# **Homework II: Linked Lists**

Due Date: February 17, 2021.

**Points: 60 points**: 2 bonus points if submitted by February 14

Submit **source code files** and **sample text I/O** and capture of I/O console covering all possible scenarios into word document.

Please make sure that all screen captures are large enough font to see. And the text is in black with white background. Small or hard to see screen captures are not accepted.

**Homework Rubric.**

**3 = Good     2 = Average     1 = Below Average     0 = Not gradable**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Criteria** | **Comments** | **Points** |  |  |  |
| **1.**The program fully implements the solution to the given problem |  | 15 | 10 | 5 | 0 |
| **1.**The solution produces the correct result |  | 14 | 9 | 5 | 0 |
| **2.**The program runs without error, has reasonable error checking and exception handling |  | 6 | 4 | 2 | 0 |
| **3.**The user is prompted for input when required |  | 4 | 3 | 1 | 0 |
| **4.**The user is presented with appropriate feedback as to the results of an operation |  | 4 | 4 | 1 | 0 |
| **5.**All project files and source code required to build and execute are submitted |  | 5 | 3 | 1 | 0 |
| **6.**Code is properly indented, and adequately commented |  | 3 | 2 | 1 | 0 |
| **7.**Code is well written and free from unnecessary complexity or redundancy |  | 3 | 2 | 1 | 0 |
| **8.**Programmer defined symbol names are reasonable (variable names, class names, function names, etc...) |  | 3 | 2 | 1 | 0 |
| **9.**The user interface is accurate and is free from misspelled words and bad grammar |  | 3 | 2 | 1 | 0 |
| **Total Points out of 30:** |  |  |  |  |  |
| **Comments:** |  |  |  |  |  |

**The assignment:**

Generate 30 Random numbers between 1 and 58

Store the numbers in an array

Array entries are not sorted and can have duplicates.

Read the numbers one at a time and store them in a link list while adhering to the following conditions:

* As you copy the content of the randomly generated numbers from the array to the linked list, be sure to eliminate redundant numbers in the linked list and
* Maintain the linked list numbers in ascending sorted order as you add more numbers to the linked list.
* Track the redundant numbers separately by storing all duplicates in an array to calculate how many times each number was repeated.

**Output**: print the following to both console and textile:

* The content of the originally generated array.
* The content of the LinkedList.
* The content of the array storing the duplicates and how often each number is being duplicated.
* A method to find and display the length of the LinkedList. The number of elements stored in the linked list.
* All output must have proper text headings and formatting when printed to both a file and the screen.
* Make sure that all your functions are specialized and kept in a separate file (e.g., .h file) from the file with the main method. **Each function** must perform only one task:
* **Overall code quality and performing proper functionalities.**

Example: Code one function to generate the random numbers and store them into an array

Another function to add to the linked list.

Another function to print the sorted array to file

And so on.

Random numbers: Array1: 15, 2, 8, 2, 2, …… : not in order and may have duplicates

Linked List 2, 8, 15 ….

Array2 of duplicates: 2, 2 : not in order and consists of all instances except the first instance of the number.

Output to file and screen:

Array1

Linked list and how many elements in the linked list.

Array2