**Submission of Your Work**

You need to prepare and submit ONE SINGLE MS Word document to Canvas (in your lab section) as LastName\_FirstName\_Labxy.doc. It must contain:

* Your NAME only on page 1
* For ***each*** question:
  + Specify the question number.
  + Source code. Copy/Paste your final source code. You must include standard “comment header” even if code is provided. *Do Not* paste a snippet of your source code, it must be copy/pasted.
  + Initial test plan. After reading the question requirements, but **before** beginning any coding, create the test case table, below, completed through column Expected Output. Include in your report.
  + Final test plan. Write your program then complete the **test table** with actual output results and include in your report *AFTER* your source code.
  + Output results. Paste in a snippet of output showing results for **every listed test case in your final test plan**, labeled with test case #

Test Table:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test # | Valid / Invalid Data | Description of test | Input Value | Expected Output | Actual Output | Test Pass / Fail |
| 1 |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |

* Add / delete rows from Test Table as necessary
* Modify column widths as necessary
* Test both valid and invalid input
* Test for every output expected
* If failure is an expected output and it happens then that test Passes
* Any test that fails means the program must be fixed so that it passes the test
  + Failing tests need a new test row, ie 1a, 1b, etc, showing corrections from original

**Program:**

User will provide 2 words; the characters in each word will be loaded into both (1) an array and (2) a sorted doubly-linked (contains both next and previous pointers) list (two arrays and two doubly-linked lists are used, one array/linked list for each word).

Create a struct containing char variable “letter” and int variable “occurrences”. Use this structure in the array. For the linked list, use the same struct layout plus pointers to the next and previous elements of the list.

Letters can be used more than once in the word but appear only once in the array and list; update occurrences field with count of how many times that letter occurs. Once the arrays and lists are created (1) merge the two arrays into a third array and (2) use overloaded operator += to combine the lists together into one list.

Show the contents of all arrays and lists after the words have been read in, and again after they are merged.