CSCI 1302 Introduction to Programming Principles

Student Name: Alexander Fields

**Due: September 22, 2016 by 1:55pm (Folio Dropbox)**

**Project: Person, Customer & PreferredCustomer Classes**

Problem Description:

Design a class named **Person** with fields for holding a person’s name, address, date of birth and telephone number. The Person class has only one constructor and that single constructor has arguments for name and date of birth only. Also write the appropriate mutator and accessor methods for the class’s fields.

Next, design a class named **Customer**, which extends the Person class. The Customer class should have fields for a customer number, email, the amount of the customer’s purchases, tax rate of 6% and a boolean field indicating whether the customer wishes to be on a mailing list. Write one or more constructors and the appropriate mutator and accessor methods for the class’s fields, **as well as a method that computes the total due by the customer.**

A retail store has a preferred customer plan where customers can earn discounts on their purchases. The amount of a customer’s discount is determined by the amount of the customer’s cumulative purchases in the store as follows:

* When a preferred customer spends $700, he or she gets 7% discount on all future purchases
* When a preferred customer spends $1500 he or she gets a 10% discount on all future purchases
* When a preferred customer spends $2500, he or she gets 15% discount on all future purchases
* When a preferred customer spends $3000 he or she gets a 20% discount on all future purchases

Design a class named **PreferredCustomer**, which inherits from the Customer class you created earlier. The PreferredCustomer class should have a field for the customer’s discount level. Write one or more constructors and the appropriate accessors methods. The PreferredCustomerm Class should two additional methods:

* getDiscountPrice(double amount):returns the discounted price of **amount** based on the current discount level (should not change the discount level)
* addPurchase(double discountedPrice) that adds the customer discounted price to his or her amount of purchases, and updates the current discount level if needed. (You should create a private method that will updates the discount level based on the amount of purchases.)

Write a client (a test class with the main method) to **test all the methods and constructors** of your classes.

Create a UML diagram that represents your classes and their relationship to each other(do not include your tester class).

Analysis & Design:

(Describe the problem including input and output in your own words. Describe the major steps for solving the problem)

Once all the getters and setters and constructors are made you need to get the tax rate equal to .06 as the tax rate is 6%. Then the discount can be evaluated using if/ else if/ else in order to find the discount rate or even if it is deserving of a discount. Then once can add on extra purchases if they so chose.

**Person**-

name:String

address:String

dob:String

tn:String

+Person()

+Person(String, String, String)

+getName():String

+setName(String):void

+getAddress():String

+setAddress(String):void

+getDob():String

+setDob(String):void

+getTn():String

+setTn(String):Void

**Customer**-extends person

cn:String

email:String

purch:double

taxr:double

mailist:boolean

+Customer()

+Customer(String, double, double, String, boolean)

+getCn():String

+setCn(String):void

+getEmail():String

+setEmail(String):void

+getPurch():double

+setPurch(double):void

+getTaxr():double

+setTaxr(double):Void

+getMailist():boolean

+setMailist(boolean):Void

**PreferredCustomer**- extends Customer

name:String

discount:double

discountPrice:double

addPurchase:double

+PreferredCustomer()

+PreferredCustomer(String, double, double, double)

+getName():String

+setName(String):void

+getDiscount():double

+setDiscount(double):void

+getDiscountPrice():double

+setDiscount(double, double):void

+getaddPurchase():double

+setaddPurchase(double):Void

-addPurchase():double

Submit the following items:

1. Complete this Word file and Submit along with your UML class diagram to the dropbox.

2. Compile, Run, and Test your code then **submit the .java files** to the dropbox.