Project
Maggie's POS (Point-of-Sale) System

Course:	INFO1150, W2017
Professor:	Janice Manning
Submitting:	Please see the last page for instructions
Due Date:	Saturday, April 1, 2017 by 11:59 pm

What are the Marks Awarded For?	Marks Available	Marks Assigned
Coding Style		
• Header with description		
• Appropriate pseudo-code statements have been done		
• Internal comments, end-brace included, accurately typed		
• Correct use of indenting , correct use of white space		
• Naming conventions have been followed for variable, constant, and method names and are all representative of what each does	2	
Functional Code		
• Program provides user with title and instructions and user name is entered and stored	1	
• Array of appropriate type and specified size is created to record purchase price amounts	1	
• A sentinel loop structure is used to get and process up to 10 purchase prices	1	
• Purchase inputs are properly read in	1	
• Data validation is done on prices, error message is displayed if invalid, and invalid input can be re-entered	2	
• Valid purchase prices are added to a purchase array	1	
• Each valid entry is echoed back to the user, along with an updated subtotal	1	
• A zero entry signals the process of reducing the previous purchase price entry from both the array and the subtotal, as long as there was a previous purchase price entry	2	
• When the loop is done, output includes:		
○ purchases and subtotal		
○ HST is calculated and rounded to two decimal places		
○ Grand total is rounded to nearest 5 cents	1	
• Tender amount input is properly read in	1	
• Data validation is done on tender amount, error message is displayed if invalid, and invalid input can be re-entered	1	
• Change is calculated and rounded to nearest 5 cents and displayed	1	
• Change denominations are calculated and displayed as either singular or plural amounts	2	
Methods		
• calculateHST(double amount)		
○ using amount, calculates and returns HST rounded to two decimal places	2	
• roundToNearestFive(double amount)		
○ using amount, returns amount rounded to the nearest 5 cents	2	
Output		
• Program compiles without errors and runs without crashing	1	
• Output is properly formatted with correct use of white space, without spelling and grammatical errors, and a currency format is used for money amounts	2	
TOTAL	25	

Program Description:

You have been hired by Maggie Jones, a retailer at the Gibraltar Trade Centre in London, to produce a point-of-sale (POS) application to record sales transactions. Currently all sales are recorded on paper, which can be time consuming if the customer buys more than two or three items. The app will eventually be ported to Android tablet devices used by the retailer's sales staff, but first she wants to see a prototype that can demonstrate the basic requirements of handling a sale.

Your task is to write the very first iteration of some prototype code as a Java console application. **Note that you are doing just the first iteration**. In real life, your prototype would be refined by your coding staff through many iterations using a rapid application development approach and eventually it would become the self-serve point-of-sale system for the store.

A description of how the user interacts with the program (the *user story*) is as follows. After the user has made his selections at the store, he will go to the checkout counter where your program will be operating. Eventually a bar code scanner input system using Android tablet devices will be implemented, but for this prototype code, the user will just enter the price of each item using a keypad. A message on the screen welcomes the user to the checkout and the program tells the user how to proceed. The on-screen instructions would follow a dialogue pretty similar to what is shown in the example below.

Hi. Welcome to Maggie's!
What is your name? *(user enters name here, and then the following instructions appear)*

OK, *userName*, enter the price of each of your purchases in dollars and cents and hit the ENTER key.
For example, if your item costs $5.99, you would enter 5.99.

If you make a mistake when you enter a price, just enter a zero for the next entry. The last price you entered will be subtracted from your subtotal.

When you have entered all of your prices, enter a negative 1 (-1) to indicate that you are done.
I'll then calculate what your total owing is.

Enter a price for item #1: $ *(user then begins entering prices)*

Program Specifications:

1. Start a new project called **Project**. Then create a new class called **PointOfSale.java**

2. First, write an algorithm (pseudo-code) for this program, and include it in a comment block at the top of your program, just below your program identification block. You can then copy and paste the portions of the algorithm to use as comments into each section of your code. Your algorithm may be more than one page long if you feel it is necessary, but the most important thing is that it clearly outlines the steps you will follow as you write your program.

3. Display a title and instructions for the application. (*See the sample output next page.*) Instructions to the user must be clear and concise. A user name must be obtained from the user.

4. Set up an appropriate loop structure with appropriate sentinel values.

5. Your program will store each of the prices entered by the user in an array so that they can all be displayed in the final output.

6. A user may purchase up to 10 items, and he will enter a sentinel of -1 when he is done. Since you will not know how many items the user might be purchasing, **set the size of the array to hold a maximum of 10 purchase prices**.

7. Your program must keep a running subtotal of the prices entered. Each price the user enters must be "echoed" back to him as a visual confirmation of what he's entered, plus display the current subtotal.

8. There is no item in Maggie's store that costs more than $99.99. Do some data validation to make sure the user cannot enter an amount higher than this. Similarly, a user should not be able to enter negative values for prices (this would reduce the total that they owe!), other than the -1 that serves as the sentinel value.

