CSCI 1302: Software Development

Fall Semester, 2016

Project 2

Instructor: *Dr. Eman Saleh*

Due Date: 10/03/2016
@ 10:30 AM

In this project you will draw a UML diagram to model your application and you will implement the code for the application using an interface, abstract class, and concrete classes. In order to implement a correct behavior, you will use polymorphism.

Project description:

1. Define an interface called **PizzaStore** with the following members:

Data members:

```
TAX=0.07;
S_VEGETERIAN_PRICE= 6.5;
M_VEGETERIAN_PRICE= 10.5;
L_VEGETERIAN_PRICE= 13.5;
S_CHEESE_PRICE=5.5;
M_CHEESE_PRICE=8.o;
S_CHEESE_PRICE=10.0;
S_PEPPERNI_PRICE=6.0;
M_PEPPERNI_PRICE=8.5;
S_PEPPERNI_PRICE=12.0;
DRINK_PRICE=2.0;
double SALAD_PRICE=4.5;
SOUP_PRICE = 5.0;
```

Methods:

```
calculateTotal();
PrintHeading();
calculateTotal();
printInvoice();
displayMenu();
```

2. Define an abstract class called **Order** that implements the interface **PizzaStore**
   Order class should have at least the following members: -

Data Memebrs:

1. **Any needed data members to support good OO design (inheritance and polymorphism**

Methods (only these methods)

1. Setters and getters
2. Override the displayMenu( ) method of interface PizzaStore. This function displays all store menu items and prices (<u>**List items in a tabular form with suitable headings)**</u>

3. Define a concrete class **CusomerOrder** that inherits class Order, with the following members:

   Methods:

   1. Override the method to displayMenu to display the main menu as follows

      Enter Choice (1-3) or hit any key to exit
      1) PIZZA
      2) APETZERS
      3) DRINKS

   2. calculateTotal(): calculates the total amount to be paid by a customer

      total= total+Tax*total;

   3 Any needed methods to allow polymorphism.

4. Define three classes: Pizza, Appetizer and Drink all are subclasses of the class CusomerOrder

4.1 Class Pizza

Defines two enumerated data types: **PizzaType** and **SizeType**

- **PizzaType** has three values: Vegetarian, Cheese and Pepperoni these should be displayed by the Pizza menu
- **PizzaSize** has three values: Small, Medium and Large, you have to read the size and accordingly compute the price.
- Override the **displayMenu** method to display an appropriate the sub-menu Pizza as follows:

  Enter Pizza choice (1-3) or hit any key to exit
  1) Vegetarian
  2) Cheese
  3) Pepperoni

4.2 Class Appetizers

- Define an enumerated data type called **ApptType** that hss two values Soup and Salad these must appear in the appetizer menu by the displayMenue() method.
- Override the **displayMenu** method to display an the sub-menu Appetizers as follows:

Enter Appetizer choice(1-2) or hit any key to exit
1) Salad
2) Soup

4.3 Class drink

Assume only one drink is offered.

Remember to read the quantity. The user may order more than one item of the same type (2 Cheese Pizza) See the sample rum below.

Each sub-menu must continue to be displayed on the screen taking new orders until the user exits this menu.

Whenever the user exits a sub-menu (Pizza, Appetizers, Drinks) an order summary is displayed on the screen showing all ordered items up to this point. Tis should be done by updating the string **output** defined in class **CusomerOrder** and calling the method **displayInvoice().**

Upon Exiting a sub-menu and after displaying the summary, the main menu reappears on the screen.

- **You must define these data members in one of your classes. What is the best class based on OO design principle?**

1. counter: a variable that stores the total number of orders made by all customers. This counter will also be used as invoice number (note that a customer may have many items but all are considered as one order)
2. noOfItems: a variable that stores the total number of items in a single customer order. For example, if a customer ordered 1 Cheese pizza and 2 Vegetarian Pizzas and 2 drinks then noOfItems= 5.
3. total: a variable that stores the total amount to be paid by a customer.
4. output: A string to store invoice information; whenever an item is added to the order this string should append new item information in the form:

```
ItemName      Price          Quantity           totalPrice
```

IF you need to define any variable as static or final state the reason in the documentation file. Use suitable qualifiers for all fields in the class.

