People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
University M'hamed Bouguerra - Boumerdes
Institute of Electrical and Electronics Engineering
Electronics Department

Option: Computer engineering

LAB REPORT n°6

January 4, 2023

Title

EE423: Advanced Programming/ Working with Inheritance, Interfaces and Packages (Part 2)

Authored by :
Agli Wafa
Zidane Aymen

Instractor :
Dr.A Zitouni

Session : 2022/2023

Contents

Introduction

Within this lab, we will learn to define interfaces in Java and implement them.

Tools and Software:

1. A PC with ECLIPSE IDE V8.
2. Online LaTeX Editor for writing the report.

1 Assignment

```
    interface Vehicle {
        public static double MAX_TANK = 50;
        public abstract void move(double distance);
        public abstract double addFuel(double amount);
        public abstract void print();
        public default void honk(){ System.out.println("Ton Ton"); }
    }

```

1.1 *Answer 01:*

```
  public class Car {
  double fuel; // the amount of fuel left in the tank
  double totalDistance; // the total distance covered by the car
  double yield; // the number of kilometers a car can cover
}


```

1.2 *Answer 02:*

```
   public class Car {
      double fuel; // the amount of fuel left in the tank
      double totalDistance; // the total distance covered by the car
      double yield; // the number of kilometers a car can cover
       Car(double yeild){
            fuel = totalDistance = 0;
            this.yield = yeild;
        }
 }


```

1.3 *Answer 03:*

```
public class Car implements Vehicle {
  double fuel; // the amount of fuel left in the tank
  double totalDistance; // the total distance covered by the car
  double yield; // the number of kilometers a car can cover
```

```
  Car(double yeild){
    fuel = totalDistance = 0;
    this.yield = yeild;
    }

     @Override
    public void move(double distance) {
        double required_fuel = (distance/yield);
        if(fuel < required_fuel)
            System.out.println("No enough fuel to move that distance");
        else {
            totalDistance+=distance;
            fuel-=required_fuel;
            System.out.println("The car moved "+distance+" KM and consumed " +
    required_fuel +"L of fuel");
        }

    }

     @Override
    public double addFuel(double amount) {
        fuel+= amount;
        fuel = (MAX_TANK < fuel) ? MAX_TANK : fuel;
        System.out.println("Amount of fuel: "+ fuel +" L ");
        return  fuel;
    }

    @Override
    public void print() {
        System.out.println("Total distance: "+totalDistance+" Remaining fuel: "+fuel+"
    Yield: "+yield);
    }
  }

```

1.4 *Answer 04:*

```
class TestCar {
    public static void main(String[] args) {
        Vehicle car = new Car(10);
        car.move(20);
        car.print();
        car.addFuel(100);
        car.honk();
        car.move(300);
        car.print();
        }
}

```

```
No enough fuel to move that distance
Total distance: 0.0 Remaining fuel: 0.0 Yield: 10.0
Amount of fuel: 50.0 L
Ton Ton
The car moved 300.0 KM and consumed 30.0L of fuel
Total distance: 300.0 Remaining fuel: 20.0 Yield: 10.0
```

Figure 1

1.5 *Answer 05:*

```java
interface VehicleDiesel extends Vehicle{
    double co2Emission();
}

```

1.6 *Answer 06:*

```java
interface VehicleDiesel extends Vehicle{
    double CO2_EMISSION_DIESEL = 0.25;
    double co2Emission();
}

```

1.7 *Answer 07:*

```java
public class CarDiesel extends Car implements VehicleDiesel{

    CarDiesel(double yeild){
        super(yeild);
    }


    @Override
    public void print() {
        super.print();
        System.out.println("Total co2-emission: " + co2Emission() + "M3");
    }


    @Override
    public double co2Emission() {
        return totalDistance * CO2_EMISSION_DIESEL;
    }


}

```

1.8 *Answer 08:*

```
interface VehicleDiesel extends Vehicle{
    double CO2_EMISSION_DIESEL = 0.25;
    double co2Emission();
    @Override
    default void honk() {
        System.out.println("Diesel Ton Ton");
    }
}

```

1.9 *Answer 09:*

```
interface Vehicle {
    public static double MAX_TANK = 50;
    public abstract void move(double distance);
    public abstract double addFuel(double amount);
    public abstract void print();
    public default void honk(){System.out.println("Ton Ton");}
    public default void start(){System.out.println("vehicle stared");}
    public default void stop(){System.out.println("vehicle stopped");}

}
```

1.10 *Answer 10:*

```
class TestCarDiesel {
    public static void main(String[] args) {
        VehicleDiesel carDiesel = new CarDiesel(20);
        carDiesel.start();
        carDiesel.move(100);
        carDiesel.addFuel(100);
        carDiesel.move(100);
        carDiesel.honk();
        carDiesel.print();
        carDiesel.stop();
        System.out.println("--------------");
        Vehicle car = new Car(10);
        car.start();
        car.addFuel(100);
        car.move(300);
        car.print();
        car.stop();
        }
        }
```

```
vehicle stared
No enough fuel to move that distance
Amount of fuel: 50.0 L
The car moved 100.0 KM and consumed 5.0L of fuel
Diesel Ton Ton
Total distance: 100.0 Remaining fuel: 45.0 Yield: 20.0
Total co2-emission: 25.0M3
vehicle stopped
--------------
vehicle stared
Amount of fuel: 50.0 L
The car moved 300.0 KM and consumed 30.0L of fuel
Total distance: 300.0 Remaining fuel: 20.0 Yield: 10.0
vehicle stopped
```

Figure 2

2 Conclusion

1. Although Java does not allow multiple inheritance, it does allow classes to implement any number of interfaces.
2. An interface is an abstract data type that defines a list of abstract public methods that any class implementing the interface must provide.
3. An interface can also include a list of constant variables and default methods.