Minor Assignment 5

Yard Sale Calculator

Note: This was a problem on the UW CSE 142 exam a while back. They had to write out the code by hand on a timed exam (yuck!).

Instructions: Create a new project in Eclipse, download the outline of the `yardSale.java` file from Canvas, and import it into your project. **Put a comment with your name at the top**. This indicates that the work is yours alone, as stated in the Collaboration and Academic Dishonesty page.

Fill in the methods appropriately, test your code, and submit the completed .java file to Canvas by the due date.

Please refer to the "Getting Help" and "Academic Dishonesty" portions of the syllabus for direction on how to get help. You are always welcome to message your instructor. Make use of this if you are stuck!

Background: We want to create a simple interactive program that models a user buying examining and buying items from a yard sale.

The Problem: Write a method named `yardSale` that accepts two parameters: a Scanner and an initial amount of money, and returns the amount of money the user ends up with after a series of purchases.

Your method should prompt the user for a series of potential purchases specified by a price per single item and the quantity of that item, until the user's available amount of money is less than $5.

The user only buys items that are a "good deal", which means the price per single item is under $10.

A message should be printed when a purchase is made indicating that the purchase was made, and the remaining amount of money.

Once the user's remaining money is less than $5, your method should print the total quantity of items purchased and the total cost (not just cost per single item) of the most expensive item purchased.

Results do not need to be rounded after the decimal place, **and you may assume the user will always type valid input** (you're welcome).

See the next page for an example.

Example:

The following call, for a scanner named `s`,

```
double remainingBudget = yardSale(s, 50.75) ;
```

would generate an interaction like this (user input underlined and bold)

```
Price? 8.75
Quantity? 2
What a deal! I'll buy it.
Remaining money: $33.25

Price? 11.50
Quantity? 3
Remaining money: $33.25

Price? 5.10
Quantity? 5
What a deal! I'll buy it.
Remaining money: $7.75

Price? 6.95
Quantity? 2
Remaining money: $7.75

Price? 0.75
Quantity? 8
What a deal! I'll buy it.
Remaining money: $1.75

Total quantity purchased: 15
Most expensive purchase: $25.5
```

You must exactly reproduce the format of this sample execution. Notice that the second item is not purchased because it is not a good deal, and the fourth item is not purchased because the total cost of $13.90 exceeds the amount of money the user has left ($7.75). The most expensive purchase is based on the total cost (5 items at $5.10) rather than the cost per single item (2 items at $8.75).

This sample call would return 1.75 as the remaining money because the user spent a total of $49.00 (17.50 + 25.50 + 6.00) and started with $50.75.