5. Lowes level classes should be defined to prevent other classes from extending them.
6. Define A class called PizzaTest that contains the main method to test the pizza store classes. Define only one reference and use it to polymorphically create all needed objects. Or define one array of objects.
   In this class continue displaying the main menu until the customer finishes his order., at this point display a detailed customer invoice. Then display the Pizza Store main menu again ready for another customer. Repeat unless the user chooses to exit the application. Upon exit display a summary of: Total Number of orders, Total number of Pizza orders, total number of appetizers orders and total number of drink orders.

Submission Instructions:

- The project (all classes and the interface) must be placed in one package.
- Submit all the source code files, i.e. .java files. You do not have to include .class files.
- A README.txt file telling us how to compile your program and how to execute it.
- A Documentation.pdf file contains the documentation that contains the following pages:
  1- Cover Page: Your Name, ID, Section CRN and Project Number
  2- Page 1: The UML diagram.
  3- Page 3: Exception handling: Identify areas where you applied exception handling listing an example (code from your program)
  4- Design Notes: In this page list the final, public and static variables used and state the reason behind defining them as final, static, public, or any combination (e.g. static public).

Submit to user `cs1302a` on ***nike*** a directory called `Project2` *(using the command:* *`submit Project2 cs1302a`*).

Evaluation of the projects will include:

1) Evaluating test cases using a pass or fail metric.
2) Implementation must be using polymorphic behavior.
3) Your program must not crash under any input. Use exception handling and display an error message
4) Programming style: Use meaningful names, indentation, comments, …

5) Following object oriented design principles: Encapsulation, information hiding, reusability, inheritance, …

See course syllabus for late submission criteria

Sample Run:

```
            Welcome to Pizza Store
Pizza:
                             Small                Medium              Large
 Vegetarian             $6.5                  $10.5               $13.5
  Cheese                 $5.5                  $8.0                 $10.0
  Pepperoni             $6.0                  $8.5                 $12.0
Appetizers
Salad ……………………………………………………………………………………… $4.5
Soup………………………………………………………………………………………….$5.0
Soft Drinks ………………………………………………………………………………$2.0

Start Order….
Enter Items: Choose (1-3) or hit any key to exit
1) PIZZA
2) APETZERS
3) DRINKS
1
                  Pizza Menue:
Enter Pizza choice (1-3) or hit any key to exit
1) VEGETARIAN
2) CHEESE
3) Pepperoni
1
Please enter size:
 1) Small
 2) Medium
 3) Large
1
Please enter qantity:
3
Enter Pizza choice (1-3) or hit any key to exit
1) VEGETARIAN
2) CHEESE
3) Pepperoni
2
Please enter size:
1) Small
2) Medium
 3) Large
3
Please enter qantity:
2
Enter Pizza choice(1-3) or hit any key to exit
1) VEGETARIAN
2) CHEESE
3) Pepperoni
4
Order Summery
Item                              Price        Qty
Small Vegetarian Pizza    6.5            3
Large Cheese Pizza         5.5            2

More to Order (Y for yes N for no)? y
Enter Items: Choose (1-3) or hit any key to exit
1) PIZZA
2) APETZERS
3) DRINKS
2
```

```
Appetizers Menu
Enter Choice (1-2 or hit any key to exit)
1) Salad
2) Soup
1
Please enter quantity:
2

Appetizers Menu
Enter Choice (1-2 or hit any key to exit)
 1)Salad
 2)Soup
E
Order Summery
Item                        Price       Qty
Small Vegetarian Pizza      6.5          3
Large Cheese Pizza          5.5          2
Soup                        5.0          2

More To Order (Y for yes N for no)? n


Invoice 001

Item                        Price       Qty
Small Vegetarian Pizza      6.5          3
Large Cheese Pizza          5.5          2
Soup                        5.0          2

Total Sales                 $40.5
Tax                         $2.8
Total                       $43.335

System off (Enter N for new customer order or E) to exit System
E

Session Sales Summary:

Total number of orders                   1
Total Number of Pizza Orders             5
Total Number of Appetizers               2
Total Number of Drink orders             0
```