Lab 12

Requirements:

- Create a Java project named yourStudentId_OOP_Lab12
- Read instructions and create classes needed. You are supposed to create 2 classes at import package, 1 interface and 5 classes (Stock, Fruit, Analyzer, FruitAnalyzer, StockAnalyzer, Company).
- All instance variables are private. Please use public methods to access private instance variables.

Description:

In reality, there are many companies in different fields that need to calculate their revenue. These companies range from fruit stores to large corporations. Today, you have been ordered to develop a system for calculating revenue for various industries, and since the subject of the revenue calculation is different, you need to use "Interface" to reduce the coupling between classes. In this case, Stock and Fruit are APIs that have been written by the previous engineer and cannot be changed. In order to respond to the calculation needs, you should create corresponding classes to hand these objects to be used to a method of an interface. The UML of the exercise problem is presented as Figure 1.

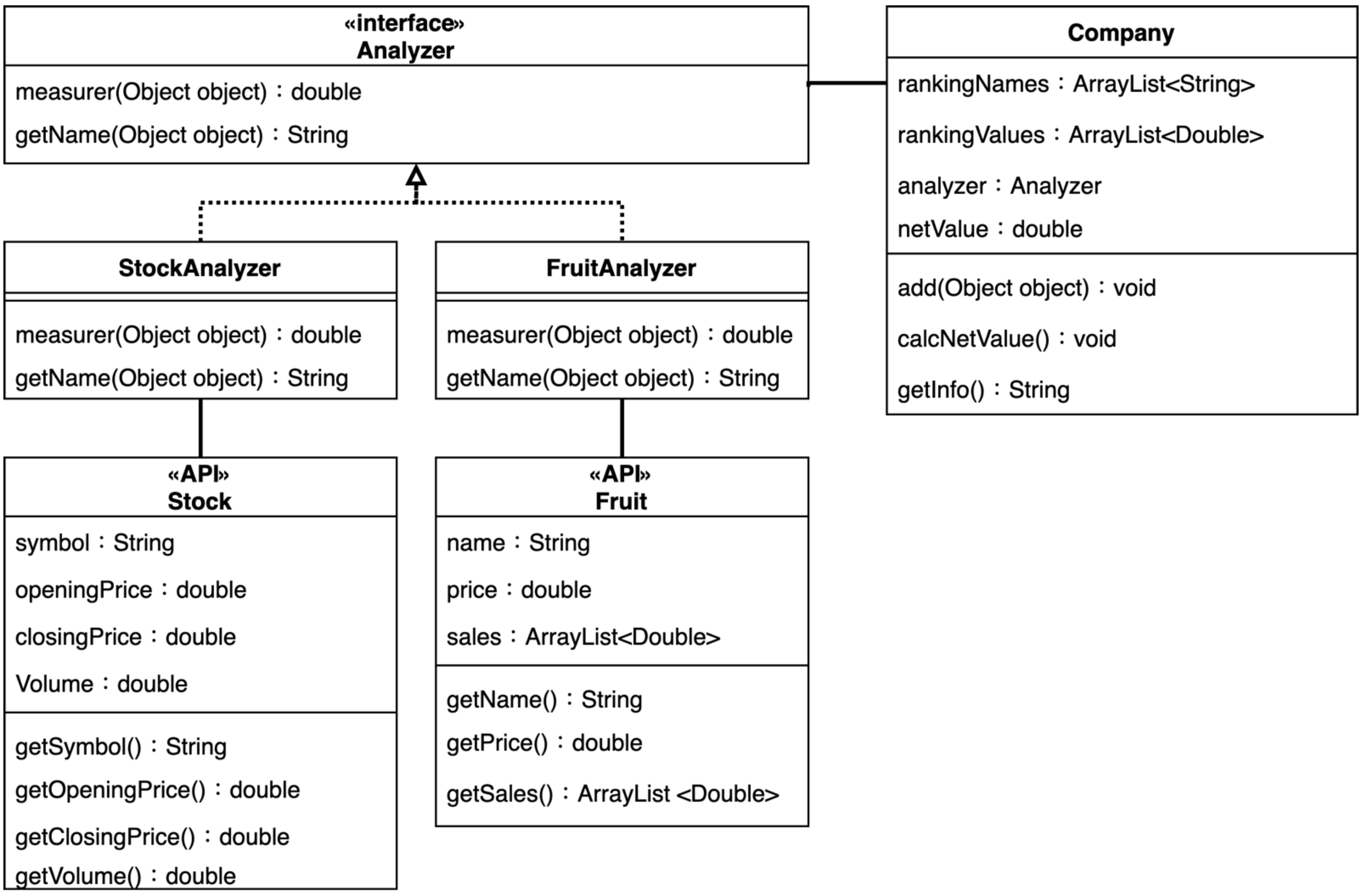
«interface»
Analyzer
measurer(Object object) : double
getName(Object object) : String
Company
rankingNames : ArrayList<String>
rankingValues : ArrayList<Double>
analyzer : Analyzer
netValue : double
add(Object object) : void
calcNetValue() : void
getInfo() : String
StockAnalyzer
measurer(Object object) : double
getName(Object object) : String
FruitAnalyzer
measurer(Object object) : double
getName(Object object) : String
«API»
Stock
symbol : String
openingPrice : double
closingPrice : double
Volume : double
getSymbol() : String
getOpeningPrice() : double
getClosingPrice() : double
getVolume() : double
«API»
Fruit
name : String
price : double
sales : ArrayList<Double>
getName() : String
getPrice() : double
getSales() : ArrayList <Double>

