the old system and placing them in the new system via a direct cutover. While this can be a risky (and complicated) move, the cutover typically happens during off-peak hours, thus minimizing the risk. Both system analysts and end-users should now see the realization of the project that has implemented changes.

**7. Operations and Maintenance**

The seventh and final phase involves maintenance and regular required updates. This step is when end users can fine-tune the system, if they wish, to boost performance, add new capabilities or meet additional user requirements.

**Q20. Explain why object oriented approach is preferable when compared to other approaches?**

Ans. In the object-oriented approach, the focus is on capturing the structure and behavior of information systems into small modules that combines both data and process. The main aim of Object Oriented Design (OOD) is to improve the quality and productivity of system analysis and design by making it more usable.

In analysis phase, OO models are used to fill the gap between problem and solution. It performs well in situation where systems are undergoing continuous design, adaption, and maintenance. It identifies the objects in problem domain, classifying them in terms of data and behavior.

The OO model is beneficial in the following ways −

* It facilitates changes in the system at low cost.
* It promotes the reuse of components.
* It simplifies the problem of integrating components to configure large system.
* It simplifies the design of distributed systems.

**Q21. Define association and aggregation.**

Ans. An **association** is defined as an organization of people with a common purpose and having a formal structure. It represents a binary relationship between two objects that describes an activity. It is a relationship between objects. For example, A doctor can be associated with multiple patients

An **aggregation** is a collection, or the gathering of things together. This relationship is represented by a “has a” relationship. In other words, aggregation is a group, body, or mass composed of many distinct parts or individuals For example, phone number list is an example of aggregation.

**Q22. Define Events, States, Signal, and Transition.**

Ans. An **event** is the specification of a significant occurrence that has a location in time and space. Anything that happens is modeled as an **event** in UML. four kinds of **events** – signals, calls, the passing of time, and a change in state. **Events** may be external or internal and asynchronous or synchronous.

* occurrence at a point in time
  + instantaneous
  + often corresponds to verb in past tense
    - e.g., alarm set, powered on
  + or onset of a condition
    - e.g., paper tray becomes empty, temperature drops below freezing

The **state** is an abstraction given by the values of the attributes that the object has at a particular time period. It is a situation occurring for a finite time period in the lifetime of an object, in which it fulfils certain conditions, performs certain activities, or waits for certain events to occur.

A **signal** represents an object that is dispatched (thrown) asynchronously by one object and then received (caught) by another. Exceptions are an example of a kind of **signal**. **Signals** may have instances, although these instances are not typically modelled explicitly.

A **transition** denotes a change in the state of an object. If an object is in a certain state when an event occurs, the object may perform certain activities subject to specified conditions and change the state

**Q23. Define Unified Process (UP). List the 4 phases of UP.**

Ans. Unified Process is based on the enlargement and refinement of a system through multiple iterations, with cyclic feedback and adaptation. The system is developed incrementally over time, iteration by iteration, and thus this approach is also known as iterative and incremental software development. The iterations are spread over four phases where each phase consists of one or more iterations [4]:

•

*Inception—*the first and the shortest phase in the project. It is used to prepare basis for the project, including preparation of business case, establishing project scope and setting boundaries, outlining key requirements, and possible architecture solution together with design tradeoffs, identifying risks, and development of initial project plan*—*schedule with main milestones and cost estimates.

•

*Elaboration—*during this phase the project team is expected to capture a majority of system’s requirements (e.g., in the form of use cases), to perform identified risk analysis and make a plan of risk management to reduce or eliminate their impact on final schedule and product, to establish design and architecture (e.g., using basic class diagrams, [package diagrams](https://www.sciencedirect.com/topics/computer-science/package-diagram), and deployment diagrams), to create a plan (schedule, cost estimates, and achievable milestones) for the next (construction) phase.

•

*Construction—*the longest and largest phase within Unified Process. During this phase, the design of the system is finalized and refined and the system is built using the basis created during [elaboration phase](https://www.sciencedirect.com/topics/computer-science/elaboration-phase). The construction phase is divided into multiple iterations, for each iteration to result in an executable release of the system. The final iteration of construction phase releases fully completed system which is to be deployed during transition phase, and

•

*Transition—*the final project phase which delivers the new system to its end-users. Transition phase includes also data migration from legacy systems and user trainings.

**Q24. Generalize your views about inception in use case.**

* Ans. Goal is obtaining buy-in from all interested parties
* Initial requirements capture
* Cost Benefit Analysis
* Initial Risk Analysis
* Project scope definition
* Development of a disposable prototype
* Initial Use Case Model (10% - 20% complete)
* First pass at a Domain Model

**Q25. Compare Aggregation and Composition.**

Ans.

| **BASIS FOR COMPARISON** | **AGGREGATION** | **COMPOSITION** |
| --- | --- | --- |
| Basic | In aggregation there exhibit a relationship where a child can exist independently of the parent. | In composition the cannot exist independently of the parent. |
| Type of relationship | "has a" | "part of" |
| Association type | Weak association | Strong association |
| UML design symbol | Represented by a hollow diamond next to assembly class. | Represented by a solid diamond next to assembly class. |
| Function | The deletion of assembly doesn't affect its parts. | If the owning class object is deleted, it could significantly affect the containing class object. |

**Q26. Describe the preconditions and post conditions in activity diagram for ATM machine money withdrawal.**

Ans. Preconditions

• The bank Customer must possess a bank card.

• The network connection to the Bank System must be active.

