University of Massachusetts Dartmouth
Department of Electrical and Computer Engineering

ECE 160 name: `modefile.cpp`
Prj 10 – Find modes of data from file Due: see `http://ece160.org`

Write a program which will read from the file "numbers.txt". The file will contain one or more data sets (each line of the file contains exactly one value). A data set is defined by an integer indicating the number of values in the data set. Following the dataset will either be a 0 (zero) indicating no more data sets or a value containing the number of values in the next set. A dataset consists of at most 100 value.

After a dataset is read, the program should then print out the original values and the values of the dataset sorted in ascending order SIDE BY SIDE, with the columns properly labeled (see sample run).

Lastly, the mode (or count) of each value should be output. See the sample run below for examples.

To add a textfile to a project:
Project/Add New Item/Utility/Textfile/type name, and click "Add"
This will put the file in the proper place, so that just a: fopen("filename.txt", "rt") with no path specified will find the file. Specifying a file path will result in loss of points.

File: numbers.txt

```
14                  ← number of values in first dataset
50
60
50
60
10
10
70
80
50
50
50
10
50
6
4                   ← number of values in next dataset
25
25
25
25
8                   ← number of values in next dataset
1
3
3
3
3
3
3
3
5                    ← number of values in next dataset
3
3
3
3
3
0                    ← no more datasets
```

Output should be:

```
  Original    Sorted
        50         6
        60        10
        50        10
        60        10
        10        50
        10        50
        70        50
        80        50
        50        50
        50        50
        50        60
        10        60
        50        70
         6        80

     Value  Count
         6 -   1
        10 -   3
        50 -   6
        60 -   2
        70 -   1
        80 -   1


  Original    Sorted
        25        25
        25        25
        25        25
        25        25

     Value  Count
        25 -   4


  Original    Sorted
         1         1
         3         3
         3         3
         3         3
         3         3
         3         3
         3         3
         3         3

     Value  Count
         1 -   1
         3 -   7


  Original    Sorted
         3         3
         3         3
         3         3
         3         3
         5         5

     Value  Count
         3 -   4
         5 -   1
```

Hints

To print out (to a file) the original and sorted side by side, it is necessary to create another array; for example you might have an orig[] and sort[] array. Copy each element of the orig[] array to the sort[] array. You need to copy the elements one at a time in a loop.

To determine a count for each of the values, you need to do some processing on the sorted array. Pseudo code follows:

```
current_count = 1
current_val = sort[0]
for (i=1; i< number_of_values; i++)
     if ( sort[i] != current_val )
          output( current_val, current_count )
          current_val = sort[i]
          current_count = 1
     else
          current_count++
output( current_val, current_count )
```

Obviously, you need to have a sorted array. If you have not yet written/tested a bubble sort routine, now would be a good time.

Again, it is strongly suggest that after you write each function, you thoroughly test it before integrating all the functions.