1908008- OBJECT ORIENTED ANALYSIS AND DESIGN
UNIT I ANSWER KEY
PART-A

1. Define OOAD.

Object-oriented analysis, emphasis on finding and describing the objects—or concepts—in the problem domain. Object-oriented design, emphasis on defining software objects and how they collaborate to fulfill the requirements.

2. What is Analysis and Design?

Analysis: Analysis is the stage where the problem domain is thoroughly understood, requirements are identified, and system behavior is modeled. It aims to answer questions like "What needs to be done?" and "What are the user's needs?"

Design: Design is the phase where the insights from analysis are transformed into a blueprint for the software solution. It specifies the system's structure, components, and interactions, bridging the gap between requirements and implementation. Designers make decisions about architecture, data structures, and interfaces to ensure the solution aligns with identified requirements.

3. Distinguish between method and messages in object.

Messages:

Specify the behavior objects need to perform.
Leave the details of the behavior to the receiver object.
Focus on what actions are needed without specifying how they are carried out.

Methods:

Specify how an operation is to be performed.
Require access to data to execute.
Need detailed knowledge of the data they manipulate.
Directly manipulate data according to the defined behavior.

4. What is the main advantage of Object Oriented Development?

The main advantage of Object-Oriented Development lies in its ability to promote modularity and reusability. By encapsulating data and behavior within objects, OOD enables developers to create components that can be easily reused in different parts of a system or in entirely different projects.

This leads to faster development, easier maintenance, and improved scalability of software systems. Additionally, OOD facilitates better organization and understanding of complex systems through abstraction and encapsulation, enhancing overall software quality and developer productivity.

5. Point out what test can help to find useful use cases?

1. **Choosing the system boundary**: Determine if it's just a software application, the hardware and application combined, or involving users and organizations.
2. **Identifying primary actors:** These are entities with goals fulfilled through the system's services.
3. **Identifying goals for each primary actor:** Understand what each primary actor aims to achieve by interacting with the system.
4. **Defining use cases to satisfy user goals:** Create use cases that align with user goals, naming them according to their respective objectives. Typically, there's a one-to-one correspondence between use cases and user goals, with exceptions to explore.

6. Give the different formats of use cases.

1. **Brief Use Case:** A concise description of the user's interaction with the system, focusing on the essential elements such as actor, trigger, and goal.
2. **Casual Use Case:** Provides a more detailed narrative of the user's actions and system responses, offering a broader understanding of the interaction.
3. **Fully dressed Use Case:** Offers a comprehensive view of the use case, including preconditions, postconditions, main flow, alternate flows, and exceptions.
4. **Use Case Diagram:** A graphical representation of actors, use cases, and their relationships, providing a high-level overview of system functionality.
5. **Activity Diagram:** Represents the flow of activities within a use case, detailing the sequence of actions and decisions made by the actors and the system.

7. What is an object? Give an example

An object is a self-contained unit of code in object-oriented programming (OOP) that encapsulates both data and behavior. Objects interact with each other to perform tasks within a program.

For instance, consider a TV remote control. Its buttons, like power, volume up, volume down, and channel change, act as functions or actions in a program. The actions performed by these buttons represent the behavior of objects or classes in OOP.

For example, when you press the volume up button, the volume increases, demonstrating how objects execute specific actions or behaviors based on user input.

8. What is UML?

UML, or Unified Modeling Language, is a standardized visual language used in software engineering for modeling software systems. It provides a set of graphical notations to represent system structure, behavior, and interactions. UML diagrams aid in communication and documentation of system requirements, design, and architecture.

9. Classify the kinds of actors in use case

Primary Actors: These are the main users or external systems interacting directly with the system to achieve their goals. They initiate use cases and benefit from the system's services.

Secondary Actors: These actors support the primary actors in achieving their goals but do not directly interact with the system. They might provide data or services necessary for the primary actors.

Supporting Actors: These actors assist in the functioning of the system but are not directly involved in achieving specific use case goals. They may include administrative roles, background processes, or external systems.

Offstage Actors: These actors are not directly involved in any use case but still have an interest in the system's behavior or outcomes. They are often stakeholders who influence system requirements or decisions.

