**MODUAL:4**

**OOPS Concept**

**2.What is OOP? List OOP Concepts**

**OOP:** OOP stands for object-oriented programming (OOP).

OOP is a computer programming model that organizes software design around data, or object, rather than functions and logic.

An object can be defined as a data field that has unique attributes and behavior.

OOP focuses on the objects that developers want to manipulate rather than the logic required to manipulate them.

This approach to programming is well-suited for programs that are large, complex and actively updated or maintained.

This includes programs for manufacturing and design, as well as mobile application.

OOP allows decomposition of program into a number of entities called objects and then builds data and function around these objects.

Object-Oriented Programming treats data as a critical element in the program development and does not allows it flow freely around system.

* **Basic OOP Concepts:**

Some of basic concepts of object-oriented programming are:

* polymorphism
* inheritance
* Dynamic Binding
* Data abstraction and Encapsulation
* class
* object

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

|  |  |  |
| --- | --- | --- |
| Feature | Object-Oriented programming(c++) | Procedural Programming  (c language) |
| Key Management | Objects and their inheritance | Procedures and function |
| Data Management | Data and behavior are encapsulated in objects | Data and behavior are separate entities |
| Abstraction | Encourages the use of abstract classes and interfaces | Does not emphasize abstraction |
| Inheritance | Supports inheritance allowing  Classes to inherit properties and methods | Does not support inheritance |
| Polymorphism | Allows objects of different types to be treated as the same type | Does not provide inheritance polymorphism |
| Code Reusability | High level of code reusability through inheritance and composition | Relies on function and subroutines for code reusability |
| Code Organization | Follows a modular approach with objects as self-contained modules | Relies on function and procedures for code organization |
| Flexibility | Provides flexibility through polymorphism and dynamic binding | Relies on structured programming offering less flexibility |
| Complexity Management | Encourages managing complexity through encapsulation and abstraction | Manages complexity through modular code organization and stepwise design |
| Real-World Modeling | Well-suited for modeling real world entities and their interactions | May not align well with real-world modeling focusing more on processes |