• The system must have at least some cash that can be dispensed.

• The cash withdrawal service option must be available.

Postconditions

• The ATM has returned the card and dispensed the cash to the Customer and the withdrawal is registered on the Customer’s account.

• The ATM has returned the card to the Customer and no withdrawal is registered on the Customer’s account.

• The ATM has returned the card but has not supplied the amount of cash registered as withdrawn from the Customer’s account. The discrepancy is registered in the ATM’s log.

• The ATM has kept the card, no withdrawal has registered on the Customer’s account and the Customer has been notified where to contact for more information.

**Q27. Generalize the use of Sequence Diagram.**

* Ans. A use case diagram shows a set of use cases and actors (a special kind of class) and their

relationships.

* address the static use case view of a system.
* especially important in organizing and modeling the behaviors of a system.

## **Purpose of a Sequence Diagram**

1. To model high-level interaction among active objects within a system.
2. To model interaction among objects inside a collaboration realizing a use case.
3. It either models generic interactions or some certain instances of interaction.

**Q28. Differentiate Class diagram and Interaction diagram.**

* Ans. A class diagram shows a set of classes, interfaces, and collaborations and their relationships.
* most common diagram found in modeling object-oriented systems.
* address the **static design** view of a system.
* Class diagrams that include active classes address the static process view of a system.
* Both **sequence** diagrams and **collaboration** diagrams are kinds of interaction diagrams.
* Arc shows an interaction, consisting of a set of objects and their relationships, including the messages that may be dispatched among them.
* **Interaction diagrams address the dynamic view of a system**

**Q29. Analyze the concepts of Noun Phrase Identification from use cases.**

Ans. Noun Phrase Approach

•Using this method, you have to read through the Use cases, interviews, and requirements specification carefully, looking for noun phrases.)

•Change all plurals to singular and make a list, which can then be divided into three categories.

It is safe to scrap the Irrelevant Classes.

•You must be able to formulate a statement of purpose for each candidate class; if not, simply eliminate it.

•You must then select candidate classes from the other two categories.

**Q30. When to use class diagram?**

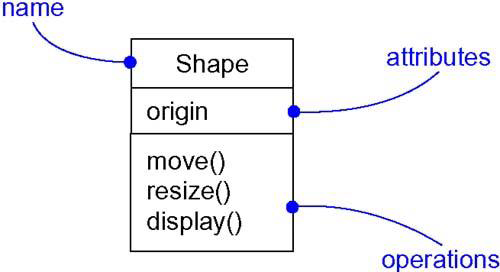
Ans. Class diagram is a static diagram and it is used to model the static view of a system. The static view describes the vocabulary of the system.

Class diagram is also considered as the foundation for component and deployment diagrams. Class diagrams are not only used to visualize the static view of the system but they are also used to construct the executable code for forward and reverse engineering of any system.

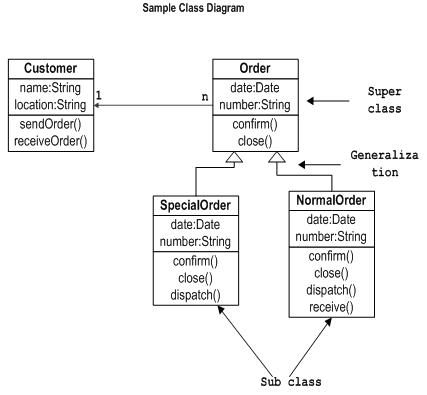
class diagrams are used for −

* Describing the static view of the system.
* Showing the collaboration among the elements of the static view.
* Describing the functionalities performed by the system.
* Construction of software applications using object oriented languages.

**Q31. Describe the UML notation for class diagram with example.**

Ans. 

**Class Diagram for Order System**



**Q32. Describe the concepts of link, association and Inheritance.**

## Ans. **Inheritance**

Inheritance is the mechanism that permits new classes to be created out of existing classes by extending and refining its capabilities. The existing classes are called the base classes/parent classes/super-classes, and the new classes are called the derived classes/child classes/subclasses. The subclass can inherit or derive the attributes and methods of the super-class(es) provided that the super-class allows so. Besides, the subclass may add its own attributes and methods and may modify any of the super-class methods. Inheritance defines an “is – a” relationship.

### Link

A link represents a connection through which an object collaborates with other objects. Rumbaugh has defined it as “a physical or conceptual connection between objects”. Through a link, one object may invoke the methods or navigate through another object. A link depicts the relationship between two or more objects.

### Association

Association is a group of links having common structure and common behavior. Association depicts the relationship between objects of one or more classes. A link can be defined as an instance of an association.

**Q33. Describe the difference between elaboration and inception with an example.**

Ans. Inception Phase

* Goal is obtaining buy-in from all interested parties
* Initial requirements capture
* Cost Benefit Analysis
* Initial Risk Analysis
* Project scope definition
* Development of a disposable prototype
* Initial Use Case Model (10% - 20% complete)
* First pass at a Domain Model

Elaboration Phase

* Requirements Analysis and Capture(deeper)
  + Use Case Analysis
    - Use Case (80% written and reviewed by end of phase)
    - Use Case Model (80% done)
    - Scenarios
      * Sequence and Collaboration Diagrams
      * Class, Activity, Component, State Diagrams
  + Glossary (so users and developers can speak common vocabulary)
  + Domain Model
    - to understand the problem: the system’s requirements as they exist within the context of the problem domain
  + Risk Assessment Plan revised
  + Architecture Document