6. Introduction to Object-Oriented Programming

1. Explain the key concepts of Object-Oriented Programming (OOP).

Object-Oriented Programming is based on objects and classes.

The main concepts of OOP are encapsulation, inheritance, polymorphism, and abstraction.

Encapsulation means binding data and functions together.

It provides data security.

Inheritance allows one class to acquire properties of another class.

It improves code reusability.

Polymorphism allows functions to behave differently in different situations.

Abstraction hides internal implementation details.

OOP represents real-world entities as objects.

It improves program structure and organization.

OOP reduces code duplication.

It increases flexibility and scalability.

OOP makes debugging easier.

It supports modular programming.

It is suitable for large projects.

Therefore, OOP is a powerful programming paradigm.

2. What are classes and objects in C++? Provide an example.

A class is a blueprint for creating objects.

It defines data members and member functions.

An object is an instance of a class.

Classes help in organizing code.

They combine data and functions together.

Objects use class properties.

Multiple objects can be created from one class.

Classes improve code reusability.

They provide data security.

Example: class Student { int roll; };.

Here Student is a class.

Objects are created using class name.

Example: Student s1;.

Classes are user-defined data types.

They are fundamental in OOP.

Therefore, classes and objects are core components of C++.

3. What is inheritance in C++? Explain with an example.

Inheritance is a feature of OOP that allows one class to inherit properties of another class.

The existing class is called base class.

The new class is called derived class.

Inheritance promotes code reusability.

It reduces redundancy.

Derived class can access public members of base class.

It improves hierarchical classification.

There are different types of inheritance.

Examples include single, multiple, and multilevel inheritance.

Inheritance improves maintainability.

It supports real-world relationships.

Example: class Student : public Person {}; shows inheritance.

Inheritance is widely used in large applications.

It makes programs efficient and organized.

Therefore, inheritance is an important OOP concept.

4. What is encapsulation in C++? How is it achieved in classes?

Encapsulation means wrapping data and functions together in a single unit.

It is achieved using classes.

Data members are kept private.

Member functions are kept public.

Private data cannot be accessed directly.

It can only be accessed through public functions.

This provides data security.

Encapsulation prevents unauthorized access.

It improves program reliability.

It makes code structured.

It supports modular programming.

Access specifiers like private, public, and protected are used.

Encapsulation hides internal implementation.

It improves maintainability.

It ensures better control over data.

Therefore, encapsulation is a key concept in OOP.