OOP

Practical 7, Week 7

Submission

1. Your submission should contain two files. One of these files is **PDF** document with screenshots of the implementation (Java code) and testing only. Another file is **ZIP** file with the Java project.
2. You must save the files with name

   {YourStudentNumber}-Practical7.pdf;

   {YourStudentNumber}-Practical7.zip;

   For example: 202107081314-Practical7.pdf, 202107081314-Practical7.zip
3. You must upload from the student website: student.zy.cdut.edu.cn

Marking scheme

You will gain up to 5 marks for the completion of the exercise.

The markers will use the following marking scheme for each exercise.

Rubric	marks
No attempt has been made to answer the question. No implementation at all, or completely inappropriate considerations	0
Some attempt has been made to design the algorithm, and some considerations shown. No effort to implement a working solution and test it.	1
Incomplete design/programming, but significant effort has gone into it. Some consideration and implementation of the result, but very limited, some of the rules have not been properly implemented, no testing.	2
Mostly complete design but the implementation does not match the design or does not follow a correct standard. The program works but does not use loops and string methods properly.	3
Complete design with fair to good implementation, and some testing shown.	4
Excellent design and implementation of the whole problem, including testing and implementation.	5

OOP

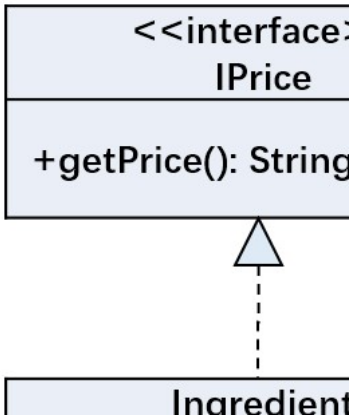
Week 7, Assessed exercises

Exercises 1

Given the Ingredients Interface class as follows:

```
public interface IPrice {
    public double getPrice();
}
```

Given a UML diagram as follows, write the Java code to implement the class **Ingredient** that implements the interface **IPrice**. An ingredient has a **name** and a **price** and a **method** to get the name of the ingredient and the price. The constructor of the class receives the **name** and the **price** of the ingredient as parameters.

