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.

Node Specifics

- void pointer for data
- next pointer
- Declared in the header file named linkedList.h

LinkedList Specifics

- Node pointer for head (no dummy head node)
- int size
- Declared in the header file named linkedList.h

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

- countRecords – receives the file pointer and number that represents the number of lines to make a record. This function counts the number of lines in the file. If the count is 0 prints an error message and exits, otherwise returns the count divided by the number of lines that comprise a record.

- buildList – takes the linked list, the FILE pointer, and the total records in the file 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 – Located in linkedList.h & linkedList.c

- menu – displays the following menu and ensures the number entered is within range – Located in utility.h & utility.c
    - 1) Print the List
    - 2) Add First
    - 3) Add Last
    - 4) Sort the List (ascending order)
    - 5) Delete a Word
    - 6) Quit –

- 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 & utility.c
- I have provided linkedlist.h and utility.h – You CANNOT change these files in any fashion, other than to #include your own .h files.
- 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.
- I have provided cscd340_hw1.c – You can't change anything in this file.

TO TURN IN

- The source code of your program. Your grader will compile your code.
- Your makefile
- Your valgrind run named cscd340_hw1val.txt
- At least one run of your program, captured as output, that illustrates your code is working properly. Name this cd340_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.