Polymorphism in OOP:

1. **Definition**:

- **Polymorphism** allows one object to take on many forms. It enables a reference variable of a parent class to refer to an object of its subclass.

2. **Types of Polymorphism**:

- **Compile-time (Static)**: Method overloading (same method name, different parameters).

- **Run-time (Dynamic)**: Method overriding (subclass provides a specific implementation of a method already defined in the parent class).

**Method Access in Polymorphism**:

1. **Reference** **Type Determines Accessible Methods:**

- The **type of the reference variable** (not the object) determines which methods can be called.

- Example: `Animal animal = new Cat();` allows access only to methods in `Animal` (parent class).

2. **Overridden Methods Are Resolved at Runtime:**

- When a method is **overridden** in the child class, calling that method on a parent reference (e.g., `animal.C()`) will execute the child class's version at runtime, due to **dynamic method dispatch**.

3. **Subclass-Specific Methods Not Accessible via Parent Reference:**

- You cannot call methods exclusive to the subclass (`Cat`) using a parent reference (`Animal`), e.g., `animal.D()` will throw an error.

4. **Accessing All Methods Through Subclass Reference:**

- If you create an object of the subclass directly (e.g., `Cat c = new Cat();`), you can access all methods, including inherited and subclass-specific methods (`A`, `B`, `C`, `D`, `E`).

# Important Rules:

**- Method Overloading vs. Overriding:**

- **Overloading**: Same method name but different signatures, resolved at compile time.

- **Overriding**: Redefining a parent class method in a subclass with the same signature, resolved at runtime.

**- Upcasting Limits Method Access:**

- When a subclass object is referenced by a parent type (upcasting), only methods in the parent class are accessible, but overridden methods will execute the subclass version.

# Key Terms:

- **Upcasting**: Referring to a subclass object using a parent class reference.

- **Dynamic Method Dispatch:** The process by which overridden methods are called based on the actual object type at runtime.