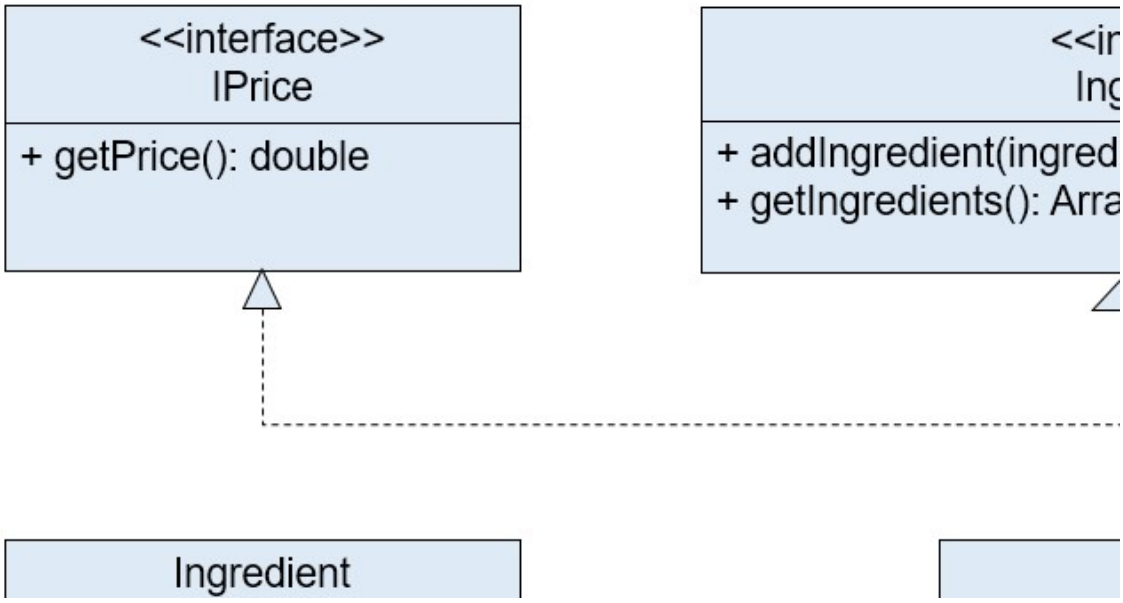


Exercises 2

Given the Ingredients Interface class as follows:

```
public interface Ingredients {
    public void addIngredient(Ingredient ingredient);
   public ArrayList<Ingredient> getIngredients();
}
```

Implement the class Meal. A **Meal** has a name and a **list** of ingredients and a price. The class implements the Ingredients and IPrice interfaces. The price of the meal is calculated by adding the prices of all the ingredients. The **Ingredient** class is the one implemented in question 2 and the link between Ingredient and IPrice has been omitted in this diagram.



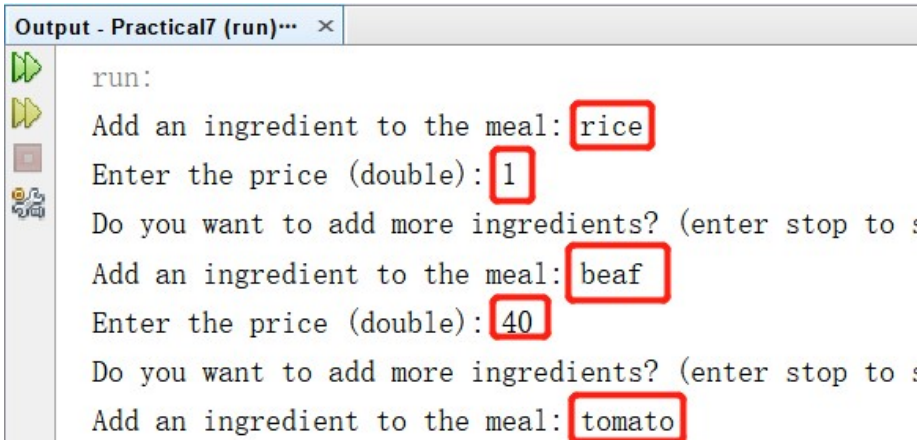
Exercises 3

Write one main class Main.java to use Meal and Ingredient based on the UML above. The requirements as below:

- Create one meal instance with name “Dinner”, ask user to add Ingredients to the meal until the input is “stop”. The ingredients can be one or more than one.
- Print out the meal information and calculate the total bill after the user added the Ingredients.

Testing Case and Result

(The red rectangles are the inputs)



Complete the implementation and testing.

（1） Implementation

Please show your design with some comments in your program and paste the source code here with screenshots.

(The order of the screenshots is as follows: Ingredients.java, Meal.java, Main.java. No need to paste the interface IPrice.java and Ingredient.java)

Ingredients.java

```java
/*
Implement the class Ingredient that implements the interface IPrice. An ingredient has a name and a price and a method
to get the name of the ingredient and the price. The constructor of the class receives the name and the price of the
ingredient as parameters
*/

9 个用法
public class Ingredient implements IPrice {
    //The class Ingredient that implements the interface IPrice
    2 个用法
    private String name;
    2 个用法
    private double price;
    2 个用法
    public Ingredient(String name,double price) {
        //The constructor of the class receives the name and the price of the ingredient as parameters
        this.name = name;
        this.price = price;
    }
    3 个用法
    @Override
    public double getPrice() {
        return price;
    }
    1 个用法
    public String getName() {
        return name;
    }
}
```

