CSC 112 – Vectors and Merging

Purpose and Goals

By the end of the lab, you will have:

- Written 2 programs in C++, making use of the fundamental aspects of the C++ language, with a focus on the vector class and its associated operations and following the desired style guidelines
- Declared instances of the vector class and made use of its fundamental operations of size(), resize(), at(), and push_back().
- Gained additional experience in using while loops
- Handled non-trivial vector indexing problems
- For two of the programs, you will extend code you have previously written.

Reading and Background

zyBook and lectures as appropriate.

Part 1

In this part of the lab, you will continue to develop your ability to work with vectors and build confidence in keeping indices straight. In this part, you may not use arrays, only vectors. Also, when working with loops, you must use while or do while loops for inputting the data. This is to get you used to the fact that there are alternate ways to do things and choosing the more effective is important.

In particular, you will have to merge two vectors alphabetically. Merging is an important concept that you will see data structures, probably as part of sorting.

Name your vectors as: que1 and que2. The input for this problem will be a series of lines with a name on each line. The names will be stored in que1 and que2. The names for que1 will be followed by the names for que2. The text ENDQ will indicate the end of the input for each vector. Do not use this as one of the names.

The program should work if either or both vectors are empty. You can set the initial size for que1 and que2 to 100 and can safely assume there will never be more than 100 names input for either vector. Names input for each vector will be in alphabetical order on input (you do not have to sort them).

After inputting the names, resize que1 and que2 (using the resize() method) to exactly match the number of names in each (they may not have needed all 100 spots). Print the name and size (using the size() method) of each vector and the contents of each.

Now, merge que1 and que2 into another appropriately sized vector named que_merge such that the names in que_merge are in alphabetical order. This will require you to iterate down both vectors, keeping track of which index you are on in each vector. Keep duplicates, merging them together as appropriate. Following the merge, que_merge should only be large enough to hold the names and no larger. Print the name "que_merge", its size, and its contents.

Note: Do not combine the queues then sort. You may not use sorting. More on reasons for this when we discuss algorithm complexity.

For guidance, input and output should be as shown in the example input and output shown on the next page. More examples can be seen via the tests in the zyLab for this assignment.

Example input

```
Dog
Pete
Rachel
Yumyum
Zorro
ENDQ
Aardvark
Cat
Simon
Zippy
ENDQ
```

Example output

```
Enter queues:
que1: 5
Dog
Pete
Rachel
Yumyum
Zorro

que2: 4
Aardvark
Cat
Simon
Zippy

que_merge: 9
Aardvark
Cat
Dog
Pete
Rachel
Simon
Yumyum
Zippy
Zorro
```

Part 2

Part 2 of the lab is the same as Part 1 with the one addition. You will need two new vectors, que_merge_no_dup and que_count. Using the values in que_merge, put the names into a new vector que_merge_no_dup but removing duplicates. Then, for each value in que_merge_no_dup, put a count of how many times the name appeared in que_merge into vector que_count. The final size of que_merge_no_dup and que_count should be just large enough to hold the data and no larger. Print everything as in part one followed by the two new queues. Example input and output is on the following page and in the zyLab tests.

Example input

```
Pete
Rach
Rachel
Simply
Simply
Yummy
ENDQ
Ciacia
```

Dog
Dog
Pete
Zorro
ENDQ

Example output

Enter queues:

que1: 6
Pete
Rach
Rachel
Simply
Simply
Yummy

que2: 5
Ciacia
Dog
Dog
Pete
Zorro

que_merge: 11
Ciacia
Dog
Dog
Pete
Pete
Rach
Rachel
Simply
Simply
Yummy
Zorro
que_no_dups: 8
Ciacia 1
Dog 2
Pete 2
Rach 1
Rachel 1
Simply 2
Yummy 1
Zorro 1