CS102A Introduction to Computer Programming

Fall 2020

Lab 7

Credit

The source code and document description are designed by ZHU Yueming.

Objectives

1. Learn how to define a Java class.
2. Learn how to use instance variables.
3. Learn how to define and use instance methods.
4. Learn how to use `get` and `set` methods.
5. Learn how to use the `ArrayList` class.

1 Prework

1.1 Step 1: How to define a circle on a 2D plane?

A circle is defined using three attributes: its `radius` and coordinates `x`, `y`. Let us define a class named `Circle` as follows:

```
public class Circle {
    private double radius;
    private double x;
    private double y;
}
```

1.2 Step 2: Define methods for computing and printing a circle's information.

Define three methods for computing the area and perimeter of a `Circle`, as well as printing its position on a 2D plane.

```
public class Circle {
    private double radius;
    private double x;
    private double y;

    public double area() {
        return radius*radius*Math.PI;
    }

    public double perimeter () {
        return 2*Math.PI*radius;
    }

    public void position() {
        System.out.printf("Position of the circle is (%.1f,%.1f)\
           n",x,y);
    }
}
```

1.3 Step 3: How to use the `Circle` class?

Create another class named `CircleTest` in the same package, in which there is a `main` method. In the `main` method, we can create a `Circle` object as follows:

```
Circle c1 = new Circle();
```

To print the perimeter, area, and position of `c1`, we need to invoke the methods of `c1` as follows:

```
public class CircleTest {

```

```
    public static void main(String[] args) {
        Circle c1 = new Circle();
        System.out.printf("The area of c1 is %.2f\n", c1.area());
        System.out.printf("The perimeter of c1 is %.2f\n", c1.
           perimeter());
        c1.position();
    }
}
```

When we run the program, the result would be:

```
The area of c1 is 0.00
The perimeter of c1 is 0.00
Position of the circle is (0.0,0.0)
```

1.4 Step 4: How to `set` and `get` attribute values?

If we attempt to directly set or get the `radius` of a `Circle` object in `main`, it would lead to an error due to the attribute's `private` access modifier. In addition, the radius of a circle cannot be negative; how can we enforce the appropriate restrictions?

```
public static void main(String[] args) {
    Circle c1 = new Circle();
    System.out.printf("The area of c1 is %.2f\n", c1.area());
    System.out.printf("The perimeter of c1 is %.2f\n", c1.
       perimeter());
    c1.position();
    c1.radius = -1;
    System.out.println(c1.radius);
}
```

For getting or setting the class variables, we will define corresponding `public` methods in the `Circle` class. This will also allow us to check the validity of the input values when setting the `radius` attribute in the `setRadius` method.

```
public class Circle {
```

```
    private double radius;
    private double x;
    private double y;

    public double area() {
        return radius * radius * Math.PI;
    }

    public double perimeter () {
        return 2 * Math.PI * radius;
    }

    public void position() {
        System.out.printf("Position of the circle is (%.1f,%.1f)\
           n", x, y);
    }

    public double getRadius() {
        return radius;
    }

    public void setRadius(double radius) {
        if (radius > 0) {
            this.radius = radius;
        }
    }

    public double getX() {
        return x;
    }

    public void setX(double x) {
        this.x = x;
```

```
    }

    public double getY() {
        return y;
    }

    public void setY(double y) {
        this.y = y;
    }
}
```

After that, we can access and modify the attributes by the get and set methods, respectively:

```
public static void main(String[] args) {
    Circle c1 = new Circle();

    c1.setRadius(5);
    System.out.println(c1.getRadius());

    System.out.printf("The area of c1 is %.2f\n", c1.area());
    System.out.printf("The perimeter of c1 is %.2f\n", c1.
       perimeter());
    c1.position();
}
```

Sample input and output:

```
5.0
The area of c1 is 78.54
The perimeter of c1 is 31.42
Position of the circle is (0.0,0.0)
```