Meal.java

```java
import java.util.ArrayList;
/*
Implement the class Meal. A Meal has a name and a list of ingredients and a price. The class implements the Ingredients
and IPrice interfaces. The price of the meal is calculated by adding the prices of all the ingredients.
*/
2 个用法
public class Meal implements IPrice,Ingredients {
    //The class Meal implements the Ingredients and IPrice interfaces
    2 个用法
    private String name;
    3 个用法
    private double price;
    4 个用法
    private ArrayList<Ingredient> ingredients;
    1 个用法
    public Meal(String name) {
        this.name = name;
        this.price = 0;
        this.ingredients = new ArrayList<>();
    }

    1 个用法
    public String getName() {
        return name;
    }

    1 个用法
    @Override
    public ArrayList<Ingredient> getIngredients() {
        return ingredients;
    }
/*
The price of the meal is calculated by adding the prices of all the ingredients
*/
    3 个用法
    @Override
    public double getPrice() {
        for (Ingredient ingredient:ingredients) {
            price += ingredient.getPrice();
        }
        return price;
    }

    2 个用法
    @Override
    public void addIngredient(Ingredient ingredient) {
        ingredients.add(ingredient);
    }
}

```

Main.java

```java
/*
Create one meal instance with name “Dinner”, ask user to add Ingredients to the meal until the input is “stop”.
The ingredients can be one or more than one.
Print out the meal information and calculate the total bill after the user added the Ingredients.
*/
import java.util.Scanner;

public class Main {
    0 个用法
    public static void main(String[] args) {
        Meal meal = new Meal( name: "Dinner");
        //Create one meal instance with name “Dinner”
        Scanner input = new Scanner(System.in);

        /*
        Ask user to add Ingredients to the meal until the input is “stop”
         */
        System.out.print("Add an ingredient to the meal: ");
        String name = input.next();
        System.out.print("Enter the price (double): ");
        double price = input.nextDouble();
        meal.addIngredient(new Ingredient(name,price));

        String result;
        //Declare the result input by users
        do {
            System.out.print("Do you want to add more ingredients? (enter stop to stop): ");
            result = input.next();
            if (result.equals("stop")) {
            } else {
                System.out.print("Add an ingredient to the meal: ");
                name = input.next();
                System.out.print("Enter the price (double): ");
                price = input.nextDouble();
                meal.addIngredient(new Ingredient(name, price));
            }
        } while (!result.equals("stop"));

        /*
        Print out the meal information and calculate the total bill after the user added the Ingredients
         */
        String mealInformation = "";
        for (Ingredient ingredient: meal.getIngredients()) {
            mealInformation += ingredient.getName()+" - "+ingredient.getPrice()+"; ";
        }
        mealInformation += "Total bill: "+meal.getPrice();
        System.out.println(meal.getName()+" with ingredient:"+mealInformation);
    }
}
```

（2） Testing

Testing 1 (Same test case)

```
D:\KaiFa\JAVA\bin\java.exe "-javaagent:D:\KaiFa\IntelliJ IDEA Community Edition
Add an ingredient to the meal: rice
Enter the price (double): 1
Do you want to add more ingredients? (enter stop to stop): yes
Add an ingredient to the meal: beaf
Enter the price (double): 40
Do you want to add more ingredients? (enter stop to stop): yes
Add an ingredient to the meal: tomato
Enter the price (double): 6
Do you want to add more ingredients? (enter stop to stop): stop
Dinner with ingredient:rice - 1.0; beaf - 40.0; tomato - 6.0; Total bill: 47.0

进程已结束,退出代码0
```

Testing 2 (Your own different test case)

```
D:\KaiFa\JAVA\bin\java.exe "-javaagent:D:\KaiFa\IntelliJ IDEA Community E
Add an ingredient to the meal: curry
Enter the price (double): 120
Do you want to add more ingredients? (enter stop to stop): yes
Add an ingredient to the meal: ice-cream
Enter the price (double): 2
Do you want to add more ingredients? (enter stop to stop): stop
Dinner with ingredient:curry - 120.0; ice-cream - 2.0; Total bill: 122.0

进程已结束,退出代码0
```