CS 372 Advanced Programming and Algorithms
Programming Assignment #1 Rubric

(12% of your cumulative grade)

Rating **Rating Category**	**Exemplary**	**Proficient**	**Basic (needs work)**	**Not Demonstrated**
Documentation	Documentation clearly explains what code does. Includes complete and accurate **file** and **function** headers, as detailed in the CS372 coding standards, along with additional in-line comments at appropriate locations.	Documentation includes all required headers, but may lack clarity or details in some instances, OR may have violated some minor details of the CS372 coding standards, OR may be completely overcommented.	Documentation is incomplete and/or incorrect and/or formatted incorrectly. May have violated significant details of the CS372 coding standards or the underlying program intent.	Only a few (or no) comments in program.
Data Storage	Constants used for ALL fixed values. Correct data types for all variables and DLL list structure, and all were defined within the correct program scope. Followed CS372 coding standards naming conventions and used descriptive names for all identifiers.	A few minor errors in data declaration scope, data typing or assignment, OR missing one or two necessary constants, OR may have violated some minor details of the CS372 coding standards.	Did not correctly define array of pointers, or included one or more other major errors in declaration scope, data typing, or assignment AND/OR violated of significant details of the CS372 coding standards.	Constants not used, identifier names not descriptive enough, multiple data structures defined incorrectly, and/or did not follow any of the CS372 coding standards.
File Usage	Passes filename in via command line argument. Checks if argument exists before trying to use it. Checks if file could be opened (i.e. it exists) before trying to read from it. Program exits gracefully if file does not exist or could not be opened. If exists, file closed immediately after use.	Passes filename in via command line argument, but only one check (exists/opened) performed, OR checks performed but program still did not exit gracefully when error occurred OR file not closed immediately after use.	Passes filename in via command line argument, but neither check performed, OR filename not passed (hardcoded or read from user) then file existence checks performed.	Filename not passed (hardcoded or read from user) AND no checks (exists/opened) performed.
Program Input (Reading and Processing)	Reads and parses words from input file correctly. Ignores words containing digits. Correctly counts words read from file. Strips off external punctuation, but not interior punctuation. Converts words to all upper or all lowercase. Recognizes end of file correctly, whether the file contains a newline on the last line or not.	Parses into words correctly, but doesn't strip off all external punctuation, OR strips off some internal punctuation, OR doesn't convert words to all one case OR count is off by one due to incorrect reading of file or counting words with digits OR does not always recognize end of file correctly.	Has problems parsing into individual words, or problems completing more than one of the other required tasks.	Fails to read any file input.

Program Processing: Linked List Creation	Correctly implements an array of doubly linked lists and initializes all lists correctly to NULL. Verifies memory allocation before using it. Inserts each word into the top of the correct list. Checks if word already in list (if so, does not insert). There are no memory leaks.	Minor problem with list implementation, OR initialization (inserts dummy nodes), OR allocation verification, OR checking for duplicate words, OR word insertion at top of list.	Major problem with list implementation, OR initialization, OR allocation verification, OR checking for duplicate words. OR inserted words at the bottom of each list OR minor problems with more than one of the above items.	Fails completely. Word lists not created at all or all lists created incorrectly.
Program Processing: Linked List Usage	Uses forward links to traverse each letter list and correctly counts words. Uses backward links to traverse each letter list and displays correct words.	Traversal word counts or displays have minor errors.	Uses forward links for both counting words and displaying words OR traversal word counts/displays have major errors.	Word lists not traversed correctly in either direction.
Display Output	Displays correct word count and word list for non-empty lists, and does not display any output for letters with empty lists. No comma displayed after last word in each list.	Works for typical input, but may fail when testing one of the special cases, OR does not satisfy one of the requirements.	Fails for typical input, for minor reason(s), OR fails multiple special cases, OR does not satisfy multiple requirements.	Fails for typical input, for a significant reason, OR produces no display output.
Final Program Results	Displays correct unique word count for file. Correctly determines the highest count for letter usage. Displays letter(s) with highest count (ties included). Method used to determine high counts and display results is clear and efficient.	Displays correct total and unique word count for file. Displays one of the letters with highest count, but not ties. Method used may not be the most efficient.	Fails to display correct word counts for file, AND/OR fails to determine the correct highest letter count for letter usage, AND/OR fails to display even one correct highest count letter.	Fails completely – all results are incorrect.
Modularity (functional breakdown)	Program is modular in design and all modules are logically organized with prototyping. Each module performs ONE well-defined task and is defined and called correctly. **main** function minimal – does little more than call other functions. Program uses code efficiently.	Program is modular in design and is mostly logically organized. Most of program uses code efficiently. May be missing one required function, contain a module that performs too many tasks, or have one function that is incorrectly defined or called.	Program is only partially modular in design, OR multiple modules perform too many tasks, OR parts of the program are not logically organized, OR code is inefficient, OR more than one required function missing, OR multiple problems with module definitions/calls.	Program is not modular in design or is not logically organized AND/OR program has little efficient use of code and program compactness AND/OR program modules cannot be easily modified/debugged.

C++ Constructs / Readability / Miscellaneous	Demonstrates understanding of program, control, and file structures. Appropriate use of language with correct parameter passing and no global variable usage. Code is exceptionally well organized and easy to follow, and adheres to all CS372 coding standards for code usage (e.g. no use of **break** to exit a loop).	A parameter or two was passed incorrectly (value vs. reference), OR an unnecessary parameter(s) was passed, OR there was minor improper control or data structure usage, AND/OR there may be some occasional spacing, indentation, and/or other minor organizational issues.	One or more major error(s) or coding standard violations, to include any incorrect variable usage, and/or many problems with parameter definition and usage, AND/OR there may be substantial spacing, indentation, and/or other organizational issues.	Demonstrates minimal understanding of control/data structures and/or proper parameter passing AND/OR contains major coding standard violations, AND/OR code is poorly organized and very difficult to read AND/OR has other major construct/readability issues.
Analysis of Code Efficiency	Correctly analyzes the computational efficiencies of each function. Correctly describes data that would produce the best, average, and worst case scenarios. Correctly determines the big-O efficiency for the entire program.	A few minor errors, but the the big-O efficiency for the entire program is logical using the function efficiencies given.	One or more major error.	Analysis not submitted.