1.5 Step 5: How to manage multiple `Circle` objects?

We can manage multiple `Circle` objects using an array or an `ArrayList` instance. In the `main` method, create an `ArrayList` with type `Circle` to store many `Circle` objects. To do so, add the following code at the end of `main`:

```
ArrayList<Circle> circleList = new ArrayList<Circle>();
circleList.add(c1);
System.out.printf("Radius of %d circle is %.2f: \n", 1,
   circleList.get(0).getRadius());
```

Sample input and output:

```
5.0
The area of c1 is 78.54
The perimeter of c1 is 31.42
Position of the circle is (0.0,0.0)
Radius of 1 circle is 5.00:
```

1.6 Step 6: Add more circles to the `ArrayList`.

Add the following code at the end of `main`:

```
    for(int i = 1; i < 5; i++) {
        circleList.add(new Circle());
        circleList.get(i).setRadius(i);
        circleList.get(i).setX(Math.random() * 5);
        circleList.get(i).setY(Math.random() * 5);
    }

    System.out.println("---Begin to print the circle list---");
    for(int i = 0; i < 5; i++) {
        System.out.printf("The area of %d circle is %.2f\n", i+1,
            circleList.get(i).area());
        System.out.printf("The perimeter is %.2f\n", circleList.
           get(i).perimeter());
```

```
12      }
```

Sample input and output:

```
5.0
The area of c1 is 78.54
The perimeter of c1 is 31.42
Position of the circle is (0.0,0.0)
Radius of 1 circle is 5.00:
---Begin to print the circle list---
The area of 1 circle is 78.54
The perimeter is 31.42
The area of 2 circle is 3.14
The perimeter is 6.28
The area of 3 circle is 12.57
The perimeter is 12.57
The area of 4 circle is 28.27
The perimeter is 18.85
The area of 5 circle is 50.27
The perimeter is 25.13
```

2 Exercises

2.1 Exercise 1

Define a `User` class as follows:

- Add the following data fields:
  - `String name`;
  - `String password`;
  - `double money`;
- Implement a public method named `introduce()` that prints the user's name and account balance.

- Implement a public method named `expense(double value)` that withdraws the money from the user's account.
- Implement a public method named `income(double value)` that deposits the money to the user's account.
- Implement getter and setter methods for each `private` field of the `User` class.

In the same package, let us create a class named `ClientTest`, which has the following `main` method:

```
public static void main(String[] args) {
    User user = new User();
    user.setName("Lucy");
    user.setPassword("123456");
    user.setMoney(1000);
    user.introduce();
    user.expense(2000);
    user.expense(500);
    user.income(1000);
    user.introduce();
}
```

Sample input and output:

```
Your name is Lucy and you have 1000.00 dollars
Insufficient funds!
You have withdrawn 500.00 dollars and the remaining amount is
500.00 dollars
The remaining amount is 1500.00 dollars
Your name is Lucy and you have 1500.00 dollars
```

2.2 Exercise 2

Design a `Food` class as follows:

- Add the following private data fields:
  - `String name`;
  - `String type`;
  - `int size`;
  - `double price`;
- Implement a public method named `showInformation()` to print all the information of this `Food` object.
- Implement getter and setter methods for each private field of `Food`.

In the `ClientTest` class, create the following four `Food` objects:

Object Name	`name`	`type`	`size`	`price`
`pizza1`	`pizza`	`Seafood`	`11`	`120`
`pizza2`	`pizza`	`Beef`	`9`	`100`
`FriedRice`	`fried rice`	`Seafood`	`5`	`40`
`Noodles`	`noodles`	`Beef`	`6`	`35`

Create an `ArrayList<Food>` object to add the above four `Food` objects, and then print their information by iterating over the `ArrayList<Food>` object you created.

Sample input and output:

```
Seafood pizza: (11 Inches) 120.00 $
Beef pizza: (9 Inches) 100.00 $
Seafood fried rice: (5 Inches) 40.00 $
Beef noodle: (6 Inches) 35.00 $
```