CSCD 240

For this lab you will write a simple string parser that will allow the user to write each string to a file using low level commands. I have provided an unchangeable tester.

The program flow is as follows. When the program starts, the user is required to enter a string. You are guaranteed the user will enter at least one letter. The user can ultimately enter up to 650 characters with words being delimitated by a period comma or space.

Once the first string is entered, the user is prompted to select from a menu of choices. The choices are:

1) Enter a new string - allows the user to replace the current string
2) Parse the string and then write that string, word by word, to an output file using low level commands
    - Write the length of the word and then the word to the output file
    - Close the file descriptor when complete
3) Read an existing output file and print the string to the screen, and also displays the length of the maximum word.
4) Quit

Helpful Stuff (Technical Term)

- The string remains the same until the user selects #1 – this includes tokenizing it
- Since the string will contain spaces, you must use fgets.
- Since you are entering numbers don't forget to strip the carriage return after you enter a number.
- You must use strcpy and strtok and strlen.
- Since you are using fgets, the carriage return will be a part of the string, you must deal with this.
- You will create a basic makefile – target lab15
- Of course check menu ranges
- If the user chooses #3 and the file does not exist, display an error message and redisplay the menu

To Turn In:

A zip that only contains:

- cscd240Lab15.c and all other C/H files to make your code work
- makefile
- an output run named cscd240Lab15Out.txt
- a valgrind run named cscd240Lab15Val.txt

You better know the name of the zip file by now. (Example: steinerslab15.zip)