ECE 71/191T – Data Structures and Algorithms

Dr. Gregory R. Kriehn, Fresno State
C++ Homework Assignment: Chapter 21

Code Due By: Midnight on Sun, May 14
Write-up Due By: Office Inbox on Mon, May 15, 10:00 AM

HOMEWORK #39 – STL Implementation of Vectors/Lists
EXTRA CREDIT HOMEWORK POINTS: 200

Implement the following nested menu programs using STL versions of **`vector`** and **`list`**. Options on the main menu give access to corresponding sub-menus. For this assignment, use the following Main Menu:

```
        Welcome to the ECE 191T STL Vector/List Menu
    ====================================================
      1. Test the STL vector with integers
      2. Test the STL vector with doubles
      3. Test the STL list with integers
      4. Test the STL list with characters

      0. Exit
    ====================================================
```

STL Vector Menu
When Option 1 or 2 is selected, the STL Vector Sub-Menu will be printed and the user can select any item off the sub-menu. When the user exits the STL Vector Sub-Menu, they will return back to the Main Menu. Do not exit the program until the user selects Option 0 in the Main Menu.

```
                        STL Vector Sub-Menu
**********************************************************
  1. Print the capacity and size of the vector
  2. Clear all the elements of the vector
  3. Add an element to the end of the vector
  4. Remove the last element from the vector an print its value
  5. Use the index operator to set and print the value of an
       element
  6. Use the reserve function to change the number of elements
       allocated
  7. Use the resize function to change the number of elements in
       use
  8. Search for a given value in the vector using find()
  9. Print the current contents of the vector using the []
       operator

  0. Return to the Main Menu
**********************************************************
```

Create a vector object using the appropriate data type based upon the selection from the main menu and use the vector interface to perform these actions. Create iterators when appropriate to perform necessary operations (such as for Option 8).

STL List Menu
When Option 3 or 4 is selected, the STL List Sub-Menu will be printed and the user can select any item off the sub-menu. When the user exits the STL List Menu, they will return back to the Main Menu. Do not exit the program until the user selects Option 0 in the Main Menu.

```
Linked List Sub-Menu
++++++++++++++++++++++++++++++++++++++++++++++++++++
  1. Insert a value at the front of the list
  2. Insert a value at the back of the list
  3. Insert a value at a given position in the list
  4. Search the list for a value
  5. Delete all instances of a value

  0. Return to the Main Menu
++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Create a list object using the appropriate data type based upon the Main Menu selection and use the list interface to perform these actions. Use iterators for the position and search results.

HW Problem Reference & Credit: Dr. Melissa Danforth, CSU Bakersfield

The results from your program should be:

```
Enter input filename: data1.txt

Depth First Traversal:  0 1 4 3 2 5 7 8 6 9
Breadth First Traversal: 0 1 3 4 2 5 7 8 6 9
```

HOMEWORK #40 – STL Implementation of Stacks/Queues
EXTRA CREDIT HOMEWORK POINTS: 100

Redo the homework from Chapter 17, which corresponds to Homework #33. Remove all references to your template classes, and solve the problem using STL's **`stack`** and **`queue`** template classes instead. The homework problem should be contained entirely in `main.cpp`.

HOMEWORK #41 – Sorting Time for a Vector
EXTRA CREDIT HOMEWORK POINTS: 50

Modify the homework from Chapter 18, which corresponds to Homework #34. First, fix your

code if it was not working previously. Then create **`list6`**, which is a 10,000 element **`vector`**. Copy the unordered elements from **`list1`** into the vector, and compare the time necessary to sort the vector compared to the other algorithms you coded. Print each of the execution times for the 5+1 algorithms to the screen for your 10,000 element unsorted arrays/linked list/vector.

```
Bubble Sorting Time:     3.573e-01 s
Selection Sorting Time:  1.580e-01 s
Insertion Sorting Time:  9.478e-02 s
Quick Sorting Time:      1.568e-03 s
Merge Sorting Time:      1.743e-03 s
Vector Sorting Time:     X.XXXXXXX s
```

HOMEWORK #42 – STL Algorithms
EXTRA CREDIT HOMEWORK POINTS: 50

Given two files named "`cities1.txt`" and "`cities2.txt`", consisting of the following lines:

`cities1.txt`

```
Santa Barbara
Fresno
Berkeley
Bakersfield
Tulare
Santa Cruz
Irvine
Davis
Merced
Redding
```

`cities2.txt`

```
Berkeley
Los Angeles
San Diego
Davis
Fresno
Lost Angeles
Berkeley
Irvine
Madera
San Francisco
```

Write a program using STL containers and iterators to produce a merged, sorted list of cities with all duplicates removed. Use **`list`** as the container, and use STL algorithms to merge, sort, and remove the duplicates from the list. Send the result to standard output using the **`output_iterator`**.