Figure 1. The UML diagram of the exercise problem

Create a package named *lab.practice* containing two classes as *Fruit*, and *Stock*

1. **Create *Fruit class***

Fruit	
Modifier and type	**Method (or Variable) and description**
Instance variable	
String	name The name of the fruit.
double	price The price of the fruit.
ArrayList<Double>	sales The sales of the fruit.
Constructor	
Fruit(String name, double price, ArrayList<Double> sales) Enable to instantiate the object of *Fruit* with given name, price, and sales.	
Instance methods	
String	getName() Return the name of the fruit.
double	getPrice() Return the price of the fruit.
ArrayList <Double>	getSales() Return the sales list of the fruit.

2. **Create Stock class**

Stock	
Modifier and type	**Method (or Variable) and description**
Instance variable	
String	symbol The stock symbol.
double	openingPrice The opening price of the stock.
double	closingPrice The closing price of the stock.
double	volume Number of shares owned
Constructor	
Stock(String symbol, double openingPrice, double closingPrice, double volume) Enable to instantiate the object of Stock with given symbol, openingPrice, closingPrice, and volume.	

Instance methods	
String	getSymbol() Return the stock symbol.
double	getOpeningPrice() Return the opening price of the stock.
double	getClosingPrice() Return the closing price of the stock.
double	getVolume() Return the number of shares owned.

Create the following interface and class in the same package as the main method

3. **Create *Analyzer* interface**

Analyzer	
Modifier and type	**Method (or Variable) and description**
Abstract methods	
double	measurer(Object object) The abstract method is used to calculate the value of the object.
String	getName(Object object) The abstract method is used to get the name of the object.

4. **Create *FruitAnalyzer* class**

FruitAnalyzer	
Modifier and type	**Method (or Variable) and description**
Instance methods	
double	measurer(Object object) A specific callback is used to get the value of the fruit using following formula. **<u>The value of the fruit:</u>** *The value of the fruit = Total sales of the fruit * Unit price*
String	getName(Object object) A specific callback is used to get the name of the fruit.

5. **Create *StockAnalyzer* class**

StockAnalyzer	
Modifier and type	**Method (or Variable) and description**
Instance methods	
double	measurer(Object object) A specific callback is used to get the value of the stock using following formula. **<u>The value of the stock:</u>** *(Closing price - Opening price) * volume * 1000*

String	getName(Object object) A specific callback is used to get the stock symbol.

6. **Create *Company* class**

Company	
Modifier and type	**Method (or Variable) and description**
Instance variable	
ArrayList<String>	rankingNames An arraylist which uses to store the name of the analyzer.
ArrayList<Double>	rankingValues An arraylist which uses to store the value of the analyzer
Analyzer	analyzer An interface used to analyze the value for object.
double	netValue The total value of the company.
Constructor	
Company(Analyzer analyzer) Enable to instantiate the object of *Company* with given analyzer and initialize all Array Lists.	
Instance methods	
void	add(Object object) After transforming the object, use the analyzer to get its value and name, and put it into the corresponding ArrayList.
void	calcNetValue() Calculate and update the net value of all incoming objects.

String	getInfo()
	Return the information as example:
	Example:
	`Net value: 228.00`
	
	`Name          Value`
	`----------------------`
	`Strawberry    120.00`
	`Apple         60.00`
	`Banana        48.00`

Tester
```java
import java.util.ArrayList;
import lab.practice.Fruit;
import lab.practice.Stock;

public class Tester {
    public static void main(String[] args) {
        // TODO Auto-generated method stub

        System.out.println("<<Fruit store>>");
        // Create the sales for every fruit
        ArrayList<Double> sales = new ArrayList<Double>();
        sales.add(1.0);
        sales.add(2.0);
        sales.add(3.0);

        // For fruit store
        Company fruitStore = new Company(new FruitAnalyzer());

        fruitStore.add(new Fruit("Apple", 10, sales));
        fruitStore.add(new Fruit("Banana", 8, sales));
        fruitStore.add(new Fruit("Strawberry", 20, sales));

        System.out.println(fruitStore.getInfo());

        System.out.println("<<Investment company>>");
        // For investment company
        Company investmentCompany = new Company(new StockAnalyzer());

        investmentCompany.add(new Stock("2330", 615, 620, 30));
        investmentCompany.add(new Stock("2317", 117, 119, 20));
        investmentCompany.add(new Stock("2603", 17, 16, 50));

        System.out.println(investmentCompany.getInfo());
    }
}
```

Sample output

```
<<Fruit store>>
Net value: 228.00

Name          Value
----------------------
Strawberry    120.00
Apple         60.00
Banana        48.00
```

```
<<Investment company>>
Net value: 140000.00

Name          Value
----------------------
2330      150000.00
2317       40000.00
2603      -50000.00
```

Submission: Submit your project as ".zip file" via Moodle. No other submissions will be graded.

Reminder: Please zip the whole project

Deadline: Tomorrow's midnight (for both Mon56 and Tue23)