TEST - 4 (CSA0961 – JAVA)

SET – 1 (18-07-2024)

1. Create a base class called Shape with a virtual function area(). Derive two classes Rectangle and Circle from the base class. Implement the area() function for each class.

PROGRAM :

```
abstract class Shape {
    public abstract double area();
}

class Rectangle extends Shape {
    private double width, height;

    public Rectangle(double w, double h) {
        width = w;
        height = h;
    }

    @Override
    public double area() {
        return width * height;
    }
}

class Circle extends Shape {
    private double radius;
    private final double PI = 3.14159;

    public Circle(double r) {
        radius = r;
    }

    @Override
    public double area() {
        return PI * radius * radius;
```

```
    }
}

public class Main {
    public static void main(String[] args) {
        Rectangle rectangle = new Rectangle(4.0, 5.0);
        Circle circle = new Circle(3.0);

        Shape shape1 = rectangle;
        Shape shape2 = circle;

        System.out.println("Area of Rectangle: " + shape1.area());
        System.out.println("Area of Circle: " + shape2.area());
    }
}
```

OUTPUT:

```
Area of Rectangle: 20.0
Area of Circle: 28.274309999999996
```

2 .Create a base class called Animal with a virtual function speak(). Derive two classes Cat andDog from the base class. Implement the speak() function for each class.

PROGRAM:

```
abstract class Animal {
    public abstract void speak();
}

class Cat extends Animal {
    @Override
    public void speak() {
        System.out.println("Meow");
    }
}

class Dog extends Animal {
    @Override
    public void speak() {
```

```
        System.out.println("Woof");
    }
}

public class Main {
    public static void main(String[] args) {
        Animal cat = new Cat();
        Animal dog = new Dog();

        System.out.print("Cat says: ");
        cat.speak();

        System.out.print("Dog says: ");
        dog.speak();
    }
}
```

OUTPUT:

```
Cat says: Meow
Dog says: Woof
```

3. Create a base class called Employee with a virtual function calculatePay(). Derive two classes Manager and Engineer from the base class. Implement the calculatePay() function for each class.

PROGRAM:

```
abstract class Employee {
    private String name;
    private int employeeId;

    public Employee(String name, int employeeId) {
        this.name = name;
        this.employeeId = employeeId;
    }

    public abstract double calculatePay();

    public String getName() {
```

```
        return name;
    }

    public int getEmployeeId() {
        return employeeId;
    }
}

class Manager extends Employee {
    private double salary;

    public Manager(String name, int employeeId, double salary) {
        super(name, employeeId);
        this.salary = salary;
    }

    @Override
    public double calculatePay() {
        return salary;
    }
}

class Engineer extends Employee {
    private double hourlyRate;
    private int hoursWorked;

    public Engineer(String name, int employeeId, double hourlyRate, int
hoursWorked) {
        super(name, employeeId);
        this.hourlyRate = hourlyRate;
        this.hoursWorked = hoursWorked;
    }

    @Override
    public double calculatePay() {
        return hourlyRate * hoursWorked;
    }
}

public class Main {
```

```
    public static void main(String[] args) {
        // Create instances of Manager and Engineer
        Manager manager = new Manager("Alice", 101, 5000.0);
        Engineer engineer = new Engineer("Bob", 201, 30.0, 40);

        System.out.println("Employee Name: " + manager.getName() + ",
Employee ID: " + manager.getEmployeeId() + ", Pay: $" +
manager.calculatePay());
        System.out.println("Employee Name: " + engineer.getName() + ",
Employee ID: " + engineer.getEmployeeId() + ", Pay: $" +
engineer.calculatePay());
    }
}
```

OUTPUT :