CSCI-C 200

**Lab 4 – Conditional Executions**

**Description**

Design a program that will perform tasks based on at least two user selections or entries, e.g. user will select from a list of choices presented, and enter another input to generate outputs.

Sample ideas:

1. Length conversion. User will be asked to enter a number in centimeter, then will ask the user to choose what unit of measurement to convert into, and the choice can be:
2. meter
3. millimeter
4. inch
5. foot
6. …

In this example, the first entry is a value in centimeter, and the second entry is a value to indicate choice.

1. Draw shapes. User will be asked to select what shape to draw, e.g.
2. Square
3. Rectangle
4. Triangle
5. …

In this example, after the first entry (to indicate a choice), user will be asked to enter more data in order to complete the task, for instance, if user selected 1, user needs to enter a length value, and if user selected 2, user needs to enter a length value and a width value. You decide how to determine the length of the shape based on user inputs, e.g. use \*\*\* to represent 1.

1. Calculate gas price based on what type of gas and how many gallons user has pumped into his/her car.
2. Simulate a coffee vending machine. User will first be asked to select a coffee type, then select cup size, strength, etc. Your program will output the final price based on user entries.

Use your imagination, be creative and have fun!

**Rubric**

* 1. Ask for at least two keyboard inputs(10’)
  2. If/else structure is used. (10’)
  3. Each choice is defined as a function and the functions are called in the executable file. (10’)
  4. Have a utility file and group all the function definitions in this file. (5’)
  5. Take invalid inputs into considerations. E.g. if an input should be numeric, only process the input if it is indeed numeric. (5’)
  6. Program runs successfully and generates meaningful outputs. (30’)
  7. Follow professional coding standards (file headers, comments, code blocks, naming conventions, etc). (10’)
  8. Use pseudocodes to describe your algorithm design in a .docx file. Your code must follow the algorithm design. Your pseudocodes must be more similar to the Python language rather than the English language. See sample pseudocode file named “keyboardInputPseudocode.txt”
     1. Pseudocodes are used (5’)
     2. Algorithm design is detailed enough. Must also have algorithm design for each function. (5’)
     3. Algorithm design is accurate and complete. (5’)
     4. Program follows the algorithm design. (5’)

**Submission**

Submit your program files, and an algorithm design file on Canvas.