CSCD 340
Homework 1

One of the most important data structures that will be used is a Linked List. It is important that we develop a very generic linked list that we can use during the course of the quarter.

Linked List Specifics
- Singly Linked List
- void pointer for data
- next pointer
- Must be declared in a header file named linkedList.h
- The choice of using a size is completely up to you

Word Structure
- char * pointer for word
- int for the length of the word

Specific Functions
- openFile – prompts the user for the name of the input file. The function attempts to open the input file. If it does not then the user is re-prompted. When the file is opened the FILE pointer is returned – Located in utility.h & utility.c

- processFile – takes the FILE pointer and the linked list as a parameters. It reads each word from the file and adds the word as part of a word structure to the linked list in the order read in. The memory for each Node will be dynamically allocated and the memory for the name will be dynamically allocated. This function will run until the end of file is reached. Don't forget to close the file at the end of the function – Located in utility.h & utility.c

- printList – if the list is empty prints "Empty List" otherwise prints the list one per line in this format word:length – Located in linkedList.h & linkedList.c

- menu – displays the following menu 1) Add a Word 2) Delete a Word 3) Clear the List 4) Find a Word 5) Print the List 6) Sort the List by word (ascending order) 7) Sort the list by the lengths of the word primary, alphabetical secondary 8) Quit – This function will ensure the number is within range – Located in utility.h & utility.c

- doChoice – execute the corresponding functions that go with the menu choices – Located in utility.h & utility.c

- clearList – clears all the words and nodes in the list. Ensure you don't leak memory for the word or the node – Located in linkedList.h & linkedList.c

Other Things to Be Done

- You must create linkedlist.c and linkedlist.h – these files will only deal with list related stuff – these will be very general files – ie the word structure will not be in linkedlist.h
- You must create utility.c and utility.h – these files will only deal with non linked list stuff like the menu and doChoice.
- You may create other .h and .c files as you see fit; however, you will be graded on design.
- You will create an output file that shows the run of the program using valgrind. This output file should show that no memory is being leaked.
- You must create a make file named makefile, which the grader will use to compile your code.

TO TURN IN

- The source code of your program. Don't forget to include the C file that contains main. Your grader will compile your code.
- Your makefile
- Your valgrind run named cscd340_s14_hw1val.txt
- At least one run of your program, captured as output, that illustrates your code is working properly. This does not need to be extensive. We will test your code this is just to illustrate you did your own testing. Name this cd340_s14_hw1output.txt.

You will submit a zip file named your last name first letter of your first name hw1.zip. (Example steinershw1.zip)

If you don't include all the files including the input file(s) you will receive a 0. Give us everything required to compile, run and test your code.