2 nd Semester

Module	- Object Oriented Programming using Java
Module Lecturer	- Mr. Mohamad Shafraz
Submission Date	- 22.05 . 2023

Student Name	- W.M. Ayodhya Weerabahu (28914)
Batch	- 22.2
Degree Program	- BSc in Management Information Systems

1.

a)

- Access Modifiers:
    - Public: Public members are accessible from anywhere, both within the class and outside the class.
    - Private: Private members are accessible only within the class they are declared in.
- Data Types:
    - int: Represents integer values, such as 1, -5, or 1000.
    - double: Represents floating-point values with double precision, such as 3.14 or -0.5.
    - boolean: Represents a boolean value, either true or false.
    - char: Represents a single character, such as 'a' or 'X'.
    - String: Represents a sequence of characters, such as "Hello" or "Java".
    - Arrays: Represents a collection of elements of the same data type

b)

The purpose of getters and setters, also known as accessor and mutator methods, is to provide controlled access to the private fields (variables) of a class. They serve several important purposes:

c)

```
Class Employee{

Private int employeeID;

Private String employeeName;

Float employeesalary;

}
```

d)

```
Public void Employee(int employeeID, String employeeName, Float employeesalary){

this. employeeID= employeeID;
this. employeeName= employeeName;
this. Employeesalary= employeesalary;
```

```
        }
```

e)

```
Public void Employee(int employeeID, String employeeName, Float employeesalary){

this. employeeID= employeeID;
this. employeeName= employeeName;
this. Employeesalary= employeesalary;
```

f)

```
  public String getName() {
    return name;
  }


  public void setName(String employeeName) {
    this.name = employeeName;
  }

  public int getId() {
    return Id;
  }
  public void setId(int employeeID) {
    this.age = employeeID;
  }
  public float getSalary() {
    return salary;
  }
  public void setId(float employeesalary) {
        this.age = employeesalary;
    }



}
```

2.

```
Public class Car{
public class Car {
    private int year;
    private double mileage;
    private String make;
    private String model;
    private String color;

    public Car(int year, double mileage, String make, String model, String color) {
        this.year = year;
        this.mileage = mileage;
        this.make = make;
        this.model = model;
        this.color = color;
    }

    public int getYear() {
        return year;
    }

    public void setYear(int year) {
        this.year = year;
    }

    public double getMileage() {
        return mileage;
    }

    public void setMileage(double mileage) {
        this.mileage = mileage;
    }

    public String getMake() {
        return make;
    }

    public void setMake(String make) {
        this.make = make;
    }

    public String getModel() {
        return model;
    }

    public void setModel(String model) {
        this.model = model;
    }

    public String getColor() {
        return color;
    }


    public void setColor(String color) {

        this.color = color;

    }

}

public void drive(double distance) {

        mileage += distance;

}

     public void displayCarInfo() {

        System.out.println("Car Information:");

        System.out.println("Year: " + year);

        System.out.println("Mileage: " + mileage);

        System.out.println("Make: " + make);

        System.out.println("Model: " + model);

        System.out.println("Color: " + color);



}
```

- In Java, inheritance is represented using the extends keyword

```
Public class Shpe
{

Public void calculateArea()
{

}

}
```

```
Public class Rectangle extends Shape
{
private double radius;

    public Circle(double radius) {
        this.radius = radius;
    }

public double calculateArea()
{
return Math.PI * radius * radius;
```

```
Public class Circle extends Shape
{

private double length;
    private double width;

    public Rectangle(double length, 
double width) {
        this.length = length;
        this.width = width;
    }

    public double calculateArea() {
        return length * width;
    }
}
}
```

```
public class Main {
    public static void main(String[] args) {
        Circle circle = new Circle(5.0);
        System.out.println("Circle Area: " + circle.calculateArea());
        System.out.println("Circle Perimeter: " + circle.calculatePerimeter());

        Rectangle rectangle = new Rectangle(4.0, 6.0);
        System.out.println("Rectangle Area: " + rectangle.calculateArea());
        System.out.println("Rectangle Perimeter: " + rectangle.calculatePerimeter());
    }
}
```