ECE 1410 Final Project Requirements

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 insertion and deletion instructions concerning 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 graded using Cygwin and g++.

2. The input file will contain one operator / operand pair per line, with the two values separated by white space. 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 4 digits after the decimal point.

3. Valid operators are "a" (add) and "d" (delete).

4. Valid operands in the input file will be natural numbers between 0 and 99. There is no limit to the size of the input file. Entries in the input file will not be ordered.

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

6. Duplicate adds in the input file must be ignored (only one corresponding entry in the output file). Non-existent removes should also be ignored.

7. Include a makefile that builds your complete solution.

8. 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
   ```

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

10. Integers shall be stored in a sorted binary tree. You must use dynamic memory allocation and deallocation. Your binary tree class should support insertion of nodes, deletion of nodes, and ordered writing of the entire tree to a text file.

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

12. Submit a single zip file that contains all your source code and your Makefile.

Example Files:

If the input file appeared like this:

```
a 66
a 79
d 84
a 63
a 10
a 53
a 10
a 66
a 43
d 66
a 70
a 44
d 66
d 79
d 63
d 10
```

The associated output file should look like this:

```
43      1.6335
44      1.6435
53      1.7243
70      1.8451
```

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