9. If the user enters an incorrect price, he can enter a zero to remove the last entry. Your code will adjust the subtotal and also remove the incorrect entry from the array. The program should show the removal of the last price. (*See the sample output next page.*)

10. When user enters -1 to indicate he is done, program will calculate and display the following information:
    - the price of each item purchased
    - the subtotal before HST
    - the amount of sales tax (HST) owing, rounded to two decimal places
    - the Grand Total of purchases plus taxes, rounded to the nearest 5 cents

11. You will need to write two separate methods to do the following calculations:

    a. The first method will be called calculateHST(double amount), and it will calculate the amount of sales tax or HST that must be paid. It will take as an argument the subtotal, and it will calculate the amount of Harmonized Sales Tax owing. This method must also round the HST amount to two decimal places before returning its value. The current HST rate in Ontario is 13%.

    b. The second method will be called roundToNearestFive(double amount). This method will take as its argument the grand total owing, and it will return the grand total rounded to the nearest nickel. For example, if the grand total is $39.57, it would *round down* to $39.55, or the grand total is $39.58, it would *round up* to $39.60.

12. Format your currency amounts in output. If after rounding to the nearest nickel the cents amount ends in a zero (i.e. $29.60) Java will not print the trailing zero. Instead, it will display this: $29.6, which is not what we want. You will need to apply some FORMATTING to the value to get it to show the trailing zero. There are several ways to do this. You might want to use the JDK Docs and research the DecimalFormat or NumberFormat classes, or check Chapter 4 in your textbook for information about the 'printf' method and how it can be used.

13. Then, calculate the change to give to the user. After the user sees the grand total owed, he then enters a "tender amount", that is, the amount of cash he will be putting into the payment slot. Your program will validate that the tender amount is sufficient to pay the grand total. Your program will then calculate the amount of change, rounded to the nearest nickel using the above method.

14. If the change is greater than $0.05, calculate the change in exact denominations of the number of bills and coins to give back to the user. NOTE: the largest bill that the checkout will return is a $20 bill; it does not hold $50 or $100 bills to minimize losses in case of a robbery. And, its smallest coin is a nickel; it does not hold pennies.

    Your program will display the user's change in correct denominations, along their units displayed as either singular or plural amounts, as long as the change is greater than $0.50.

15. Finally, be sure your program will adjust its output if the user enters -1 as the first entry.

Sample Output:

```
*****************************************************************
                Maggie's POS Checkout System Ver. 1.0
*****************************************************************
Instructions:
1.  Enter your name for our system.
2.  Then, enter the price of each of your purchases.
    Maximum price for any purchase is $99.99.
3.  If you make a mistake, enter a zero.
    The last price will be subtracted from your subtotal.
4.  You may purchase up to 10 items at a time.
5.  When you are done, enter -1 to signal the end of your purchases.
*****************************************************************
Please enter your name:  Gary

Welcome, Gary, to our self-serve, automated check-out system!
You may begin checking out now . . .

Enter a price for Item #1:  $19.99
That was $19.99.  Your subtotal:  $19.99

Enter a price for Item #2:  $12.50
That was $12.50.  Your subtotal:  $32.49

Enter a price for Item #3:  $17.43
That was $17.43.  Your subtotal:  $49.92

Enter a price for Item #4:  $28.11
That was $28.11.  Your subtotal:  $78.03

Enter a price for Item #5:  $0
ZERO ENTERED:  removing last item $28.11 . . .
Subtotal has been corrected:  $49.92

Enter a price for Item #4:  $7833
INVALID PRICE:  please re-enter . . .
Enter a price for Item #4:  $78.33
That was $78.33.  Your subtotal:  $128.25

Enter a price for Item #5:  $-1

Gary, here are your purchases:
$19.99
$12.50
$17.43
$78.33
Subtotal:         $128.25
HST:              $16.67
Grand Total:      $144.90

Enter your tender amount:  $144
INSUFFICIENT FUNDS:  Re-enter your tender amount:  $200

From $200.00 your change is $55.10
    2 $20 bills
    1 $10 bill
    1 $5 bill
    1 dime
Please check that your change received is correct.

Thank you for shopping at Maggie's, Gary.  Please come again!
```

Testing your program:

You should test your program for the following and ensures your program behaves as expected:

1. If a user enters -1 as the first purchase price, your program should conclude and display its final statement:
   "Thank you for shopping at Maggie's, Gary. Please come again!"
2. If a user enters 0 as the first purchase price, your program should display:
   "ZERO ENTERED: There are no items to remove . . ."
3. If a user enters an invalid purchase price, your program should inform the user and prompt for the previous purchase price.

Submitting your work:

Submit your *PointOfSale.java* file to the Project Dropbox in FOL by due date indicated on page 1 of this document.
If your teacher prefers a zipped .java file, please zip it up.

Submit your project on time!

Code submissions to the FOL drop box must be made on time! Late projects will be subject to late penalties as follows:

- up to 24 hours late - 20% deduction
- over 24 hours late but less than 48 hours late - an additional 30% off
- over 48 hours late - mark of zero is given

Submit your own work! ***YOU MUST WRITE YOUR OWN CODE!***

It is considered cheating to submit work done by another student or from another source as your own work. This is called plagiarism. Helping another student cheat by letting them copy your source code files is called abetting plagiarism. Both are considered to be academic offences.

Students are encouraged to share ideas and to work together on practice exercises, and you can help someone else to fix a bug in their code, but any code or documentation prepared for a project must be done by you. Penalties for committing plagiarism, or helping another student plagiarize by sharing source code with them, include a zero grade and an academic offence being filed against the student or students involved. All submissions will be analyzed using plagiarism detection software especially designed to analyze Java code.