CSCI 140 -- Project 1 (Vending Machine)

Due Thursday, 03/23/2017 (TTh Section)
Due Wednesday, 03/22/2017 (MW Section)

Write a program to work with a vending machine that accepts only a one-dollar bill and returns changes in quarters, dimes, and nickels. Upon startup of this program, the machine is filled with a number of coins once for each denomination from keyboard input before any purchase. This vending machine will only accept one-dollar bill and all items cost one dollar or less, but the cost for each item must be a multiple of 5 cents.

Your program would read an integer value between 0 and 100 (inclusive), representing the amount of a purchase in cents from a vending machine. Produce an error message if the input value is not in that range or it is not a multiple of 5 cents. If the input is valid, determine the amount of change that would be returned from one dollar, and print the number of quarters, dimes, and nickels. **It is important to maximize the coins with the highest value first and utilize the next denomination if the current denomination is exhausted**. This strategy will work for most cases, but there are a few cases where it wouldn't work (such as Q = 2, D = 4, N = 0, and you need to give back 55 cents) and you do not have to handle such situations (go ahead and reject such transaction as insufficient coins). Reject a valid purchase if it cannot be processed (not enough available coins to make the change) but program must allow more transactions. The machine must be able to keep track of an accurate count of coins in the machine at all time. Your program must allow purchases to be repeated until a sentinel value of 0 is entered. Print number of dollar bills, the number of coins for each denominations, and current balance just before terminating the program. Follow the format below and **you must plan and write down the pseudocode before attempting your code** on the computer. Although functions are not required for this project, you should try to utilize functions so your program is more readable. You must follow the user-interface as shown below and run those test cases at the minimum:

```
Vending Machine Version 1 by [Your Name]

Fill up machine with changes.  Please wait ...
Enter number of quarters, dimes, and nickels --> 2 4 4<Enter>
There are 2 quarters, 4 dimes, and 4 nickels.
Initial machine balance is $1.10.
Only one-dollar bill will be accepted.
Only amount between 0 to 100 and multiple of 5 is accepted.
Enter 0 to stop the program.
Machine is now ready.

Enter a purchase amount [0 - 100] --> 35<Enter>
You entered a purchase amount of 35 cents.
Please insert one-dollar bill.
Processing your purchase ...
Your change of 65 cents is given as:
    quarter(s): 2
    dime(s):    1
    nickel(s):  1

Enter a purchase amount [0 - 100] --> 31<Enter>
```

```
You entered a purchase amount of 31 cents.
Invalid amount (not a multiple 5).  Try again.

Enter a purchase amount [0 - 100] --> 35<Enter>
You entered a purchase amount of 35 cents.
Please insert one-dollar bill.
Processing your purchase ...
Insufficient changes!
Your transaction cannot be processed.
Please take back your dollar bill.

Enter a purchase amount [0 - 100] --> 75<Enter>
You entered a purchase amount of 75 cents.
Please insert one-dollar bill.
Processing your purchase ...
Your change of 25 cents is given as:
    quarter(s): 0
    dime(s):    2
    nickel(s):  1

Enter a purchase amount [0 - 100] --> 105<Enter>
You entered a purchase amount of 105 cents.
Invalid amount (outside valid range).  Try again.

Enter a purchase amount [0 - 100] --> 0<Enter>
Final report is being generated ...
Number of dollars : 2
Number of quarters: 0
Number of dimes   : 1
Number of nickels : 2
Machine balance   : $2.20
Machine is shutting down.  Good bye.
```

Extra credit: You would only earn extra credit points for either extra credit 1 or 2 and not both.

