Introduction to Computer Science 2

Lab 2: Interfaces and Polymorphism

Learning Goals:

- To learn how to use interfaces.
- To learn how the principle of polymorphism allows us to write generic classes.
- To learn how to use the strategy pattern.

Exercise 1 (6 points)

Consider interface `Item`:

```
public interface Item {

    public String getName();

    public double getPrice();

}
```

Implement a class `Product` that represents a product you can buy from a supermarket. Every product is represented by its name, price, and bio info (bio info is a text that explains the bio aspects of the product). Supply parametric constructor for the class and make the class `Product` implement the interface Item.

Implement a class `ShoppingBag` whose objects store `Product` objects. Since products can be repeated in a real shopping bag , a `ShoppingBag` object can contain several `Product` objects that have the same name and price; i.e. they correspond to the same product. Supply the following methods for class `ShoppingBag`:

- the default constructor;
- a mutator method `add` that adds a `Product` object to the `ShoppingBag` object;
- an accessor method `totalPriceForProduct` that computes the total price for a product given with its `String` name (for example, if there are two butter `Product` objects in the `ShoppingBag` object and each costs 1.25 euro, then the method outputs the total price for the two butter `Product` objects equal to 2.50 euro).

The implementation details of the class `ShoppingBag` are as follows:

- the class `ShoppingBag` has to store the product data in an `ArrayList<Item>` object;
- the class `Product` has to be defined as a nested class of the class `ShoppingBag`; (you need to decide whether the class `Product` will be an inner class or a `static` class)
- the definitions of the methods of the class `ShoppingBag` have to be as follows:

```
public void add(Item item) // adds an object with class that implements
          // interface Item
public double totalPriceForProduct(String name) // computes the total for
          // price for an item given with name
```

Test the class `ShoppingBag` in the `main` method of this class: create a `ShoppingBag` object, add to this object several objects of class `Product`, and then compute the total price for product objects with the same name.

Exercise 2 (4 points)

Define an interface `Filter` as follows:

```
public interface Filter{
    boolean accept (Object x);
}
```

Consider the `DataSet` class that was designed for the **strategy pattern** (see slides of lecture 2). Modify the method `add` in this class to use a `Measurer` object and a `Filter` object. The method has to process only those objects that are accepted by the `Filter` object. Test the modified `DataSet` class in the `main` method by creating an object of this class that processes several `Car` objects (the `Car` class was implemented for Lab 1). The `DataSet` object has to filter out any `Car` object that can drive less than 100 km. If a `Car` object is accepted, its fuel is measured. After the last `Car` object has been processed, print the average fuel in cars that can drive at least 100 km.

Note that to test the modified `DataSet` class you will need to implement additionally:

- class `CarMeasurer` which objects can measure the fuel of a `Car` object. **Hint:** The class has to implement interface `Measurer`.
- class `CarFilter` which objects filter out any `Car` object that can drive less than 100 km. **Hint:** The class has to implement interface `Filter`.

Honor code, coding style, and deliverable:

Try to solve the exercises with what you already know. You are welcome to expand your program to do extra things but they are not mandatory.

Plagiarism is not allowed! We will run sophisticated software that automatically detects similarities on source code among students. All plagiarism incidents will be immediately reported to the Board of Examiners!

Submission!

Submit your java files to canvas.
Ask your instructor in case there is a problem with your submission.

DO NOT SEND SUBMISSIONS VIA EMAIL
YOUR LAB WILL NOT GET GRADED!