Object-Oriented Programming (OOP) Concepts

Object-Oriented Programming (OOP) is a programming paradigm that uses objects and classes to design and develop applications. It helps manage complexity by structuring software into reusable, interconnected objects.

Core Concepts of OOP:

1. Classes and Objects

- **Classes**: A blueprint for creating objects, defining attributes (properties) and methods (functions). For example, a "Car" class defines characteristics like make, model, and year.
- **Objects**: Instances of classes. They are specific examples with actual values. For example, a 2020 Toyota Corolla is an object of the "Car" class.

2. Encapsulation

- **Definition**: Encapsulation bundles data (attributes) and methods (functions) that operate on the data into a single unit, or class.
- **Purpose**: It restricts direct access to some of an object's components, protecting the integrity of the data. This is achieved by making some attributes private and providing public methods to access and modify them.

3. Inheritance

- **Definition**: Inheritance allows one class (child class) to inherit attributes and methods from another class (parent class).
- **Purpose**: Promotes code reusability and establishes a hierarchical relationship between classes. The child class can inherit and extend the behavior of the parent class.

4. Polymorphism

- **Definition**: Polymorphism means "many forms." It allows objects of different classes to be treated as objects of a common superclass.
- **Purpose**: Enables the same method to perform different actions based on the object it is acting upon. This is achieved through method overriding (a child class provides a specific implementation of a method already defined in its parent class) and method overloading (multiple methods with the same name but different parameters).

5. Abstraction

- **Definition**: Abstraction involves hiding complex implementation details and showing only the essential features of an object.

- **Purpose**: Simplifies interaction with objects by exposing only necessary components, making it easier to use and understand. Abstraction focuses on what an object does rather than how it does it.

Summary

OOP concepts help in creating structured, modular, and reusable code. By using classes and objects, encapsulation, inheritance, polymorphism, and abstraction, developers can design complex systems that are easier to manage and extend