German University in Cairo
Media Engineering and Technology
Prof. Dr. Slim Abdennadher

Computer Programming Lab,
Lab Assignment2

This lab assignment adds on the work of the previous labs, by introducing the concepts of Polymorphism and Abstraction into the project.

Polymorphism

Java Tips

- The method that has the same behaviour for all subclasses should be implemented once in the superclass, and thus inherited by the subclasses.
- The method that has a different behaviour for each and every subclass should be declared in the superclass and overridden in each subclass.
- A method is declared as FINAL if it is not to be overridden by any other subclass.
- When two or more methods in a class have the same name but different input parameters, the method is said to be overloaded. Input parameters can differ in:
  - Number
  - Type
  - Order

Exercise 2-1

Beverage class

The `Beverage` class should have the method `double getActualPrice(double extra)`. Some customers are special and have vouchers for beverages! Those vouchers give them extra discount. In this case, the final price will be $finalPrice = price - price \times ((discount + extra)/100)$.

Exercise 2-2

Cart Class

A visitor to the supermarket might want to buy something and thus will require a cart. The provided `Cart` class performs the basic actions of a cart.

- It should be placed in the `guc.supermarket.cart` package.
- A cart should have an `ArrayList<GroceryProduct> products` to hold the `GroceryProduct`s bought by the customer. This property is read-only.
- Appropriate constructor should be implemented, in order to allow creating a new empty cart.
- You should implement the `void addProduct(GroceryProduct p)` method which adds a new product `p` to the cart.
- You should implement the `double getTotal()` method which returns the total of summing up the actual price of the products in the cart.
- You should implement the `double getTotal(double extra)` method which returns the total of summing up the actual price of the products in the cart taking into consideration the voucher discount `extra`. As stated before, the voucher discount only applies to `beverage`s

- You should override the `String toString()` method to produce a string representing all information about the products inside the cart seperated by a line. For example:
  Name: Coke Zero
  Price: 5.0 L.E.
  Discount: 12.5 %
  Sugar Level: ZERO

  Name: Juhayna Milk
  Price: 10.0 L.E.
  Discount: 25.0 %
  Fat Level: Half-Cream

Abstraction

Java Tips

- Attempting to instantiate an object of an abstract class results in a compilation error.
- You cannot declare constructors or static methods as abstract. Constructors are not inherited and thus they should not be abstracted.
- A method that behaves completely differently in each subclass should be declared as `abstract` in the super class and implemented in each subclass.

Exercise 2-3

Abstraction

Depending on the type of the grocery product, it might need to be kept in a refrigerator or not! Dairy products need to be kept in a refrigerator, while beverages can be left outside. Implement the `boolean refrigerate()` method. Keep in mind Tip no. 3 :)

Your task in this exercise is to figure out which classes and methods should be abstracted.

The Tricky Part!

Exercise 2-4

Do you really get all the concepts?

Given the following pieces of code:

```
GroceryProduct sprite= new Beverage("Sprite", 10, 5, SugarLevel.ADDED_SUGAR);
Beverage sprite2= new Beverage("Sprite", 10, 5, SugarLevel.LIGHT);
System.out.println(sprite.toString());
System.out.println(sprite2.getActualPrice(25));
System.out.println(sprite.getActualPrice(25));
System.out.println(((Beverage)sprite).getActualPrice(25));
System.out.println(sprite.refrigerate());
```

What will happen in each line??

Now imagine `SoftDrink` and `PowerDrink` classes extend the `Beverage` class and that the `Beverage` class will be declared as an `abstract` class. The `refrigerate()` method behaves differently in each subclass. Which of the following should be done:

- Declare the abstract method `abstract boolean refrigerate()` in the `Beverage` class.
- Not to declare the abstract method `abstract boolean refrigerate()` in the `Beverage` class as it is already declared in the `GroceryProduct` class.