ASSIGNMENT-3

Q1. Write a Java program to create a class called Animal with a method called makeSound(). Create a subclass called Dog that overrides the makeSound() method to bark.

- Code:

```
class Animal{

    public void makeSound(){

        System.out.println("Sound of the Animal");

    }

}

class Dog extends Animal{

    public void makeSound() {

        System.out.println("Bark.");

    }

}

public class assignment3_1 {

    public static void main(String[] args) {

        Dog d1 = new Dog();

        d1.makeSound();

    }

}
```

- Output:

```
Bark.
```

ASSIGNMENT-3

Q2. Write a Java program to create a class known as "BankAccount" with methods called deposit() and withdraw(). Create a subclass called SavingsAccount that overrides the withdraw() method to prevent withdrawals if the account balance falls below one hundred.

- Code:

```
class BankAccount{
    int ac_no,bal;
    String name,type;
    public void deposit(int amount){
        System.out.println("Amount deposited.");
        bal += amount;
        System.out.println("Balance is: " + bal);
    }
    public void withdraw(int amount){
        System.out.println("Amount withdraw.");
        bal -= amount;
        System.out.println("Balance is: " + bal);
    }
}
class Saving extends BankAccount{
    Saving(int ac,int balance,String n){
        ac_no = ac;
        bal = balance;
        name = n;
        type = "Saving";
    }
    public void withdraw(int amount) {
        if((bal - amount) <= 100){
            System.out.println("You cant withdraw amount.");
        }
        else {
            System.out.println("Amount withdraw.");
            bal -= amount;
            System.out.println("Balance is: " + bal);
        }
    }
}
public class assignment3_2 {
    public static void main(String[] args) {
        Saving ashish = new Saving(101,1000,"Ashish");
        ashish.deposit(2500);
        ashish.withdraw(3490);
        }
    }
```

- Output:

```
Amount deposited.
Balance is: 3500
You cant withdraw amount.
```

ASSIGNMENT-3

Q3. Write a Java program that creates a class hierarchy for employees of a company.The base class should be Employee, with subclasses Manager, Developer, and Programmer.Each subclass should have properties such as name, address, salary, and job title.Implement methods for calculating bonuses, generating performance reports, and managing projects.

- ➢ Code:

```
class Employee {
    String name, address, jobTitle;
    double salary;
    public Employee(String name, String address, double salary,
String jobTitle) {
        this.name = name;
        this.address = address;
        this.salary = salary;
        this.jobTitle = jobTitle;
    }
    public double calculateBonus() {
        return 0.0;
    }
    public String generatePerformanceReport() {
        System.out.println("Good");
        return "Performance report for " + name;
    }
    public void manageProject(String projectName) {
        System.out.println(jobTitle + " " + name + " is managing
the project: " + projectName);
    }
}
```

```
class Manager extends Employee {
    public Manager(String name, String address, double salary) {
        super(name, address, salary, "Manager");
    }
    @Override
    public double calculateBonus() {
        return 0.1 * salary;
    }
}
class Developer extends Employee {
    public Developer(String name, String address, double salary) {
        super(name, address, salary, "Developer");
    }
    @Override
    public double calculateBonus() {
        return 0.05 * salary;
    }
}

class Programmer extends Employee {
    public Programmer(String name, String address, double salary)
{
        super(name, address, salary, "Programmer");
    }
    @Override
    public double calculateBonus() {
        return 0.03 * salary;
    }
}


public class assignment3_3 {
    public static void main(String[] args) {
        Manager manager = new Manager("ravi", "123 Main St",
80000.0);
        Developer developer = new Developer("Raj", "456 Oak St",
60000.0);
        Programmer programmer = new Programmer("Roshan", "789 Pine
St", 50000.0);
        System.out.println("Manager Bonus: $" +
manager.calculateBonus());
        System.out.println("Developer Performance Report: " +
developer.generatePerformanceReport());
        programmer.manageProject("Software Project X");
    }
}
```

- Output:

```
Manager Bonus: $8000.0
Good
Developer Performance Report: Performance report for Raj
Programmer Roshan is managing the project: Software Project X
```

ASSIGNMENT-3

Q4. Write a Java program to create a class called Shape with methods called getPerimeter() and getArea().Create a subclass called Circle that overrides the getPerimeter() and getArea() methods tocalculate the area and perimeter of a circle.

