**1.What is OOP? List OOP concepts**

OOP stands for Object-Oriented Programming. It is a programming paradigm based on the concept of "objects", which can contain data (attributes) and code (methods). Here are some key concepts of OOP:

**Class:** A blueprint for creating objects. It defines the attributes and methods that will be associated with objects created from it.

**Object:** An instance of a class. It is a concrete realization of the class blueprint, possessing its own unique data and behavior.

**Encapsulation:** Encapsulation refers to the bundling of data (attributes) and methods (functions) that operate on the data into a single unit, typically a class. It hides the internal state of an object from the outside world and only exposes necessary functionalities through methods.

**Inheritance:** Inheritance is a mechanism by which a class can inherit properties and behavior from another class, known as the superclass or parent class. The class inheriting from the superclass is called the subclass or child class. This allows for code reuse and the creation of hierarchical relationships between classes.

**Polymorphism:** Polymorphism means the ability to present the same interface for different data types or objects. In OOP, polymorphism allows objects of different classes to be treated as objects of a common superclass through method overriding or method overloading.

**Abstraction:** Abstraction involves focusing on the essential characteristics of an object while hiding its complex implementation details. It allows developers to create models that capture the important aspects of real-world entities without getting bogged down in unnecessary details.

**Association:** Association represents a relationship between two or more objects where they are connected by some form of link. It can be one-to-one, one-to-many, or many-to-many. Associations are typically represented as instance variables within the classes involved.

**Composition:** Composition is a form of association where one class contains objects of another class as part of its state. It implies a stronger relationship between the containing class (composite) and the contained class (component) than just association.

**2.What is the difference between OOP and POP?**

**Main Focus:**

**OOP:** The main focus of OOP is on objects and their interactions. It emphasizes the concept of objects, which encapsulate data and behavior, and classes, which are blueprints for creating objects.

**POP:** The main focus of POP is on procedures or functions. It emphasizes the use of procedures, which are sequences of instructions to be executed step by step.

**Data and Behavior:**

OOP: In OOP, data (attributes or properties) and behavior (methods or functions) are encapsulated within objects. Objects communicate with each other by sending messages and invoking methods.

POP: In POP, data and behavior are often separated. Data is stored in variables and manipulated by procedures or functions.

**Abstraction:**

**OOP:** OOP emphasizes abstraction through the use of classes and objects. Abstraction allows developers to model real-world entities by focusing on their essential characteristics while hiding unnecessary details.

**POP:** While abstraction can also be achieved in POP, it is typically less emphasized compared to OOP. Procedural code tends to be more explicit and may involve direct manipulation of data structures.

**Code Reusability:**

**OOP:** OOP promotes code reusability through the use of inheritance and composition. Inheritance allows classes to inherit properties and behavior from other classes, while composition involves combining multiple classes to create more complex objects.

**POP:** Code reusability in POP is typically achieved through the use of functions or procedures. Functions can be called multiple times from different parts of the program to perform a specific task.

**Encapsulation:**

**OOP:** OOP emphasizes encapsulation, which involves bundling data and methods that operate on the data into a single unit (class). Encapsulation helps to hide the internal state of an object and only expose necessary functionalities through methods.

**POP:** While encapsulation can also be achieved in POP, it is often less strict compared to OOP. Data and functions in POP may not always be tightly coupled or encapsulated within a single unit.