Polymorphism — Many form.

→ Same obj having different behaviour.

→ implement using method overriding.

Types

- Runtime
- Compile time

java — Copy code

```java
public class Animal {
    public void makeSound() {
        System.out.println("The animal makes a sound.");
    }
}

public class Dog extends Animal {
    public void makeSound() {
        System.out.println("The dog barks.");
    }
}

public class Cat extends Animal {
    public void makeSound() {
        System.out.println("The cat meows.");
    }
}

public class Main {
    public static void main(String[] args) {
        Animal animal1 = new Animal();
        Animal animal2 = new Dog();
        Animal animal3 = new Cat();

        animal1.makeSound(); // prints "The animal makes a sound."
        animal2.makeSound(); // prints "The dog barks."
        animal3.makeSound(); // prints "The cat meows."
    }
}
```

Encapsulation

- Declaring variables as private
- Modifying we use Getter & Setter method as public
- For Data hiding
- Maintain Integrity
- Also help in abstraction

```java
public class Employee {
    private String name;
    private int age;
    private double salary;

    public void setName(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }

    public void setAge(int age) {
        this.age = age;
    }

    public int getAge() {
        return age;
    }

    public void setSalary(double salary) {
        this.salary = salary;
    }

    public double getSalary() {
        return salary;
    }
}
```

// Setter

// Getter

Abstraction

-> process of hiding the implementation details from the user.

-> Only the highlighted set of services provided to the user.

-> Provides a way to implement polymorphism

Adv

-> security

-> Enhancement

Abstraction

- -> abstract class
- -> Interface

Abstract Class

→ Declared using "abstract" keyword

→ Contain both abstract & Non abstract Methods

→ Can have Constructor static Method just like any other Class

→ have feild & Variable

→ We Can't make obj of abstract Class.

→ To use an abstract class you have to inherit it from sub classes.

Interface

→ Collection of abstract method

→ Use of "interface" keyword & implements

→ By default method are public & abstract

→ Variables public + static + final

→ Interface method must be overriden inside the implementing classes.

```java
public interface Shape {
    double getArea();
    double getPerimeter();
}

public class Circle implements Shape {
    private double radius;

    public Circle(double radius) {
        this.radius = radius;
    }

    public double getArea() {
        return Math.PI * radius * radius;
    }

    public double getPerimeter() {
        return 2 * Math.PI * radius;
    }
}
```

It should be public

Object child class ka बाप है।।