CS1050 – Lab 8

Spring 2020

Concepts to Practice

- Characters
- Strings
- Standard Library functions for characters
- Expand the prelab

Submission Information

Submit this assignment by following the instructions given by your TA. SUBMIT ONLY the .c file (no a.out or executable file is required). All of the lab assignments must be submitted before the end of the lab using the lab code given by the TA.

Use the following submit command:

mucs submit <class> <assignment_type> <filename> <filename2>

For example:

mucs submit 1050 lab x-lab8main.c x-lab8stringfuncs.c

Filenames must be: *sectionletter-lab8main.c* (include your respective lab section, e.g., a-lab8main.c) and *sectionletter-lab8funcs.c* (include your respective lab section, e.g., a-lab8funcs.c).

Description

For the lab assignment, you will implement a program that takes pre-defined test string and prints out all of the words in that string in order, one-by-one. You may only use the header files <stdio.h> and <ctype.h>. You may not use any additional header files, nor any functions (other than those functions you write yourself) that are not defined in these two header files.

Your x-lab8main.c file should look like this:

```
// Prototypes
void ShowWords(char * string);

int main(void)
{
    char testString[] =
"Once there was a way to get back homeward.\n\
Once there was a way to get back home .   \n\
Sleep pretty darling.\n\
Do not cry, and I will sing this lullabye.";

    ShowWords(testString);
}
```

You may cut and paste that into your own x-lab8main.c if you like, or you can download this file using this command:

wget https://tinyurl.com/1050sp20lab8 -O x-lab8main.c

Notice that the “O” must be capitalized. You can change x-lab8main.c to your lab letter (e.g., y-lab8main.c).

Functions You Must Write

You may write any functions you wish to implement this program, in **addition** to the following functions. However, you **must** implement the following functions, and they must be prototyped as shown:

- **void ShowWords(char * string)** – This function takes a string and simply prints each word in the string on a new line with a number (count) corresponding to each word.

Notice that the ShowWords() function should be implemented in x-lab8funcs.c, not in x-lab8main.c.

Tips

- Print a lot more than you have to while developing. You may want to print the string, the character you are currently looking at to decide whether you have reached the beginning or end of a word, etc. You may end up taking out some of these printf() calls later, but it will help you to debug your program if you have a lot of extra printf() info.
- Break things into smaller parts. I said you have to have ShowWords(), but that function might call several other functions if you want to break things down. Maybe you need a function to find the beginning of the next work? Maybe you need a function to find the end of the current word? What other ways might you break things down?
- If my test string is too long or hard, consider starting with a shorter string. Just comment out my string, and add a short one of your own like “Once there” or “Jim Ries”. If you can make things work with that much, maybe you can make it longer (like “Once there was” or “Once there,,was” to make sure you deal with extra spaces and punctuation) until it works on anything.
- Notice that my test string is “multi-line”. It has a backslash at the end of each line, which is C’s way of saying that the string continues on the next line. I could have put the whole string on one line and it would work just the same, but it would be harder to read.
- Notice that some parts of the test string have multiple spaces, newlines, and punctuation. This is intentional, and all of it can be used to delimit words.

Sample Output

```
jimr@JimRArea51:~/CS1050/SP2020/lab8$ compile x-lab8main.c x-lab8funcs.c
jimr@JimRArea51:~/CS1050/SP2020/lab8$ ./a.out
Word 1- Once
Word 2- there
Word 3- was
Word 4- a
Word 5- way
Word 6- to
Word 7- get
Word 8- back
Word 9- homeward
Word 10- Once
Word 11- there
Word 12- was
Word 13- a
Word 14- way
Word 15- to
Word 16- get
Word 17- back
Word 18- home
Word 19- Sleep
Word 20- pretty
Word 21- darling
Word 22- Do
Word 23- not
Word 24- cry
Word 25- and
Word 26- I
Word 27- will
Word 28- sing
Word 29- this
Word 30- lullabye
```

Guidelines for Grading Lab 8
40 Points Possible

General

If your program does not compile or produce any input/output (I/O) because most of the source code is commented out then your lab will receive a grade of ZERO POINTS. Further, if your program does not actually follow the specifications, but merely prints out lines that make it appear to follow the specifications, you will receive a grade of ZERO POINTS. For partial credit your C program must not only compile but also produce some valid I/O that meets the lab specifications.

You program is expected to have a comment header at the top that includes your name, pawprint, the course you are taking, and the lab that you are solved (e.g., "Lab 8"). Your code should be nicely indented. **You will lose up to 10 points if you do not meet these basic requirements**.

5 points: Your program uses 2 files (a "main" and another file for functions you write). The "main" file has no function other than main() implemented in it.
5 points: The original test string is identical to the one I provided and is declared **only** in main().
5 points: ShowWords() (or the functions it calls) does not make a copy of the original string, but modifies it in place if needed (modification may not be necessary).
20 points: Each word is printed on a line by itself with the correct word number and no additional stray characters (i.e., there is no extra space or punctuation after each word).
5 points: Output essentially matches the sample.