CS102 - Algorithms and Programming I

Lab 01

Lab Objectives: review - classes and objects.

For all labs in CS 102, your solutions must conform to the CS101/102 style guidelines.

LAB RULES:

- You are allowed to refer to your lecture notes and textbooks during lab sessions. You are not allowed to talk or exchange any form of information with your classmates. All solutions should be your own. If you have questions, please consult your TAs or tutors.
- When you are done ask your TA to come by and s/he will ask you questions about the assignment and have you sign the attendance sheet. You must attend the lab session and show your code to the assistants to receive a grade for the lab.

Question 1:

a) Download the classes MenuItem, Customer and MenuApp from Moodle and using these Classes, complete the following:

b) Create class **FoodOrder** according to the following:

   a. Each FoodOrder has a **yemekSepeti** which stores a list of MenuItems.
   b. Add a Constructor which initializes the **yemekSepeti** ArrayList.

c) Implement the following methods:

- **addItemToYemekSepeti()**: takes a MenuItem as a parameter. Finds the first item in the list whose price is more than the price of the item passed as a parameter and inserts the item to the found position. If there are no items with a price more than the parameter , add to the end of the list.
- **emptyYemekSepeti():** deletes all items in the yemekSepeti.
- **calculateTotalCalories():** sums and returns the total calories of all items in the yemekSepeti.
- **searchYemekSepeti():** takes a keyword as a parameter and returns a list of MenuItems whose description contains the given keyword.
- **toString():** returns the String representation of a FoodOrder (you should determine the format).

d) Download class, **Customer**, and add the following:

Data Members:

- **FoodOrder**: stores the customer's FoodOrder.

Methods:

- **Constructor**: update to initialize the FoodOrder.
- **getFoodOrder():** accessor method for the FoodOrder.
- **getTotalCalories ()** – returns the total calories for the FoodOrder. If the customer is a **diet** customer, their food order will have 20% fewer calories.
- **toString()**: returns a String representation of a Customer object.

e) Update the class **MenuApp** to do the following:

- **loadCustomers ():**
  - Method has been partially implemented, add required code according to the comments given.
  - takes a String filename as a parameter and returns an array of Customers in the file.
  - For each customer in the file :
    - reads the customer data,
    - creates Customer object.
  - For each customer, the file includes an int that represents the number of items in their FoodOrder. Read the items from the file and add them to the Customer's food order using the appropriate methods.
  - Each customer should be added to an array list of customers and the method should return the array of customers.

- **main():** the main method should do the following:
  - Create an *array* of Customers and initialize *array* using the loadCustomers() method.
  - Display all customers who have "meat" in their FoodOrder.
  - Empty the FoodOrder of customer with email address entered by the user.
  - Display all customers.