1. **What is OOAD? Explain why OOAD of software system should be preferred.**

**Ans:** Object-oriented analysis and design (OOAD) is a popular technical approach for analyzing and designing an application, system, or business by applying object-oriented programming, as well as using visual modeling throughout the development life cycles to foster better stakeholder communication and product quality.

According to the popular guide Unified Process, OOAD in modern software engineering is best conducted in an iterative and incremental way. Iteration by iteration, the outputs of OOAD activities, analysis models for OOA and design models for OOD respectively, will be refined and evolve continuously driven by key factors like risks and business value.

OOAD of software system should be preferred because

- Focuses on data rather than the procedures as in Structured Analysis.
- The principles of encapsulation and data hiding help the developer to develop systems that cannot be tampered by other parts of the system.
- It allows effective management of software complexity by the virtue of modularity.
- It can be upgraded from small to large systems at a greater ease than in systems following structured analysis.

2. **What is class diagram? How it is different from Object diagram? Draw class diagram for Online Examination System.**

**Ans:** Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. Class diagram describes the attributes and operations of a class and also the constraints imposed on the system.

The class diagrams are widely used in the modeling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

The class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram. 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.

**Different from Object diagram**

A Class Diagram will show what the Objects in your system consist of (members) and what they are capable of doing (methods). In contrast, an Object Diagram will show how objects in your system are interacting with each other at some point in time, and what values those objects contain when the program is in this state. Class diagram is a graph of Classifier elements connected by their various static relationships.

Note that a “class” diagram may also contain interfaces, packages, relationships, and even instances such as objects and links. Perhaps a better name would be “static structural diagram”, but “class diagram” is shorter and well established. Object diagram is a graph of instances, including objects and data values. A static object diagram is an instance of a class diagram; it shows a snapshot of the detailed state of a system at a point in time. The use of object diagrams is fairly limited, mainly to show examples of data structures. In simple words... Class diagram shows a collection of declarative (static) model elements, such as classes, types, and their contents and relationships.

Object diagram encompasses objects and their relationships at a point in time. It may be considered a special case of a class diagram or a collaboration diagram.