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

* OOP, or Object-Oriented Programming, is a programming paradigm that uses objects, which are instances of classes, to design and structure software. In OOP, the key concepts revolve around modeling real-world entities and their interactions. Some of the fundamental OOP concepts include:

1. Class:A class is a blueprint or template that defines the structure and behavior of objects. It specifies attributes (data members) and methods (functions) that the objects of the class will have.
2. Object: An object is an instance of a class. It represents a real-world entity and encapsulates data (attributes) and behavior (methods) related to that entity.
3. Encapsulation: Encapsulation is the concept of bundling data (attributes) and the methods (functions) that operate on that data into a single unit, i.e., a class. It hides the internal details of how the class works and provides a controlled interface for interacting with objects.
4. Inheritance: Inheritance allows a class to inherit properties and behavior (attributes and methods) from another class, known as the base class or superclass. It promotes code reuse and the creation of hierarchies of classes.
5. Polymorphism:Polymorphism allows objects of different classes to be treated as objects of a common base class. It enables flexibility in method calls, as objects can respond differently to the same method based on their actual class.
6. Abstraction: Abstraction is the process of simplifying complex reality by modeling classes based on the essential properties and behaviors of real-world entities. It focuses on what an object does rather than how it does it.
7. Method Overloading: Method overloading is a form of polymorphism where multiple methods in the same class have the same name but differ in their parameter lists. The correct method is selected at compile-time based on the arguments provided during the method call.
8. Method Overriding: Method overriding is another form of polymorphism where a subclass provides a specific implementation of a method that is already defined in its superclass. The overridden method in the subclass is executed instead of the method in the superclass when called on an object of the subclass.
9. Constructor: A constructor is a special member function of a class used to initialize objects when they are created. It has the same name as the class and is automatically called when an object is instantiated.
10. Destructor: A destructor is a special member function of a class used to clean up resources when an object is destroyed or goes out of scope. It has the same name as the class preceded by a tilde (~) and is automatically called when an object is no longer needed.
11. Composition: Composition is a way to create complex objects by combining simpler objects as their parts. It allows for building complex structures from smaller, reusable components.
12. Aggregation: Aggregation is a form of association where one class represents a whole and another class represents part of that whole. It models "has-a" relationships between classes.
13. Association: Association represents a relationship between two or more classes, typically involving the use of one class's objects within another class. It can be a "has-a" or "uses-a" relationship.

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

* OOP promotes modularity, reusability, and maintainability in software development by organizing code into classes and objects, which correspond to real-world entities and their interactions.
* Object-Oriented Programming (OOP) and Procedural Programming (POP) are two different programming paradigms, each with its own approach to structuring code and solving problems. Here's a comparison between OOP and POP in two sections:

1. Programming Approach\*\*:

1. Object-Oriented Programming (OOP):

- OOP is based on the concept of objects, which represent real-world entities or abstractions.

- It focuses on modeling data (attributes) and the operations that can be performed on that data (methods) as self-contained units called classes.

- Objects are instances of classes and encapsulate data and behavior.

- OOP promotes code reusability through inheritance and polymorphism.

- It encourages the use of encapsulation to hide the internal implementation details of classes.

- OOP is often used for designing complex systems where data and behavior need to be closely tied together, and it allows for modeling real-world relationships effectively.

II.Procedural Programming (POP):

- POP is based on procedures or functions that define a sequence of steps to be executed.

- It emphasizes a linear flow of control, where the program is divided into functions, and the main program calls these functions to perform tasks.

- Data and functions are separate, and data is often global, which can lead to issues with data integrity and reusability.

- POP does not have built-in support for concepts like classes, objects, inheritance, and polymorphism.

- It is often used for simpler, straightforward tasks where data processing and control flow are the primary concerns.

- POP is suitable for algorithms and programs that are more task-oriented and less focused on modeling real-world entities.

2. Code Organization:

1. Object-Oriented Programming (OOP):

- Code in OOP is organized around classes and objects.

- Classes define the structure (attributes and methods) of objects, and objects are instances of those classes.

- Encapsulation is used to hide the internal details of classes, and access to data is controlled through public methods (getters and setters).

- Inheritance allows for creating hierarchies of classes and promotes code reuse.

- Polymorphism enables objects of different classes to respond to the same method in a way that is specific to their class.

II.Procedural Programming (POP):

- Code in POP is organized around functions or procedures.

- Functions define specific tasks or operations and are called sequentially from the main program.

- Data is often global or passed as function arguments, leading to potential data integrity issues.

- There is no concept of classes, objects, or inheritance.

- POP follows a more top-down or linear structure, where the focus is on the execution of a sequence of steps or procedures.

* In summary, OOP and POP are two different programming paradigms that approach software development in distinct ways. OOP focuses on modeling real-world entities, promoting code reusability, and encapsulating data and behavior within objects. POP, on the other hand, emphasizes a linear sequence of procedures and is often used for simpler, task-oriented programs. The choice between OOP and POP depends on the specific requirements and complexity of the project at hand.