- Code:

```
class Shape{

    public void getPerimeter(){

        System.out.println("Perimeter is the length of the border
of the shape.");

    }

    public void getArea(){

        System.out.println("Area is the space occupy by an
object.");

    }

}

class Circle extends Shape{

    public void getPerimeter(int r) {

        double peri = 2 * 3.14 * r;

        System.out.println("Perimeter is: " + peri);

    }

    public void getArea(int r) {

        double area = 3.14 * r * r;

        System.out.println("Area is: " + area);

    }

}

public class assignment3_4 {

    public static void main(String[] args) {

        Circle c1 = new Circle();
        c1.getArea(14);

        c1.getPerimeter(7);

    }

}
```

- **Output:**

```
Area is: 615.44

Perimeter is: 43.96
```

Q5. Write a Java program to create a vehicle class hierarchy. The base class should be Vehicle,with subclasses Truck, Car and Motorcycle.Each subclass should have properties such as make, model,year, and fuel type. Implement methods for calculating fuel efficiency, distance traveled, and maximum speed.

- Code:

```
class Vehicle {
    String make;
    String model;
    int year;
    String fuelType;
    public Vehicle(String make, String model, int year, String
fuelType) {
        this.make = make;
        this.model = model;
        this.year = year;
        this.fuelType = fuelType;
    }
    public double calculateFuelEfficiency() {
        return 0.0;
    }
    public double calculateDistanceTraveled(double fuelEfficiency,
double fuelConsumed) {

        return fuelEfficiency * fuelConsumed;
    }
    public int getMaxSpeed() {
        return 0;
    }
```

```
}
class Truck extends Vehicle {
       @Override
    public double calculateFuelEfficiency() {
        double fuel_eff = 3.2;
        return fuel_eff;
    }
    @Override
    public int getMaxSpeed() {
        int max_speed = 120;
        return max_speed;
    }
    public Truck(String make, String model, int year, String fuelType) {
        super(make, model, year, fuelType);
    }
}
class Car extends Vehicle {
    @Override
    public double calculateFuelEfficiency() {
        double fuel_eff = 4.1;
        return fuel_eff;
    }
    @Override
    public int getMaxSpeed() {
        int max_speed = 140;
        return max_speed;
    }
    public Car(String make, String model, int year, String fuelType) {
        super(make, model, year, fuelType);
    }
}

class Motorcycle extends Vehicle {
    @Override
    public double calculateFuelEfficiency() {
        double fuel_eff = 5;
        return fuel_eff;
    }
    @Override
    public int getMaxSpeed() {
        int max_speed = 160;
        return max_speed;
    }
    public Motorcycle(String make, String model, int year, String fuelType) {
        super(make, model, year, fuelType);
    }
}
public class assignment3_5 {
    public static void main(String[] args) {
        Truck truck = new Truck("Ford", "F150", 2022, "Gasoline");
        Car car = new Car("Toyota", "Camry", 2023, "Hybrid");
        Motorcycle motorcycle = new Motorcycle("Harley-Davidson", "Sportster", 2021, "Gasoline");
        System.out.println("Truck Fuel Efficiency: " +
truck.calculateFuelEfficiency());

        System.out.println("Truck Fuel Efficiency: " +
truck.calculateDistanceTraveled(truck.calculateFuelEfficiency(),
46));

        System.out.println("Truck Fuel Efficiency: " +
truck.getMaxSpeed());

        System.out.println("Car Distance Traveled: " +
car.calculateFuelEfficiency());

        System.out.println("Car Distance Traveled: " +
car.calculateDistanceTraveled(25.0, 10.0));

        System.out.println("Car Distance Traveled: " +
car.getMaxSpeed());

        System.out.println("Motorcycle Max Speed: " +
motorcycle.getMaxSpeed());

        System.out.println("Motorcycle Max Speed: " +
motorcycle.calculateFuelEfficiency());

        System.out.println("Motorcycle Max Speed: " +
motorcycle.calculateDistanceTraveled(motorcycle.calculateFuelEffic
iency(), 50));

    }

}
```

- **Output:**

```
Truck Fuel Efficiency: 3.2

Truck Fuel Efficiency: 147.20000000000002

Truck Fuel Efficiency: 120

Car Distance Traveled: 4.1

Car Distance Traveled: 250.0

Car Distance Traveled: 140

Motorcycle Max Speed: 160

Motorcycle Max Speed: 5.0

Motorcycle Max Speed: 250.0
```