CSE 102 Programming Assignment 2

Due:

Monday, 13-April-2020 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.

- Assignment02_{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 and modify the following set of classes (All constructors should take parameters in the order given and initialize empty sets)::

1. Product
    a. Attributes – no change
    b. Methods – all methods from Project 01 with following modifications
        i. setPrice(price: double): None
            1. raises InvalidPriceException if amount is negative
        ii. addToInventory(amount: int): None
            1. raises InvalidAmountException if amount is negative
        iii. purchase(amount: int): double
            1. if amount is negative or greater than count, do not change count and raise InvalidPurchaseException
2. FoodProduct – a child of Product
    a. Attributes – no change
    b. Methods
        i. setCalories(calories: int): None
            1. raises InvalidAmountException if calories is negative
3. ClothingProduct – no change
4. Customer
    a. Attributes – no change (hint: additional attributes will make this class easier to implement)
    b. Methods
        i. addToCart(product: Product, count: int): None
            1. Adds the passed product and number to the customer cart
            2. For this assignment, it will mean calling the purchase() method of the Product passed and adding to the receipt String and totalDue if the purchase is successful
            3. If the purchase() method raises an InvalidPurchaseException, this method catches it and displays a message to the screen beginning with “ERROR: ”

   ii. receipt(): String
      1. returns each Product in the cart on a separate line in the format below
         {Product Name} – {Product Price} X {count} … {total for Product}
         Total Due – {Total amount}
   iii. getTotalDue(): double – returns the total amount due
   iv. pay(amount: double): double
      1. If amount is greater than or equal to total due, displays a "Thank you" message to the screen and returns the amount that should be given as change
      2. If amount is less than total due, raises a NotEnoughPaymentException
5. ClubCustomer – no change
6. Store
   a. Attributes
      i. An array or Arraylist of ClubCustomer objects
      ii. An array or Arraylist of Product objects
   b. Methods
      i. Constructor no longer takes number for the array. The size set of Products and ClubCustomers will now be dynamic
      ii. getCount(): int – returns the number of Products saved
      iii. addProduct(product: Product): None
         1. Adds the passed Product to the set
      iv. addCustomer(customer: ClubCustomer): None
         1. Adds the passed Customer to the set
      v. getProduct(name: String): Product
         1. returns the Product with the name passed
         2. if the name is not found in the set, raises an ProductNotFoundException
      vi. getCustomer(phone: String): ClubCustomer
         1. returns the ClubCustomer with the phone number passed
         2. if the phone number is not found in the set, raises an CustomerNotFoundException
      vii. removeProduct(name: String): None
         1. removes the Product with the name passed from the set
         2. if the name is not found in the set, raises an ProductNotFoundException
      viii. removeCustomer(phone: String): None
         1. removes the ClubCustomer with the phone number passed from the set
         2. if the phone number is not found in the set, raises an CustomerNotFoundException

7. Custom Exceptions (All must be instances of RunTimeException)
    a. InvalidPriceException
        i. Additional Attribute – price: double
        ii. toString() “InvalidPriceException: “ + price
    b. InvalidAmountException
        i. Additional Attribute – amount: int
        ii. toString() “InvalidAmountException: “ + amount
    c. InvalidPurchaseException
        i. Additional Attributes – amount: int, remaining: int
        ii. toString() “InvalidPurchaseException: “ + amount + “ requested, “ + remaining + “ remaining”
    d. NotEnoughPaymentException
        i. Additional Attributes – amount: double, due: double
        ii. toString() “NonEnoughPaymentException: “ + due + “ due, but only “ + amount + “ given”
    e. ProductNotFoundException
        i. Additional Attribute – name: String
        ii. toString() “ProductNotFoundException: “ + name
    f. CustomerNotFoundException
        i. Additional Attribute – phone: String
        ii. toString() “CustomerNotFoundException: “ + phone

Design: Your program does not require a main method. You are only responsible for creating the six (6) classes and six (6) Exceptions described above.
Code: The file you submit will be named Assignment02_{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. It would be a very good idea to write your own test cases for this assignment.

Grading:
Google Classroom 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.

```java
public class Assignment02_123456789 {
   public static void main(String[] args) {
      Store s = new Store("Migros", "www.migros.com.tr");

      Customer c = new Customer("CSE 102");

      ClubCustomer cc = new ClubCustomer("Club CSE 102", "05551234567");
      s.addCustomer(c); // compiler error because c is Customer not ClubCustomer
      s.addCustomer(cc);

      Product p = new Product("Computer", 1000.00, 20);
      FoodProduct fp = new FoodProduct("Snickers", 2, 100, 250, false, true, false);
      ClothingProduct cp = new ClothingProduct("Shoes", 89, 28, "44");

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

      System.out.println(s.getCount());
      System.out.println(s.getProduct("shoes"));

      System.out.println(cp.purchase(2));
      s.getProduct("Computer").addToInventory(3);
      System.out.println(fp.purchase(200));  // results in Exception

      c.addToCart(p, 2);
      c.addToCart(s.getProduct("snickers"), -2); // NOTE: This does not stop the program because the Exception is caught
      c.addToCart(s.getProduct("snickers"), 1);
      System.out.println("Total due - " + c.getTotalDue());
      System.out.println("\n\nReceipt:\n" + c.receipt());

      System.out.println("After paying: " + c.pay(2000)); // results in Exception

      System.out.println("After paying: " + c.pay(2020));

      System.out.println("Total due - " + c.getTotalDue());
      System.out.println("\n\nReceipt:\n" + c.receipt());

   }
}
```

An example main() method.

```
Assignment02_123456789.java:10: error: incompatible types: Customer cannot be converted to ClubCustomer
      s.addCustomer(c); // compiler error because c is Customer not ClubCustomer
```

Compiler error

```
3
Product Shoes has 28 remaining
178.0
Exception in thread "main" InvalidPurchaseException: 200 requested, 100 remaining
    at Product.purchase(Assignment02_123456789.java:79)
    at Assignment02_123456789.main(Assignment02_123456789.java:26)
```

Commenting out compiler error gets first Runtime error (“fp.purchase(200)”)

```
3
Product Shoes has 28 remaining
178.0
ERROR: InvalidPurchaseException: -2 requested, 100 remaining
Total due - 2002.0


Receipt:
Computer - 1000.0 X 2 ... 2000.0
Snickers - 2.0 X 1 ... 2.0

------------------------------------------------

Total Due - 2002.0

Exception in thread "main" NotEnoughPaymentException: 2002.0 due, but only 2000.0 given
    at Customer.pay(Assignment02_123456789.java:173)
    at Assignment02_123456789.main(Assignment02_123456789.java:34)
```

Commenting out first Exception line gets second Exception (“c.pay(2000)”)

```
3
Product Shoes has 28 remaining
178.0
ERROR: InvalidPurchaseException: -2 requested, 100 remaining
Total due - 2002.0


Receipt:
Computer - 1000.0 X 2 ... 2000.0
Snickers - 2.0 X 1 ... 2.0

------------------------------------------------

Total Due - 2002.0

Thank you for your business
After paying: 18.0
Total due - 0.0


Receipt:

------------------------------------------------

Total Due - 0.0
```