Extra credit 1: You can earn up to 4 addition points if your program allows user to use a dollar bill or coins and you can submit just one version (`Vending Machine Version 1A)`. Prompt user to select an option: dollar bill (1) or coins (2). For example, if user selected coins then user can use 1 dime and 3 quarters for a purchase amount of 80 cents. User would be able to enter a list of coins in any order and 0 will be used as a sentinel value (i.e., 25 10 25 25 0). Ignore invalid coins such as 2 or 13 and make sure that a sufficient amount is entered; otherwise return the money. Make sure to run original test cases above using option 1 and the test case below. Your interface would change as follow:

```
Vending Machine Version 1A by [Your Name]

Fill up machine with changes.  Please wait ...
Enter number of quarters, dimes, and nickels --> 2 4 4<Enter>
There are 2 quarters, 4 dimes, and 4 nickels.
Initial machine balance is $1.10.
One-dollar bill or coins will be accepted.
Only amount between 0 to 100 and multiple of 5 is accepted.
Enter 0 to stop the program.
Machine is now ready.


Enter a purchase amount [0 - 100] --> 80<Enter>
You entered a purchase amount of 80 cents.
Select an option (1 - dollar bill and 2 - coins) --> 2<Enter>
Please insert your coins --> 25 10 25 5 0<Enter>
You inserted an amount of 65 cents.
Insufficient amount! Returning all coins.
```

```
Please insert your coins --> 25 10 25 25 13 0<Enter>
You inserted an amount of 85 cents.
Processing your purchase ...
Your change of 5 cents is given as:
    quarter(s): 0
    dime(s):    0
    nickel(s):  1

Enter a purchase amount [0 - 100] --> 0<Enter>
Final report is being generated ...
Number of dollars : 0
Number of quarters: 5
Number of dimes   : 5
Number of nickels : 3
Machine balance   : $1.90
Machine is shutting down.  Good bye.
```

Extra credit 2: You can earn up to 4 addition points if your program can handle special cases as discussed earlier (such as Q = 2, D = 4, N = 0, and you need to give back 55 cents) and you can submit just one version (`Vending Machine Version 1B)`. Make sure to run original test cases above and the test case below. Come up with another test case where the original program would not be able to handle and try it out with this version. You would only earn extra credit points for either extra credit 1 or 2 and not both.

```
Vending Machine Version 1B by [Your Name]

Fill up machine with changes.  Please wait ...
Enter number of quarters, dimes, and nickels --> 2 4 0<Enter>
There are 2 quarters, 4 dimes, and 0 nickels.
Initial machine balance is $0.90.
Only one-dollar bill will be accepted.
Only amount between 0 to 100 and multiple of 5 is accepted.
Enter 0 to stop the program.
Machine is now ready.

Enter a purchase amount [0 - 100] --> 45<Enter>
You entered a purchase amount of 45 cents.
Please insert one-dollar bill.
Processing your purchase ...
Your change of 55 cents is given as:
    quarter(s): 1
    dime(s):    3
    nickel(s):  0

Enter a purchase amount [0 - 100] --> 0<Enter>
Final report is being generated ...
Number of dollars : 1
Number of quarters: 1
Number of dimes   : 1
Number of nickels : 0
Machine balance   : $1.35
Machine is shutting down.  Good bye.
```

Please provide documentation and applying good coding style because it is part of the grade. Do not forget to use the provided template. You must run the specified test cases and come up with additional test cases if applicable since the test cases are also part of the grade. Please submit a hardcopy of the following items **in a folder** if flash drive or CD is included (can also submit source code ahead of time via Moodlerooms (MR) and a folder is not needed). You can also submit just a hardcopy of the title page or this project sheet and all remaining items electronically via MR (a PDF file of all printouts and actual cpp file).

1. Title page with name, class, project number, and relevant information about your program (compiler and system used, file names).
2. Notes about your program (status of your program at the minimum, but you should discuss issues/bugs regarding your program).
3. Pseudocode.
4. Printouts of any input/output (must include provided test cases at the minimum).
5. A printout of the source code.
6. A copy of your source code on a flash drive or CD or MR -- source code (.cpp) for this program.

Your program will be graded as follow:

- Correctness: 25 points
- Test Cases: 5 points
- Pseudocode: 5 points
- Documentation/Coding Style: 5 points
