ENGR 2930 Program 13 Requirements – Fall 2017

Introduction: Some mathematical computations are extremely complex and time-consuming for a computer to perform. Often, we resort to a look-up table instead, where outputs are pre-calculated and stored in a file. Mathematical functions that fall into this category include trigonometric functions (sine, cosine, tangent, etc.), roots, and logarithms.

Task: You are to create a program that reads natural numbers (i.e. integers with value zero or greater) from an input text file and writes a table of integers and corresponding base 10 logarithms to an output text file.

Requirements:

1. The program will be done using Cygwin and gcc.

2. The input file will contain one integer per line. The output file shall contain one integer / logarithm pair per line, with the two values separated by a tab. The logarithms shall be printed as floating-point values with 6 digits after the decimal point.

3. All integers in the input file will be natural numbers between 0 and 99. There is no limit to the size of the input file. Numbers in the input file will not be ordered.

4. The output file must be sorted in ascending order (smallest to biggest).

5. Duplicates in the input file must be ignored (only one corresponding entry in the output file).

6. The program shall be named LogTable.exe and source files shall be called LogTable.c, myLinkedList.h, and myLinkedList.c.

7. The program will expect 2 command-line arguments to specify the names of the input and output files. Example command-line input:

   ```
   $ .\LogTable.exe input.txt output.txt
   ```

8. Malformed command-line syntax and/or failure to open input or output files shall result in an error message and program termination.

9. Integers shall be stored in a sorted linked list. You must use dynamic memory allocation.

10. Dynamically allocated memory must be freed before program termination.

11. Remember function headers and pseudocode!

Example Files:

If the input file appeared like this:

```
66
79
63
10
53
10
66
43
70
44
```

The associated output file should look like this:

```
10   1.000000
43   1.633468
44   1.643453
53   1.724276
63   1.799341
66   1.819544
70   1.845098
79   1.897627
```

Notice that the output is in ascending order and duplicates have been removed. Think about different input file scenarios and craft appropriate input files to test those scenarios.