Tutorial of Class and Object (Basic)

> Based on the tutorial of "2020S-Java-A" designed by teaching group in SUSTech
>
> Designed by ZHU Yueming
>
> Improved by WANG Wei
>
> Modified (mainly change to markdown file) by ZHU Yueming in 2021. March. 29th

Experimental Objective

- Learn how to define a Java class and create its object
- Learn how to define and use instance variables
- Learn how to define and use instance methods
- Learn how to use get and set methods
- Learn how to use `ArrayList` and make the object as its element.

Before Exercise

Attribute and Method

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

A circle has three attributes including the **radius**, the **x** *coordinate* and the **y** *coordinate*.

We can define a class named `Circle`, in which there are three private attributes.

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

Step 2: Define the methods of a circle.

Define three public methods for computing the area, perimeter and print position of the circle.

```
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 cricle is (%.1f, %.1f)\n",x,y);
```

```
    }
}
```

Step 3: How to use the class Circle?

Create another class named `CircleTest` in the same package, in which there is a main method to be used.

In the main method, we can create an object of `Circle` by using the statement as follows:

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

After that, we want to know the perimeter, area and position about the `c1`, so we need to invoke the method of `c1`.

```
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 as follows:

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

Getter and Setter

Step 4: Set and get the values of the attributes

If we set or get the radius of a circle object in main method directly, it would lead to an error because of its private privilege.

In addition, the radius of a circle should not contain a negative number, how can we set the restriction?

```
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);
    }
```

We can define several public methods in class Circle for getting or setting the class variables, and we can check the validity of input value in the set 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 cricle 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 the attributes by the get and set methods.

```
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 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)
```

ArrayList

Step 5: How to manage multiple circle objects ?

We can use an array or an `ArrayList` to manage them.

In the main method, create an arrayList with a Circle type, to store many objects of Circle. Add the following code at the end of main method.

```
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 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:
```

Step 6: Add more circles in the ArrayList

Add the following code at the end of main method.

```
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());
}
```

Sample 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 25.1
```

Exercise

Exercise 1 : User

Declare a class named **User**. The class contains:

- **Private** data fields:
  - String **account;**
  - String **password**;
  - double **money**;
- Implement a public method named **introduce()** to print the user account and his account balance.
- Implement a public method **expense(double value,Scanner in)**. It withdraws the money from the user account if the password is correct.
- Implement a public method **income(double value).** It deposits the money to the user account.
- Implement the **getter** and **setter** methods for each private field of the class User.

In the same package, we create a class named `UserTest`, which has a main method.

```
User user = new User();
Scanner in = new Scanner(System.in);
user.setUser("Lucy");
user.setPassword("123456");
user.setMoney(1000);
user.introduce();
user.expense(2000, in);
user.expense(500, in);
user.income(1000);
user.introduce();
in.close();
```

Sample Output:

```
My name is Lucy and I have 1000.00 dollar
no sufficient funds
You have expense 500.00 dollar and the remained amount is 500.00
The remained amount is 1500.00
My name is Lucy and I have 1500.00 dollar
```

Exercise 2 : Food

Design a class named **Food**. The class contains:

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

In `FoodTest` class, create four objects of Food as follows:

Object Name	id	name	type	size	price
pizza1	1	pizza	Seafood	11	12
pizza2	2	pizza	Beef	9	10
Fried rice	3	fried rice	Seafood	5	12
Noodles	4	noodles	Beef	6	14

Create an `ArrayList<Food>` to add those four Food objects, and then show the information of them as follows by iterating the `ArrayList<Food>` we created.

```
Seafood pizza: (11 Inches) 12.00 $
Beef pizza: (9 Inches) 10.00 $
Seafood fried rice: (5 Inches) 12.00 $
Beef noodle: (6 Inches) 14.00 $
```

Exercise 3: Combine Food and User

Design a class named `SoftOpening`. The class contains no data fields but:

- Implement a public static method named `generateMenu()` to generate 4 object of Food and add them to the `ArrayList<Food>`.
- Implement a public static method named `getMenu(ArrayList<Food> foodList)` to print the items in the `ArrayList<Food>` as designed.
- Implement a public static method named to `generateUser(Scanner in)` to generated a user whose account and money is get by using the Scanner object 'in'.
- Implement a public static method named `UserConsume(ArrayList<Food> foodList, User user, Scanner in)` to invoke the `getMenu()`, ask user to select the foods in the Menu, count the cost and invoke the expense of the user.
- Invoke the method `introduce()` of the User object to show his/hers balance.

Statements in main method:

```
Scanner in = new Scanner(System.in);
ArrayList<Food> foodList = genarateMenu();  //generate a Menu
User user = genarateUser(in);               //generate a user
user.introduce();                           //show the account of the user
userConsume(foodList,user,in);              //user consume
user.introduce();                           //show the account of the user
in.close();
```

Sample Output:

```
Generate a user,please input name:Bob
balance($):2000
Bob's account has a balance of 2000.00 dollar
--------------------welcome,this is Start of the Menu--------------------
[id] 1  [type] Seafood    [name] pizza      [size] 11 (Inches) 12.00 $
[id] 2  [type] Beef       [name] pizza      [size]  9 (Inches) 10.00 $
[id] 3  [type] Seafood    [name] fried rice [size]  5 (Inches) 12.00 $
[id] 4  [type] Beef       [name] noodles    [size]  6 (Inches) 14.00 $
--------------------welcome,this is  End  of the Menu--------------------
please input the foodID and the number you want,to exit input 0 as foodID
food id(input 0 to end select):2
number of this food:10
food id(input 0 to end select):4
number of this food:1
food id(input 0 to end select):0
select end
Plan to expense 114.00 dollar
Please input your password:
123456
Expense 114.00 dollar and balance 1886.00 dollar
Bob's account has a balance of 1886.00 dollar
```