CS112: Programming II

Dr. Sameer M. Alrehaili

January 23, 2022

srehaili@taibahu.edu.sa
college of computer science and engineering ,yanbu, Taibah University

Lab01

Laboratory Objectives:

- To describe objects and classes, and use classes to model objects.
- To use UML notation to draw classes and objects design.
- To demonstrate how to define classes and create objects.
- To create objects using default constructors.
- To access objects via object reference variables.
- To define a reference variable using a reference type.
- To access an objects data and methods using the object member access operator (.).

Note about visibility:
+ public
# protected
- private
~ package (default)
/ derived

Exercises

Part1: Circle class

- Design a class called **Circle** that represents relevant information about a circle that has radius as a field. The class should have a parameterised constructor which is explicitly defined in the class to accept radius value. The class should also include getArea, and getPermieter and methods to set and get thier data fields. Draw the class diagram using UML notation and then write the class definition using Java language?

Circle
- radius : double
+ Circle(newRadius : double) + getArea() : double + getPermieter() : double + setRadius(newRadius : double) : void + getRadius() : double

Listing 1: Defining Circle class

```java
public class Circle{
  private double radius;


  public Circle(double newRadius){
    this.radius=newRadius;
  }
  public double getArea(){
    return radius * radius * Math.PI;
  }
  public double getPermieter(){
    return 2 * radius * Math.PI;
  }
  public void setRadius(double newRadius){
    this.radius=newRadius;
  }
  public double getRadius(){
    return this.radius;
  }
}
```

Part2: Object creation

Now, Use UML to represent objects of the circle class, and then write Java code to implement the creation of these objects.

circle1: Circle
radius = 1.0

circle2: Circle
radius = 25

<u>**circle3: Circle**</u>
radius = 125

Listing 2: create objects from Circle

```java
public class test{
  public static void main(String[] args) {
    Circle circle1 = new Circle(1);
    System.out.println(circle1.getArea());

    Circle circle2 = new Circle(25);
    System.out.println(circle2.getArea());

    Circle circle3 = new Circle(125);
    System.out.println(circle3.getArea());
  }
}
```

Part3: Static variables

Add a static variable named **numberOfObjects** to the class Circle for count number of objects created from Circle. Do not forget to add a method to get the new data field value.

Circle
- radius : double - numberOfObjects : int
+ Circle(newRadius : double) + getArea() : double + getPermieter() : double + setRadius(newRadius : double) : void + getRadius() : double + getNumberOfObjects() : int

Listing 3: Defining Circle class with static data field

```java
public class Circle{
  private double radius;
  private static int numberOfObjects=0;

  public Circle(){
    this.radius=1;
    numberOfObjects++;
  }

  public Circle(double newRadius){
    this.radius=newRadius;
    numberOfObjects++;
  }
  public double getArea(){
    return radius * radius * Math.PI;
  }
  public double getPermieter(){
    return 2 * radius * Math.PI;
  }
  public void setRadius(double newRadius){
    this.radius=newRadius;
  }
  public double getRadius(){
    return this.radius;
  }
  public static int getNumberOfObjects(){
    return numberOfObjects;
  }
}
```

Listing 4: Defining Circle class with static data field

```java
public class test{
  public static void main(String[] args) {
    Circle circle1 = new Circle(1);
    System.out.println(circle1.getArea());
    System.out.println("The number of objects is : " +
        circle1.getNumberOfObjects());

    Circle circle2 = new Circle(25);
    System.out.println(circle2.getArea());
    System.out.println("The number of objects is : " +
        circle1.getNumberOfObjects());

    Circle circle3 = new Circle(125);
    System.out.println(circle3.getArea());

    System.out.println("The number of objects is : " +
        circle1.getNumberOfObjects());
  }
}
```

Part4: Car class

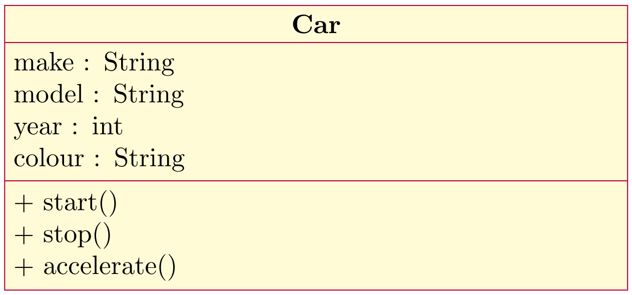
Design a class called **Car** to represent cars. The Car class has the following fields:

– make of the car represented as string.
– model of the car represented as string.
– year which represented as integer.
– colour which represented as string.

The class should include a constructor and the following methods:

– **start()**:to print some text relevant to the start such as ”start the engine”.
– **stop()**:to print ”stop the engine”.
– **accelerate()**:to print ”accelerate”.

Draw UML notation for the class then write it in Java language.



Listing 5: Declaration of an array

```
public class Car{
  private String make;
  private String model;
  private int year;
  private String colour;

  public car(){
  }

  public void start(){
    System.out.println("Start the engine!");
  }

  public void stop(){
    System.out.println("Stop the engine!");
  }
  public void accelerate(){
    System.out.println("Accelerate!");
  }
}
```