CSE 102 Programming Assignment 1

Due:

Wednesday, 29-March-2023 by 23:59

Deliverables:

The following Java file should be submitted to Google Classroom by the due date and time specified above. Submissions received after the deadline will be subject to the late policy described in the syllabus.

- Assignment01_{StudentNumber}.java

Specifications:

Overview: You will continue the program this semester to maintain the inventory for a store. Do not forget your headers with @author and @since information.

Requirements: Write a set of classes according to the following specifications:

1. Product
   a. Attributes
      i. Id: String
      ii. Name: String
      iii. Quantity: int
      iv. Price: double
   b. Methods
      i. Constructor that takes the ID, name, quantity, and price as parameters
      ii. getId(): String and setId(id: String)
      iii. getName(): String and setName(name: String)
      iv. getPrice(): double and setPrice(price: double)
      v. remaining(): int – returns the current quantity
      vi. addToInventory(amount: int): int – increase count by amount
         1. if amount is negative, do nothing
         2. returns new quantity
      vii. purchase(amount: int): double
         1. decrease count by amount and return the total price (amount X price)
         2. if amount is negative or greater than quantity, do not change count and return 0
      viii. toString(): String – "Product {name} has {quantity} remaining"
      ix. equals(o: Object): boolean – returns true if the passed object is also a Product and has the same price (within 0.001)
2. FoodProduct – a child of Product
   a. Attributes
      i. Calories: int
      ii. Dairy: boolean
      iii. Eggs: boolean
      iv. Peanuts: boolean
      v. Gluten: boolean

  b. Methods
     i. Constructor that takes the ID, name, quantity, price, calories, dairy, peanuts,eggs and gluten as parameters
     ii. getCalories(): int and setCalories(calories: int)
     iii. containsDairy(): boolean, containsEggs(): boolean, containsPeanuts(): boolean, and containsGluten(): boolean

3. CleaningProduct – a child of Product
   a. Attributes
      i. Liquid: boolean
      ii. WhereToUse: String
   b. Methods
      i. Constructor that takes the ID, name, quantity, price, and whereToUse as parameters
      ii. getWhereToUse () and setWhereToUse (size: String)
      iii. isLiquid(): boolean
4. Customer
   a. Attributes
      i. Name: String
   b. Methods
      i. Constructor that takes the name as parameter
      ii. getName(): String and setName(name: String)
      iii. toString(): String – returns the name of the Customer
5. ClubCustomer – a child of Customer
   a. Attributes
      i. Phone: String
      ii. Points: int
   b. Methods
      i. Constructor that takes the name and Phone as parameters and sets points to 0
      ii. getPhone(): String and setPhone(phone: String)
      iii. getPoints(): int
      iv. addPoints(points: int): none
          1. adds the passed points to the Customer's point total
          2. if passed value is negative, does nothing
      v. toString(): String – returns the name of the Customer + " has " {points} + " points"

6. Store
    a. Attributes
        i. Name: String
        ii. Website: String
    b. Methods
        i. Constructor that takes the name and website
            1. Creates an empty list of products
        ii. getName(): String and setName(name: String)
        iii. getWebsite(): String and setWebsite(website: String)
        iv. getInventorySize(): int – returns the number products stored
        v. addProduct(product: Product, index: int)
            1. Adds the passed product to the index
            2. If the index is negative or greater than the size of the list, adds the product to the end of the list
            3. Returns none
        vi. addProduct(product: Product)
            1. Adds the passed product to the end of the list
            2. Returns none
        vii. getProduct(index: int): Product
            1. returns the Product at the position passed
            2. if the index passed is negative or greater than the index of the last entered product, return null
        viii. getProductIndex(p: Product): int
            1. returns the index of the Product passed
            2. if the Product does not exist, return -1

Design: Your program does not require a main method. You are only responsible for creating the six (6) classes described above. An example of how your program should operate is given below:

```
public class Assignment01_123456789 {
   public static void main(String[] args) {
      Store s = new Store("Migros", "www.migros.com.tr");

      Customer c = new Customer("CSE 102");
      System.out.println(c);

      ClubCustomer cc = new ClubCustomer("Club CSE 102", "05551234567");
      cc.addPoints(20);
      cc.addPoints(30);
      System.out.println(cc.getPhone());
      System.out.println(cc);


      Product p = new Product("1234", "Computer", 20, 1000.00);
      FoodProduct fp = new FoodProduct("3456", "Snickers", 100, 2, 250, true, true, true, false);
      CleaningProduct cp = new CleaningProduct("5678", "Mop", 28, 99, false, "Multi-room");

      s.addProduct(p);
      s.addProduct(fp);

      for(int i = 0; i < s.getInventorySize(); i++)
         System.out.println(s.getProduct(i));
      s.addProduct(cp);

      s.addProduct(new Product("4321", "iPhone", 50, 99.00));

      System.out.println(s.getProductIndex(new FoodProduct("8888", "Apples", 500, 1, 50, false, false, false, false)));

      System.out.println(cp.purchase(2));
      if(fp.containsGluten())
         System.out.println("My wife cannot eat or drink " + fp.getName());

       else
          System.out.println("My friend can eat or drink " + fp.getName());
       s.getProduct(0).addToInventory(3);

       for(int i = 0; i < s.getInventorySize(); i++) {
          Product cur = s.getProduct(i);
          System.out.println(cur);
          for(int j = i + 1; j < s.getInventorySize(); j++)
             if(cur.equals(s.getProduct(j)))
                System.out.println(cur.getName() + " is the same price as " + s.getProduct(j).getName());
       }

   }
}
```

```
 ----jGRASP exec: java Assignment01_123456789
CSE 102
05551234567
Club CSE 102 has 50 points
Product Computer has 20 remaining
Product Snickers has 100 remaining
-1
198.0
My wife can eat or drink Snickers
My friend cannot eat or drink Snickers
Product Computer has 23 remaining
Product Snickers has 100 remaining
Product Mop has 26 remaining
Mop is the same price as iPhone
Product iPhone has 50 remaining

 ----jGRASP: operation complete.
```

Code: The file you submit will be named Assignment01_{StudentNumber}. You should put all java classes for this assignment inside of this file as discussed in class.
Test: You are responsible for testing your program. It is important to not rely solely on the examples presented in this Assignment description.

Grading:
MS Teams Submission: If anything is ambiguous, it is your responsibility to ask questions. It is also your responsibility to complete this assignment in a timely manner. Questions regarding this assignment will likely not be answered if received after 17:00 on the due date of the assignment.