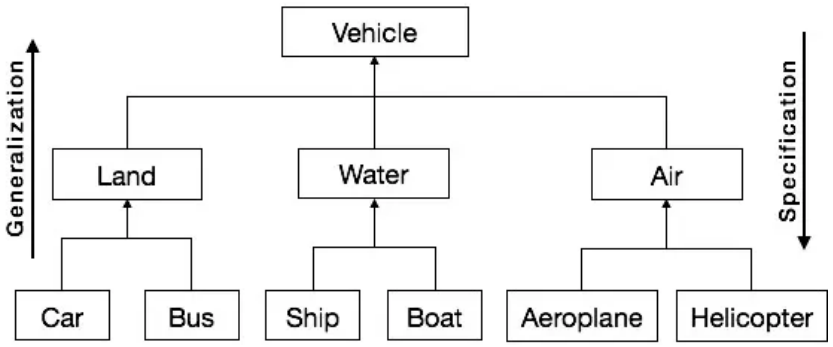
10. Define Unified Process (UP).

The Unified Process has emerged as a popular iterative software development process for building object oriented systems. The Unified Process (UP) combines commonly accepted best practices, such as an iterative lifecycle and risk-driven development, into a cohesive and well-documented description. The best-known and extensively documented refinement of the Unified Process is the Rational Unified Process (RUP). There are 4 phases in Unified Process,

1. Inception
2. Elaboration
3. Construction
4. Transition

11. Illustrate the concepts of Generalization Relationship.

In the generalization process, the common characteristics of classes are combined to form a class in a higher level of hierarchy, i.e., subclasses are combined to form a generalized super-class. It represents an “is – a – kind – of” relationship. For example, “car is a kind of land vehicle”, or “ship is a kind of water Vehicle”.



12. Compare Include and Extend use case relationships.

Include Relationship: Specifies that one use case includes the functionality of another. It's used when a use case requires the behavior described in another use case. The included use case is always executed, representing common behavior shared by multiple other use cases.

Extend Relationship: Indicates optional or conditional behavior that may extend the functionality of a base use case. It's used when a use case may be enhanced by another under specific conditions. The extension use case is optional and only executed under defined circumstances.

13. Describe the POS system and list the components of the POS system.

A **Point of Sale (POS)** system is a computerized system used by businesses to conduct sales transactions, manage inventory, and process payments. It typically consists of both hardware and software components designed to streamline the checkout process and improve overall operational Efficiency.

Components of a POS system:

1. Checkout Terminal (hardware)
2. Point of Sale Software
3. Barcode Scanner
4. Payment Processing System
5. Inventory Management System
6. Customer Relationship Management (CRM)

14. Give the primary goals in the design of UML

UML's primary goal is to establish a standardized notation for all object-oriented methods, integrating the best elements of precursor notations. It's designed to cater to a broad range of applications within software engineering, ensuring versatility and effectiveness across different methodologies and domains.

15. Illustrate the relationship used in Use cases.

Relationship Type	Description	When to Use
Association	Indicates a general association between use cases.	When two or more use cases are loosely related or associated.
Include	Specifies that one use case includes another.	When one use case is essential for the execution of another.
Extend	Represents optional or conditional behavior.	When a use case may extend the functionality of another.
Generalization	Indicates inheritance between use cases.	When a specific use case inherits behavior from a general one.
Dependency	Shows reliance between use cases.	When one use case depends on another indirectly.

16. What are the three ways and perspectives to Apply UML?

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

17. Generalize the concepts of use case modeling

Use case modeling involves conceptualizing and representing interactions between users (actors) and a system to achieve specific goals. Key concepts include:

1. **Actors:** Individuals, systems, or external entities interacting with the system.
2. **Use Cases**: Descriptions of system functionality or behavior from the perspective of actors.
3. **Relationships:** Define interactions between use cases, such as inclusion, extension, and generalization.
4. **Scenarios:** Specific sequences of interactions depicting how use cases are executed in different situations.
5. **System Boundaries**: Define the scope of the system, outlining what's included and excluded.

