Java Lab 3

Piyush Raj

1. **WAP to create class Num with only one private instance variable as a double primitive type, include the following methods isZero(), isPositive(), isNegative( ), isOdd( ), isEven( ), isPrime(), isAmstrong() in this class and all above methods should return boolean primitive type like for isPositive() should return “Positive = true”.**

Code:-

import java.util.Scanner;

public class Num {

    private double number;

    public Num(double number) {

        this.number = number;

    }

    public boolean isZero() {

        return number == 0;

    }

    public boolean isPositive() {

        return number > 0;

    }

    public boolean isNegative() {

        return number < 0;

    }

    public boolean isOdd() {

        return number % 2 != 0;

    }

    public boolean isEven() {

        return number % 2 == 0;

    }

    public boolean isPrime() {

        if (number <= 1) return false;

        for (int i = 2; i <= Math.sqrt(number); i++) {

            if (number % i == 0) return false;

        }

        return true;

    }

    public boolean isArmstrong() {

        int temp = (int) number;

        int digits = 0;

        int result = 0;

        int originalNumber = temp;

        while (originalNumber != 0) {

            originalNumber /= 10;

            ++digits;

        }

        originalNumber = temp;

        while (originalNumber != 0) {

            int remainder = originalNumber % 10;

            result += Math.pow(remainder, digits);

            originalNumber /= 10;

        }

        return result == temp;

    }

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter a number: ");

        double userInput = scanner.nextDouble();

        scanner.close();

        Num num = new Num(userInput);

        System.out.println("isZero() : " + num.isZero());

        System.out.println("isPositive() : " + num.isPositive());

        System.out.println("isNegative() : " + num.isNegative());

        System.out.println("isOdd() : " + num.isOdd());

        System.out.println("isEven() : " + num.isEven());

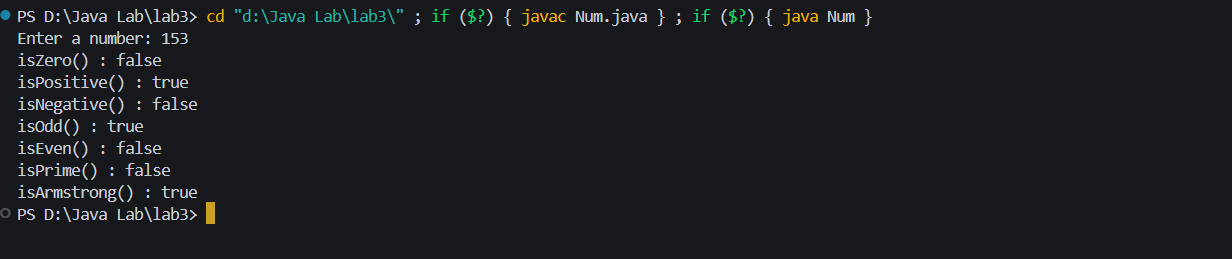
        System.out.println("isPrime() : " + num.isPrime());

        System.out.println("isArmstrong() : " + num.isArmstrong());

    }

}

OUTPUT:-



1. **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 {

    protected String make;

    protected String model;

    protected int year;

    protected String fuelType;

    public Vehicle(String make, String model, int year, String fuelType) {

        this.make = make;

        this.model = model;

        this.year = year;

        this.fuelType = fuelType;

    }

}

class Truck extends Vehicle {

    public Truck(String make, String model, int year, String fuelType) {

        super(make, model, year, fuelType);

    }

    public void calculateFuelEfficiency(double distanceDriven, double fuelUsed) {

         System.out.println( "Maximum Fuel Efficiency is: "+distanceDriven / fuelUsed);

    }

    public void Distance(int totalDistance) {

        System.out.println("Total Distanced Covered is: "+totalDistance);

    }

    public void MaxSpeed(int maxSpeed) {

        System.out.println("Maximum Speed is: "+maxSpeed);

   }

}

class Car extends Vehicle {

    public Car(String make, String model, int year, String fuelType) {

        super(make, model, year, fuelType);

    }

    public void calculateFuelEfficiency(double distanceDriven, double fuelUsed) {

        System.out.println( "Maximum Fuel Efficiency is: "+distanceDriven / fuelUsed);

   }

   public void Distance(int totalDistance) {

    System.out.println("Total Distanced Covered is: "+totalDistance);

}

    public void MaxSpeed(int maxSpeed) {

        System.out.println("Maximum Speed is: "+maxSpeed);

   }

}

class Motorcycle extends Vehicle {

    public Motorcycle(String make, String model, int year, String fuelType) {

        super(make, model, year, fuelType);

    }

    public void calculateFuelEfficiency(double distanceDriven, double fuelUsed) {

        System.out.println( "Maximum Fuel Efficiency is: "+distanceDriven / fuelUsed);

   }

    public void Distance(int totalDistance) {

        System.out.println("Total Distanced Covered is: "+totalDistance);

