CSCE 1030 – Homework 4

Due: 11:59 PM on Wednesday, October 22, 2014 CST

Problem Statement:

For this assignment you will edit your homework 3 to include functions. Unless so indicated, all requirements of homework 3 hold in homework 4. **Note that there are some changes to the assignment!**

You will take your solution from homework 3 or the solution posted on BB, and will edit it as follows:

1. You will write a function to display your personal information (department and course number, program number, your name, your EUID, and your e-mail address).
2. You will write a function to print out the menu.
3. You will write a function that, given the number of objects bought and the price per object, will return the total cost for these objects. **Do not use a for-loop in this function. There must be a single function for all objects!**
4. You will write a function that will determine if a sale should receive a 10% discount, and will return the correct sales price.
5. You will write a function to compute the tax. The tax rate should be passed as a parameter to the function!

Your functions should receive as arguments only the minimally needed ones of the appropriate type, and should return a value of the appropriate type (wherever a return value is needed).

Design:

On a piece of paper, write down in English the sequence of steps you will perform to solve the problem. Pretend this is a “recipe” for someone else to follow. Refine your “recipe” until it is clear. Be sure to include the steps for prompting for input, etc. This document should contain the sequence of steps used.

Type these steps into a document (Word, txt, PDF, etc.). Also be sure to include your algorithm steps as comments in your code file. Do this before you start coding as completing it afterwards does not help you in learning the design process!

Implementation:

Now that you have a working design, your next step is to translate these steps into C code. Use the algorithm development techniques discussed in class to implement your solution to the problem above. Add your C code a little at a time, and compile and test as you go.

Remember to add your comments to your code to explain your program. Do this before/during programming instead of waiting until the end. At a minimum, you should comment the program header (e.g., name, class, date, brief description of the program, etc.), all variables (i.e., what they are used for), function headers (see below), and specific "blocks" of code. For example, use comments to describe the inputs, the formulas used, and any other important steps in your code.

Your program will be graded based largely upon whether it works correctly on a CSE Department machine, so you should make sure your program compiles and runs on a CSE machine.

Your program will also be graded based upon your programming style. At the very least, your program should include:

- A consistent indentation style as recommended in the textbook and in class;
- Meaningful variable names;
- A program header comment section that includes: your name, e-mail address, and a brief description of the program.
- **Function headers as described below.**

Documentation:

When you have completed your C program, write a short report (2–3 paragraphs) describing:

1. what the objectives were,
2. what you did to solve the problem,
3. the status of the program. Does it work properly for all test cases? Are there any known problems?
4. **You will also include an example of your program working. To do so you will need to study the Linux command "script." Do not do a screen capture.**
5. **Some calculations or verification to show that your examples of your program working are correct. In other words, show that your program calculates sales prices correctly, tax, discount, etc., by comparing the output of your program to a few (a few!) hand calculations.**

Save this report in a separate file to be submitted electronically. You should also include any specific instructions required to compile or execute your code, such as linking a specific library (e.g., "`-lm`" for the math library).

Testing:

Test your program to check that it operates as desired with a variety of inputs, including any boundary conditions. Compare the answers your code gives with the ones you get from hand-calculations.

Homework Submission:

CSCE 1030 – Homework 4

Due: 11:59 PM on Wednesday, October 22, 2014 CST

In this class, we will be using electronic homework submission to make sure that all students hand their programming projects (and labs) on time. You will submit your program source file to the class website through the "**Homework 4**" drop box by the due date and time.

Note that this project must be done individually. The program will be checked using a code plagiarism tool against other solutions, so please ensure that all work submitted is your own.

Note that the dates on your electronic submission will be used to verify that you met the due date above. All homework up to 24 hours late will receive a 50% grade penalty. Later submissions will receive zero credit, so hand in your best effort on the due date.

Summary:

- You will design a solution to the problem.
- You will implement it on the CSE machines using C. You will make sure to use good style, good variable names, indentation, etc. You will compile, run, and test your code.
- You will write a brief report describing what your code does and how well it works.
- You will submit electronically your C code, your design, and your brief report.

General Guidelines (for ALL of your programming assignments):

- Your program's output should initially display the department and course number, program number, your name, your EUID, and your e-mail address.
- Use meaningful variable names.
- Use appropriate indentation.
- Use comments, including a program header. Example program header:

```
/*
Author: Jane Doe (Jane.Doe@my.unt.edu)
Date: 9/18/2014
Purpose: This program reads in three numbers and computes their
average
*/
```

- Add a header to each function. Example function header:

```
/*
Function: deposit
Parameters: a double representing the account balance,
and a double representing the deposit amount
Return: a double, the account balance after the deposit
Description: This function computes the account balance after a
deposit
*/
```