Object Oriented Programming (OOP)

- OOP is methodology or paradigm to design a program using classes and objects.
- It simplifies software development and maintenance.

Overview of OOP

Object

- An entity that has state and behavior.
- For eg, a chair, pen, table, etc.
- It can be physical or logical.
- It is an instance(result) of a class.
- Takes some space in heap memory.
- Example:
    - A dog is an object because it has states like color, name, breed, etc.
    - It also has behavior such as wagging tail, eating, barking, etc.

Class

- Blueprint from which you can create objects.
- It is logical entity.
- Logical entity means an entity which represents a general category like Animal, Vehicle, etc.
- Physical entity means an entity which represent an actual object of that category.
- For eg, consider Animal as logical entity, so dog can be a physical entity of the category Animal.
- Similarly, consider Vehicle as logical entity, so bike can be a physical entity of the category Vehicle.
- Class doesn't consume any space.

Inheritance

- When one object acquires all the properties and behavior of a parent object, it is called as inheritance.
- Provides code reusability.

Polymorphism

- Performing a task in different ways.
- In Java, we use method overloading and method overriding to achieve polymorphism.
- In general, assume the act of speaking, humans can speak in English, Spanish, Russian, etc.
- Similarly, a dog speaks whoof or barks, whereas a cat meows.
- So the act of speaking is carried out in different ways.

Abstraction

- Hiding internal details and showing only the functionality.
- For eg, when sending an sms, we know how to send the message (functionality), but we are not aware of the interal process.
- In Java, we use abstract classes and interface to achieve abstraction.

Encapsulation

- Wrapping the data members and methods together into a single unit.

Coupling

- Refers to the information or dependency of another class.
- Arises when classes are aware of each other.
- If a class has detailed information about another class, there is strong coupling.

Cohesion

- Refers to the level of a component which performs a single well-defined task.
- A single well-defined task is done by a highly cohesive method.
- A weakly cohesive method will split the task into seperate parts.
- The `java.io` package is highly cohesive package because it has only i/o related classes and interfaces.
- The `java.util` package is a weakly cohesive package as it has unrelated classes and interfaces.

Association

- Represents relationship between the objects.
- An object can be associated with one object or many objects.
- Four types of association:
  - One to One
  - One to Many
  - Many to One
  - Many to Many
- For example: A country can have only one Prime Minister (One to One).
- A Prime Minister can have many Ministers (One to Many).
- Many MP's can have one Prime Minister (Many to One).
- Ministers can have many departments (Many to Many).
- Association can be unidirectional or bidirectional.

Aggregation

- A way to achieve association.
- Represents the relationship where one object contains other objects as a part of its state.
- Represents weak relationship between object.
- Also termed as 'HAS-A' relationship.
- Another way to reuse objects.

Composition

- Another way to achieve association.
- Represents the relationship where one object contains other objects as a part of its state.
- Strong relationship between the containing object and the dependent object.
- If you delete parent object, child elements will be deleted automatically.

OOP vs procedure-oriented programming language

- OOP make development and maintanence easier, where in procedure oriented, it is not easy to manage if code grows.

- OOP provides data hiding, whereas in procedure-oriented languages, the global data can be accessed from anywhere.
- OOP provides the ability to simulate real-world event much more effectively.