    }

    public void MaxSpeed(int maxSpeed) {

         System.out.println("Maximum Speed is: "+maxSpeed);

    }

}

public class Vehicles {

    public static void main(String[] args) {

        Truck truck = new Truck("Ford", "F150", 2022, "Gasoline");

        Car car = new Car("Toyota", "Corolla", 2023, "Gasoline");

        Motorcycle motorcycle = new Motorcycle("Harley-Davidson", "Sportster", 2021, "Gasoline");

        System.out.println("Truck: " + truck.make + " " + truck.model + " " + truck.year + " " + truck.fuelType );

         truck.Distance(1220);

         truck.MaxSpeed(200);

         truck.calculateFuelEfficiency(1220,140.0);

        System.out.println("Car: " + car.make + " " + car.model + " " + car.year + " " + car.fuelType);

        car.Distance(1600);

        car.MaxSpeed(300);

        car.calculateFuelEfficiency(1600,300.0);

        System.out.println("Motorcycle: " + motorcycle.make + " " + motorcycle.model + " " + motorcycle.year + " " + motorcycle.fuelType);

        motorcycle.Distance(2000);

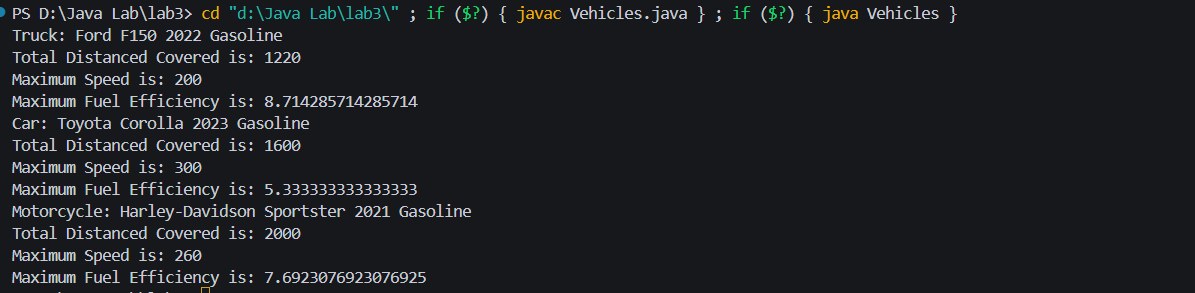
        motorcycle.MaxSpeed(260);

        motorcycle.calculateFuelEfficiency(2000,260.0);

    }

}

Output:-



1. **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 {

    protected String name;

    protected String address;

    protected double salary;

    protected String jobTitle;

    public Employee(String name, String address, double salary, String jobTitle) {

        this.name = name;

        this.address = address;

        this.salary = salary;

        this.jobTitle = jobTitle;

    }

    public void generatePerformanceReport() {

        System.out.println("Performance report for " + name + ":");

    }

}

class Manager extends Employee {

    private int numOfProjects;

    public Manager(String name, String address, double salary, String jobTitle, int numOfProjects) {

        super(name, address, salary, jobTitle);

        this.numOfProjects = numOfProjects;

    }

    public void manageProjects() {

        System.out.println(name + " is managing " + numOfProjects + " projects.");

    }

    public double calculateBonus() {

        return salary \* 0.1;

    }

}

class Developer extends Employee {

    private double percentCompletion;

    public Developer(String name, String address, double salary, String jobTitle, double percentCompletion) {

        super(name, address, salary, jobTitle);

        this.percentCompletion = percentCompletion;

    }

    public void performanceCompletion() {

        System.out.println(name + "'s performance completion: " + percentCompletion \* 100 + "%");

    }

    public double calculateBonus() {

        return salary \* 0.05 \* percentCompletion;

    }

}

class Programmer extends Employee {

    private int numOfHoursWorked;

    public Programmer(String name, String address, double salary, String jobTitle, int numOfHoursWorked) {

        super(name, address, salary, jobTitle);

        this.numOfHoursWorked = numOfHoursWorked;

    }

    public void hoursWorked() {

        System.out.println(name + " worked for " + numOfHoursWorked + " hours.");

    }

    public double calculateBonus() {

        return numOfHoursWorked \* 10;

    }

}

public class Company {

    public static void main(String[] args) {

        Manager manager = new Manager("Rahul", "Kolkata", 80000, "Manager", 5);

        Developer developer = new Developer("Rohan", "Mumbai", 70000, "Developer", 0.8);

        Programmer programmer = new Programmer("Raman", "Delhi", 60000, "Programmer", 40);

        manager.generatePerformanceReport();

        manager.manageProjects();

        System.out.println("Manager's bonus: $" + manager.calculateBonus());

        developer.generatePerformanceReport();

        developer.performanceCompletion();

        System.out.println("Developer's bonus: $" + developer.calculateBonus());

        programmer.generatePerformanceReport();

        programmer.hoursWorked();

        System.out.println("Programmer's bonus: $" + programmer.calculateBonus());

    }

}

Output:-

