Program 2 - Pointers and Bits

CS 580U - Fall 2017

Due Date: 5:00 p.m., September 26, 2017

All programs will be tested on the machines in the Q22 lab. If your code does not run on the system in this lab, it is considered non-functioning EVEN IF IT RUNS ON YOUR PERSONAL COMPUTER. You can write your code anywhere, but always check that your code runs on the lab machines before submitting.

Driver Code and Test Input Files

- **Driver Code**
  - program2.c //Driver Code
    - *Include the driver code with your submission, but do not alter it.*

Grading Rubric

TOTAL: 20 points

- **Part 1: (12 points)**
  - myStrStr function works with all test cases (12 points / 2 for each case)
- **Part 2 (7 points):**
  - Passes Counting 1's test (2 points) (must pass all tests)
  - Passes Binary Tests (5 points total / 1 point each)
- **Style Guidelines (1 point)**
  - Follows requested program structure and submission format
  - Follows formatting guidelines

Guidelines

This is an individual assignment. You must do the vast majority of the work on your own. It is permissible to consult with classmates to ask general questions about the assignment, to help discover and fix specific bugs, and to talk about high level approaches in general terms. It is not permissible to give or receive answers or solution details from fellow students.

You may research online for additional resources; however, you may not use code that was written specifically *to* solve the problem you have been given, and you may not have anyone else help you write the code or solve the problem. You may use code snippets found online, providing that they are appropriately and clearly cited, within your submitted code.

By submitting this assignment, you agree that you have followed the above guidelines regarding collaboration and research.

Part 1 -- String Manipulations

In the second program you will create a library file containing 4 functions. I have provided driver code that runs and tests your library code. You will need to create a second file that can be included in the driver code so the compiler can link them into an executable.

To include a file in c, you should use the preprocessing statement `#include` and the name of the file as a string at the top of your code. I have included a file called “mylib.h” in the driver code provided. This means you must write all of your code in a file called “mylib.h” and place it in the same directory as the provided driver code for the program to compile.

- For this part of the program you will implement your own version of the library function strstr(), with the following function interface:
  *int myStrStr (char * haystack, char * needle, char * buffer);*
  - Your function will take 3 strings, a 'haystack' string, a 'needle' string, and a buffer string. You will search the haystack string for a sequence matching the needle string, and copy the found result (the entire substring) from the haystack string into the buffer (do not copy the needle string).
  - You will return a 1 if the matching sequence in the haystack is found and a 0 if the needle is not found.
  - *You can not use the library strstr() function in your implementation. It is only for reference.*
- I will test your function with the following hardcoded strings. **Do not ask for user input.**
  - haystack="chocolate", needle="choc"
  - haystack="vanilla", needle="lla"
  - haystack="caramel", needle="am"
  - haystack="strawberry", needle="strawberry"
  - haystack="banana", needle="na"
  - haystack="cherry", needle="terrible"

Part 2 - Bits and Bytes

In Part 2 you are going to work with binary in a couple different ways.

- First you will write a function that counts the number of 1’s in the binary representation of an integer passed as a parameter to a function called countOnes() (see driver code for exact interface). You should return the number of 1s in the function’s parameter as an unsigned int.

- Next you will write a looping binary interpreter that creates an array representation of a binary value.
    - Write a function that uses bit shifting to store the binary representation of an integer in an array passed as a parameter. You will be given the size of the array and should make sure you fill out all elements in the array. To generate the array of values, you will need to use a bit mask and bitwise right shift.
        - 8 would store the following value in your array
            - 0000 0000 0000 0000 0000 0000 0000 1000
        - The provided driver code (main) which calls your binaryArray function with each of the below values and an array buffer to print the binary representation (from right to left) for each one:
        - 2
        - 255
        - -1
        - INT_MAX
        - INT_MIN
            - The driver code includes the library <limits.h> at the top of the main source code file so you can use the global constant INT_MAX and INT_MIN

    - You can use the following website to check your results: http://www.binaryhexconverter.com/binary-to-decimal-converter

Part 3 - Submission

You must use any driver code as I have given you. DO NOT change it to take user input, different output, etc.

- Required code organization:
    - **<user_id>**_program2.c //Driver Code
    - mylib.h
        - Any additional files should be included in your project2.zip submission
- While inside your program2 folder, create a zip archive with the following command
    - zip -r **<user_id>_program2** <user_id>_program2.c mylib.h
        - This creates an archive of all file and folders in the current directory called **<user_id>_program2.zip**
        - **Do not zip the folder itself, only the files required for the program**
- Upload the archive to Blackboard under Program 2.