Inf-1100, Assignment 1

Submission: 23:59 September 16th, 2022

Overview

The focus of this assignment is to write and document a small C program. The C program will consist of several functions. You will learn to use the tools necessary to write and compile a program, and to practice the skills of documenting programs you create.

Description

Your assignment is to write several small functions and make use of these in a C program.

1. Write a C function that compares two integers, and prints the largest value of these integers to the console. For example, if the function is called with the value 5 assigned to *x* and the value 10 assigned to *y*, the function should print the value of *y* to the screen.

   The C function should have the following signature:
   ```
   void compare_values(int x, int y);
   ```

   To solve this problem, you need to read up on how to compare values using *if* statements.

2. Write a C function that prints a triangle. The number of lines in the triangle should be a parameter to the function. For example, if the function is called with the value 4 as an argument, the following triangle should be printed:
   ```
   *
   **
   ***
   ****
   ```

   The C function should have the following signature:
   ```
   void myTriangles(int numlines);
   ```

   To solve this problem, you need to read up on how to express loops and how to create nested loops in C.

3. Write a C function that determines whether a number have a given prime factor. For example, if the function is called with the value 10 as *number* and 5 as *primeFactor*, the function should return 1 indicating that 5 is a prime factor of 10. Similarly, the function should return 0 if the *number* in the example was 9.

   The C Function should have the following signature:
   ```
   int myPrimeFactor(int number, int primeFactor);
   ```

   To solve this problem, you need to read up on use of the modulus operator in C.

4. Write a C function that prints numbers in a given range, along with an indication of whether the number is odd or even and whether 5 is a prime factor in the numbers. You must use the *myPrimeFactor* function to determine whether 5 is a prime factor of the number. The function should print something like:

   1 is odd and 5 is not a prime factor
   2 is even and 5 is not a prime factor
   ...
   10 is even and 5 is a prime factor

   The C function should have the following signature, where *startnum* indicates what number to start printing on, and *endnum* indicates where to stop:
   ```
   void myNumbers(int startnum, int endnum);
   ```

   To solve this problem, you need to read up on use of the modulus operator in C.
5. The binary logarithm of *n* (written as: log2*n*) is the power to which the number 2 must be raised to obtain the value *n*. Write a C function that calculates log2 of an unsigned integer *n* using the following approach: find the most significant set bit in *n* and return the position of this bit. For example, if *n* is 17 (0b10001), the function should return 4.

   The C function should have the following signature:
   ```
   int myLog2(unsigned int n);
   ```

   To solve this problem, you need to read up on how to express logical and bitwise shift operations in C.

6. Write a C function that reverses a string. In C, a string is represented as an array of characters. The functions should manipulate the character array in such a way that it's content is reversed after the function has completed. If the function was called with an array containing the characters "A string", after the completion of the function the array should contain "gnirts A".

   The C function should have the following signature:
   ```
   void reverseString(char string[]);
   ```

   To solve this problem, you need to read up on how strings are represented in C, and how to manipulate arrays.

C solution

Create a file 'functions.c' with the C source code of your program. To test your implementation of the assignment, call the functions you have written from the main function with various inputs. Compile the code with the C compiler generating a program 'functions', and run this program to see that you get the expected result.

Extra credit

Create a menu prompting the user to select which of the functions to execute. Allow the user to specify the inputs relevant to each function, and display the results (hint: look at the documentation for the functions f*scanf* and/or *getchar*).

Report

Describe the implementation of the functions you made. Do not include the source code in the text (it should be submitted as separate files). However, the source code should be readable enough, both in structure and the way it is commented, for another programmer to verify its correctness. In other words, your report together with the source code needs to convince us that your program work, not execution of the actual program!

Include a ‘discussion section’ in which you note your observations on this assignment. Maybe you observed something interesting (or just surprising to you). If so, write about it in your report.

Also report any particular assistance you got from course staff or your fellow students that has been crucial to the fulfillment of the assignment. You are strongly advised to write documentation/notes along the way.

Submission

You should submit the report (‘assignment1.pdf’) and your code (‘functions.c’) no later than September 16 23:59. The submission should be done using Canvas.

You should submit two files, the report and the C source code. We only accept the report as a single file in the PDF format (not scanned). The C source code should be packed in a compressed archive, such as ‘zip’, ‘rar’ or ‘tar.gz’.