Homework 1

CS 211

Due Date: 1/18/2018 11:59pm

/ Instructions

This should be review from your CS210 course.

Create two Classes (Purchase.java and MyBasket.java) which support my desktop purchasing software BasketFrame.java (this GUI code is provided). The two classes you create (Purchase and MyBasket) should have:

- No System.out calls
- No TODO comments
- No ArrayList Class usage (we'll do this assignment again in Chapter 10 when we get to that topic)

Purchase.java specifications (for each item purchased, the name, how many and cost per):

- Three fields, a String for name of the purchase, int for units purchased, and a double for price per unit.
- Standard accessors and modifier methods for each field. Constructor to initialize these three fields (String, int, double) in that order.
- Constructor overload, (String, double) assumes the int quantity is zero.
- Default constructor that assumes name is “” and numbers are zero, must call the three-argument constructor.
- A getCost method that is simply the number of units purchased times unit cost.
- A toString method to produce output as shown in examples

MyBasket.java specifications (like a shopping cart, contains the purchases from above class):

- A field private Purchase[] as an array of purchases.
- Another int field that tracks how many purchases actually made.
- Accessor .length() method that returns your int field.
- Accessor .get(int) for the Purchase array needs a parameter that will index the array. So get(0) returns the first element (a Purchase object) of the array.
- Default constructor that creates a new array Purchase[100], and initializes number of purchases to zero.
- Overloaded add method:
    - add(Purchase, int) say we are purchasing an int number of that item. This is tricky, because if you already have that purchase in the basket, you must change it (or remove it) so the new quantity is correct for the second passed parameter. We do not just add more, it’s a replacement process.
    - add(Purchase) will leave the quantity purchased at whatever is already in the Purchase parameter.

- A getTotalCost() method which is also a little tricky, as you need to go through the array of current purchases and add up the grand total of everything purchased so far. Be certain your totals match what I've shown in examples.

Be particularly careful that the program user can add, subtract (change quantity to zero), replace, and modify quantity as many times as they want. And you program always displays the correct total. The trickiest part of this assignment is how to accommodate those changes while shopping. You might get the right total at first, but then change a purchase to zero, and change it back, and change, and change, and be certain you get back to the right totals (see examples).

/ Included Files

BasketCase.java – this file is the console application driver for testing your 2 classes.

BasketFrame.java – this file is the GUI class driver for testing your 2 classes.

You should not make any alterations to either file. It is your job to integrate with them.

/ Hints & Advice

Write your classes and test them with BasketCase.java first! This is a simple console application and is easier to debug with.

Only after BasketCase.java works, then move on to testing with BasketFrame.java.

/ Submission Instructions

Create a zip file that contains your and upload the zip file to Canvas (tar.gz is also ok).