CptS 121 - Program Design and Development

Lab 6: Wonderful World of Iterative Statements in C Continued

Assigned: Week of September 30, 2024
Due: At the end of the lab session

I. Learner Objectives:

At the conclusion of this programming assignment, participants should be able to:

- Compose iterative statements (“while”, “for”, and/or “do-while” statements)
- Confidently apply loops to C programs
- Utilize nested loops

II. Prerequisites:

Before starting this programming assignment, participants should be able to:

- Create and utilize compound conditions
- Compose basic iterative statements
- Apply top-down design principles
- Programmatically open, process, and close files

III. Overview & Requirements:

This lab, along with your TA, will help you navigate through applying iterative statements in C. Once again we will take a modular approach to designing solutions to the problem below. As part of the lab you will need to decide which C selection structure and iterative structure is best suited for a particular function. You will have the option to use "if", "switch", "while", "do-while", and/or "for" statements for the below problem.

Labs are held in a “closed” environment such that you may ask your TA questions. Please use your TAs knowledge to your advantage. You are required to move at the pace set forth by your TA. Have a great time! Labs are a vital part to your education in CptS 121 so work diligently.

Tasks:

1. a. With your team, write a program that reads in each of the integer values in a file and determines if the sum of the integers is prime. Use functions where appropriate!

   b. Determine if the sum of the individual digits, in the sum of the integers, is prime. For example, 524 = 5 + 2 + 4 = 11, which is prime. Another example is 3201 = 3 + 2 + 0 + 1 = 6, which is not prime. Use functions where appropriate!

2. With your team, first, draw a flowchart on the whiteboard for the prompt in this question. Second, write a program that determines the factorial of `n`, represented n!, where `n` is entered by the user. Make sure that your program checks to see if `n` is greater than or equal to 0. As long as `n` is negative your program should continue to prompt for another value for `n`. Recall 0! is equal to 1 and 1! is also equal to 1. The factorial of a number such as n! is equal to n * (n – 1) * (n – 2) *… * 1. Use functions where appropriate!

3. Write a program that determines the Fibonacci number for the n^{th} term. Recall, the first Fibonacci number is 0 and the second Fibonacci number is 1. The next number in the sequence is always determined by the sum of the previous two terms or numbers in the sequence. Fib (n) = Fib (n-1) + Fib (n-2). The n value should be entered by the user. Use functions where appropriate!

4. Write a program that generates a random number between -100 to 100 and requires that the user guesses which number was generated. If the user guesses too high, then the program should say so. If the user guesses too low, then the program should say so again. The program should continue to loop until the user guesses the correct number. If the user guesses a number outside the range of -100 to 100, then the program should just prompt the user to re-enter a number without indicating if the guess was too high or too low. Keep track of the total number of guesses taken by the user. Use nested loops to solve this problem. Use functions where appropriate!

IV. Submitting Labs:

- You are not required to submit your lab solutions. However, you should keep them in a folder that you may continue to access throughout the semester.

V. Grading Guidelines:

- This lab is worth 10 points. Your lab grade is assigned based on completeness and effort. To receive full credit for the lab you must show up on time, work with a team, complete 2/3 of the problems, and continue to work on the problems until the TA has dismissed you.