Experiment-6

(Applications and implimentation of polymorphism in java)

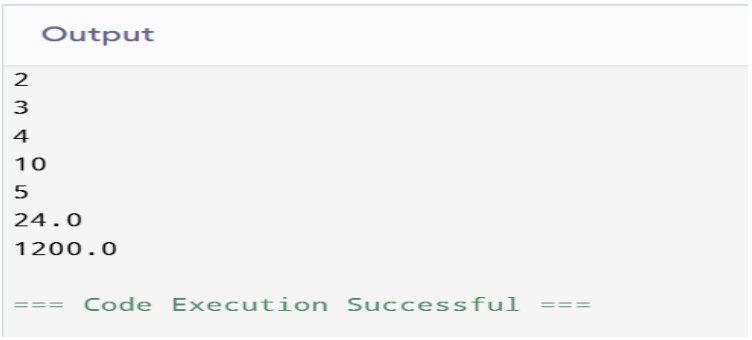
Problem 6.1: 1. A class 'Box' which contains methods and other details (width, height, depth) to calculate volume. Another class 'Boxweight' contains method for finding cost for shipping the box using weight of the box (formula to find cost of shipping distance in km * volume * cost per km) Take cost per KM by the user. WAP to find the volume of box and cost to ship using a single object.

Solution6.1:

```
import java.util.Scanner;
class Box {

double width;

double height;

double depth;

   Box(double width, double height, double depth) {

      this.width = width;

      this.height = height;

      this.depth = depth;

   }
   double volume() {
      return width * height * depth;
   }
}
class BoxWeight extends Box {

   double distance;

   BoxWeight(double width, double height, double depth, double distance) {

      super(width, height, depth);

      this.distance = distance;
   }
   double shippingCost(double costPerKm) {

      return distance * volume() * costPerKm;

   }
}
public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      double width = sc.nextDouble();

      double height = sc.nextDouble();

      double depth = sc.nextDouble();

      double distance = sc.nextDouble();

      double costPerKm = sc.nextDouble();
```

```
        BoxWeight box = new BoxWeight(width, height, depth, distance);
        System.out.println(box.volume());
        System.out.println(box.shippingCost(costPerKm));
    }
}
```

Output:

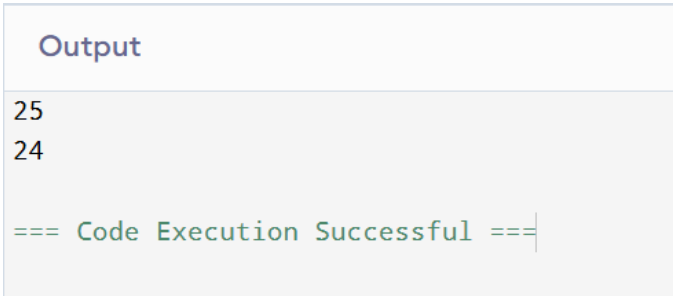


Problem 6.2:

```
class Shape {
    void find_Area() {
    }
}
class Square extends Shape {
    int side;
    Square(int side) {
        this.side = side;
    }
    void find_Area() {
        System.out.println(side * side);
    }
}
class Rectangle extends Shape {
    int length;
    int breadth;
    Rectangle(int length, int breadth) {
        this.length = length;
        this.breadth = breadth;
    }
    void find_Area() {
        System.out.println(length * breadth);
    }
}
public class Main {
    public static void main(String[] args) {
        Shape s1 = new Square(5);
        Shape s2 = new Rectangle(4, 6);
        s1.find_Area();
```

```
        s2.find_Area();
    }
}
```

Output:



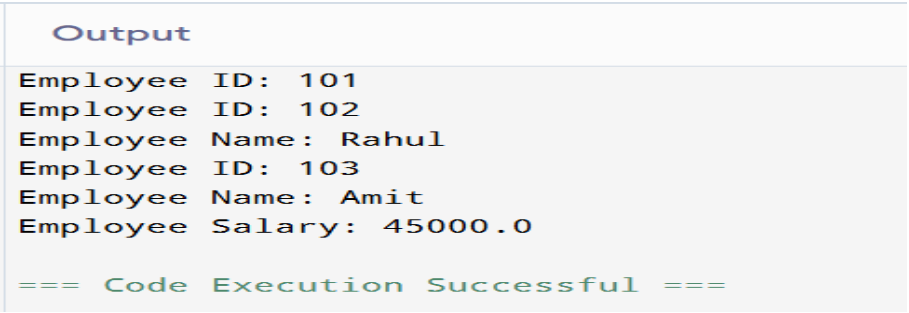
Problem 6.3: 3. Demonstrate constructor overloading with an example of employee class to print different details of an employee based on the parameters passed.

Solution 6.3:

```
class Employee {
    int id;
    String name;
    double salary;

    Employee(int id) {
        this.id = id;
        System.out.println("Employee ID: " + id);
    }
    Employee(int id, String name) {
        this.id = id;
        this.name = name;
        System.out.println("Employee ID: " + id);
        System.out.println("Employee Name: " + name);
    }
    Employee(int id, String name, double salary) {
        this.id = id;
        this.name = name;
        this.salary = salary;
        System.out.println("Employee ID: " + id);
        System.out.println("Employee Name: " + name);
        System.out.println("Employee Salary: " + salary);
    }
}
public class Main {
    public static void main(String[] args) {
```

```
        new Employee(101);

        new Employee(102, "Rahul");
        new Employee(103, "Amit", 45000);
    }
}
```

Output:

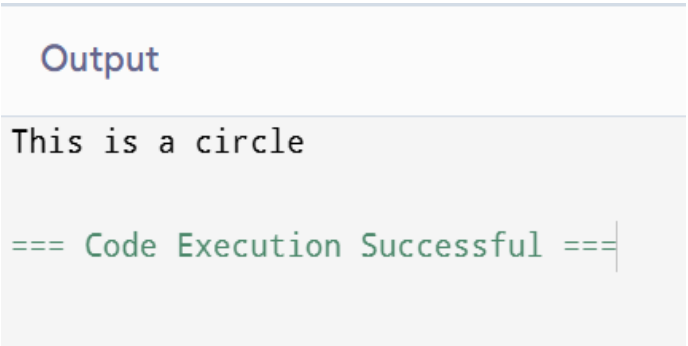


Problem 6.4: 4. A class 'shape' contains a method 'about'. Its subclass 'circle' also has method 'about' but different body. Is it possible to create an object of a circle but give its reference as shape? Demonstrate via example and provide suitable output.

Solution 6.4:

```
class Shape {
    void about() {
        System.out.println("This is a shape");
    }
}
class Circle extends Shape {
    void about() {
        System.out.println("This is a circle");
    }
}
public class Main {
    public static void main(String[] args) {
        Shape s = new Circle();
        s.about();
    }
}
```

Output:



Name: Aditya Kumar Barnwal

Regd_no:2401020031

Group and Sem :1st & 4th