CSCE 2004 – Programming Project 2
Midpoint Due Date – Feb 8, 2017 at 11:59pm
Final Due Date – Feb 15, 2017 at 11:59pm

1. Problem Statement:

The goal of this programming assignment is to give students experience creating an interactive program that uses nested conditional statements.

In particular, your task is to write a program that simulates an online restaurant, where users can choose between the "breakfast menu" the "lunch menu" or the "dinner menu" and then choose food items to purchase from one of these menus. To meet these goals, your program should do the following:

- Print a message asking the user which menu they wish to order from. You should also include instructions on what to enter to select a menu. For example, one option is to use integers 1,2,3. Another option is to use single characters B,L,D. Finally, you could have users type out the words breakfast,lunch,dinner. It is up to you to choose what the user should type.

- Read the user's input from above, and use this information to decide what list of food menu items to print. To keep things short and simple, your three menus should only include five items each. For each menu item, you should print the name of the item and the price of the item. You should also print instructions on what the user must type to select an item. For example, you could use integers 1,2,3,4,5 or single characters, or single words. Again, this is up to you.

- Read the user's input of the menu item they wish to buy, and ask them how many orders of that item they want to buy. Read the item quantity from the user, and calculate and print the total cost for this purchase using the formula "total_cost = item_price * item_quantity + sales tax". To keep things simple, lets pretend the sales tax rate is 5% of the purchase price.

- Finally, prompt the user to confirm their order with a message "Do you wish to order this food now? (yes/no)". If the user enters "yes" then print a thank you for your order message. If they enter "no" print a different message saying sorry we could not help you.

2. Design:

This project is all about printing messages, reading user inputs, and performing different operations based on user inputs. Your first design decision is to select the messages you want to print out to the user to explain the different options available at each stage of the program. Your next design decision is to decide how to read in

the user's input and use this information to take different branches in the program. For example, what variables to use to store the user's input, how to compare these to the list of expected input values, and what to do if someone types in an unexpected input. Your final design task is to work out the formula needed to calculate the total cost of the user's order.

3. Implementation:

Since you are starting with a "blank piece of paper" to implement this project, it is very important to develop your code incrementally writing comments, adding code, compiling, debugging, a little bit at a time. This way, you always have a program that "does something" even if it is not complete.

As a first step, start with an empty main function, and add the code to print the initial messages to the user, read the user's input, and then print the input values back out again. Once this part is working, you can start printing more messages, and read more user inputs to select menu items.

4. Testing:

Test your program to check that it operates correctly for all of the requirements listed above. To do this, you should check every branch through the code (the three menu choices at the start, and the five item choices for each menu). You should also verify that your total cost formula works with a range of quantity values.

You are NOT required to add error checking in this program, but it is always good to test a program to see what happens if the user inputs unexpected data. For example, what happens to your program if the user enters an invalid menu choice or a silly quantity? You should cut/paste these results into your project report to document what your program does in these cases.

5. Documentation:

When you have completed your C++ program, write a short report using the "Programming Project Report Template" describing what the objectives were, what you did, and the status of the program. Does it work properly for all test cases? Are there any known problems? Save this project report in a separate document (in docx format or pdf format) to be uploaded into blackboard with your code..

6. Midpoint Project Submission:

To encourage students to get an early start on their programming project, students are required to upload into Blackboard a partial solution to their programming project on the midpoint due date shown above. The program does not need to be complete, but it must compile and perform some of the tasks listed above. This midpoint solution is worth 10% of your final grade on the project.

7. Final Project Submission:

In this class, we will be using Blackboard for all project submissions to make sure that all students hand their programming projects on time, and to perform automatic plagiarism analysis of all programs that are submitted.

When you have completed all of the tasks listed above, go to Blackboard to upload your documentation (a single docx or pdf file), and your C++ program (a single cpp or txt file). Do NOT upload an executable version of your program.

The dates on your electronic submission will be used to verify that you met the due date above. All late projects will receive reduced credit:

10% off if less than 1 day late,
20% off if less than 2 days late,
30% off if less than 3 days late,
no credit if more than 3 days late.

You will receive partial credit for all programs that compile even if they do not meet all program requirements, so handing projects in on time is highly recommended.

8. Academic Honesty Statement:

Students are expected to submit their own work on all programming projects, unless group projects have been explicitly assigned. Students are NOT allowed to distribute code to each other, or copy code from another individual or website. Students ARE allowed to use any materials on the class website, or in the textbook, or ask the instructor and/or GTAs for assistance.

This course will be using highly effective program comparison software to calculate the similarity of all programs to each other, and to homework assignments from previous semesters. Please do not be tempted to plagiarize from another student.

Violations of the policies above will be reported to the Provost's office and may result in a ZERO on the programming project, an F in the class, or suspension from the university, depending on the severity of the violation and any history of prior violations.