By encapsulating these elements, use case modeling facilitates understanding, analyzing, and documenting system requirements and behavior effectively.

18. When to use Use cases? Evaluate it

Use-case diagrams are helpful in the following situations: Before starting a project, you can create use-case diagrams to model a business so that all participants in the project share an understanding of the workers, customers, and activities of the business.

Use cases are text documents, not diagrams, and use-case modeling is primarily an act of writing text, not drawing diagrams.**Use cases** are text stories of some actor using a system to meet goals.Here is an example brief format use case:

Process Sale: A customer arrives at a checkout with items to purchase. The cashier uses the POS system to record each purchased item. The system presents a running total and line-item details. The customer enters payment information, which the system validates and records. The system updates inventory. The customer receives a receipt from the system and then leaves with the items.

19. Generalize your views about inception in Use case.

Inception, the initial phase of use case modeling, involves defining the scope of the system and identifying key actors and their goals. It serves as the foundation for subsequent development efforts by establishing a shared understanding among stakeholders. Inception activities typically include gathering requirements, conducting stakeholder interviews, and defining the system boundary. Effective inception lays the groundwork for a successful project by ensuring that the system's purpose and objectives are clearly defined and aligned with user needs and organizational goals.

20. Evaluate and Name the UML diagrams used for the following:

a) Modeling Requirements

b) Modeling Workflows

a) Modeling requirements, the commonly used UML diagrams are Use Case Diagrams and Activity Diagrams. Use Case Diagrams illustrate the interactions between actors and the system to fulfill specific goals, while Activity Diagrams depict the flow of activities or processes within the system.

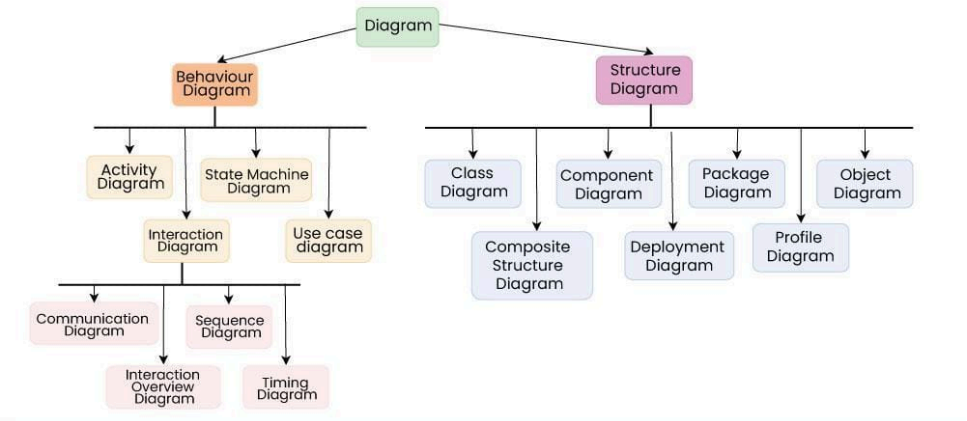
b) Modeling workflows, the commonly used UML diagrams are Activity Diagrams and Sequence Diagrams. Activity Diagrams visualize the flow of activities or tasks within a process, including decision points and parallel activities, while Sequence Diagrams illustrate the interactions and communication between different components or actors within a workflow, showing the chronological order of messages exchanged.

21.What are the 4 phases in UP?

The 4 phases in UP are,

1. Inception
2. Elaboration
3. Construction
4. Transition

22. Classify the UML Diagrams.



23. Evaluate and Name the UML diagrams used for the following:

a) Modeling behavior of an object.

b) Interaction between groups of objects.

a) Modeling the behavior of an object, the commonly used UML diagram is the State Diagram. State diagrams illustrate the different states that an object can be in and the transitions between these states based on events or conditions. They are particularly useful for modeling the lifecycle of objects and capturing their behavior over time.

b) The interaction between groups of objects, the commonly used UML diagram is the Sequence Diagram. Sequence diagrams depict the sequence of messages exchanged between objects or system components over time. They provide a dynamic view of system behavior, showing how objects collaborate to accomplish a particular task or scenario.

24. List the relationships used in class diagrams?