Module 3 – Inheritance & Polymorphism Summary

1. Inheritance

Definition: Mechanism where a class (derived) acquires properties and behaviors of another (base).
Advantages: Code reusability, easier maintenance.
Types of inheritance:

- Single: One base, one derived.
- Multilevel: Chain of inheritance.
- Hierarchical: Multiple derived from one base.
- Multiple: Derived from multiple bases.
- Hybrid: Combination of above.

2. Derived Classes & Visibility

Syntax: class Derived : [visibility] Base { ... }
Visibility modes:

- public: Base public → derived public.
- private: Base public → derived private.
- protected: Accessible in derived.

Private members are not inherited.

3. Inheritance Examples

• Single Level: Derived inherits fields/methods of single base.
• Multilevel: A → B → C chain; C gets all ancestors' members.
• Multiple: class C : public A, public B; resolves ambiguity with scope operator.
• Hierarchical: class B,C,D : public A.
• Hybrid: Combines multiple types, e.g., D inherits B and C where B inherits A.

4. Polymorphism

Means 'many forms'; achieved via:
  • Compile-time (static): Function/operator overloading.
  • Run-time (dynamic): Method overriding via virtual functions.
Function Overloading: Same name, different params.
Operator Overloading: Custom behavior for operators.

5. Virtual Functions & Abstraction

Virtual: Declared in base; allows late binding.
Base pointer pointing to derived invokes derived method.
Pure Virtual: virtual func() = 0; makes class abstract.
Abstract Base Class: Cannot instantiate; used for interfaces.