CS260 Lab 4

Instructions

For your own amusement (and for your grade in this class), you have decided to keep a "Database Of Great Computer Scientists." You will store this database in a **binary search tree**, implemented as an array according to the description in the section entitled **An array-based representation of a complete tree** on page 516 of Carrano, 5th edition (page 518 of Carrano, 4th edition). In this scheme, the left child of the node at index i will be at index 2 * i + 1, and its right child will be at node 2 * i + 2. (Do not implement the representation that includes the definition of TreeNode. The representation you want is the one after that one. You *do not need*, and *should not use*, the TreeNode definition that starts on page 515.)

Note that while the textbook says this representation is for a *complete* binary tree, it can actually be used for a binary tree of *any* shape. The only problem is that if the tree is not complete, there will be space wasted in the array. But the wasted space is not a problem for this small application.

Your array should start with a size of 5, and grow as needed. It is not acceptable for your array to "run out of room."

For each computer scientist, you will store their name as a standard C string (i.e. as char *).

This assignment comes in two versions: ***Grade B*** *(which is easier) and* ***Grade A*** *(which is more difficult). If you want to get an A in this course, you will need to do the Grade A version. If, instead, you will be satisfied with a B, you can do the Grade B version and save yourself some work.*

Grade B Version

Implement all of the operations defined for the BST class *except for* the remove operation.

Grade A Version

Do all of the work needed for the Grade B version, and also implement the remove operation that is already defined in the BST class. You are required to do this by implementing the algorithm described in Carrano. In particular, when deleting a node with two children, *you are required to use the inorder successor*. The algorithm can also be implemented using the inorder predecessor, but you should not do this because then your output will then not be identical to mine and your grade will suffer accordingly.

Hints for the Grade A Version:

- *draw pictures of the trees*
  There are a number of different possibilities you will have to handle, and there are different things you will have to do for each of these cases. Drawing pictures of the trees and of how they change will help you keep track of all of this. (It helped me!)
- *in this array-based implementation of the tree, each node has its own predefined location in the array*
  Therefore, you cannot just "update links" like you can with a pointer-based implementation. You will have to move things around in the array. Depending on how your code works, you may need to traverse subtrees *breadth first* to do this. "Breadth first" means that, starting from the top, each level is traversed from left to right before the next level is traversed. Remember that some of the slots in the array will be *empty*.

Program Output

Here is the printed output that your code should produce once it's fully implemented (but with your own name, and with the version you have done accurately identified). You will need to use I/O manipulators (defined in <iomanip>) with cout in order to format the printout properly. Your code should produce this printout *exactly*, both in formatting and in the order in which items are listed. Note that this is the output from the Grade A version. If you do the Grade B version, you can just let all of the calls to BST::remove continue to fail.

```
CS260 - Lab4 - Grade A - Michael Trigoboff

Database Of Great Computer Scientists

>>> array order:

name                    leaf?  index
----                    -----  -----
Ralston, Anthony                   0
Liang, Li                          1
Von Neumann, John                  2
Jones, Doug                        3
Trigoboff, Michael                 5
Goble, Colin            leaf       7
Knuth, Donald                      8
Turing, Alan            leaf      12
Kay, Alan               leaf      17
(items printed)                      (9)

>>> preorder:

name                    leaf?  index
----                    -----  -----
Ralston, Anthony                   0
Liang, Li                          1
Jones, Doug                        3
Goble, Colin            leaf       7
Knuth, Donald                      8
Kay, Alan               leaf      17
Von Neumann, John                  2
Trigoboff, Michael                 5
Turing, Alan            leaf      12
(items printed)                      (9)

>>> inorder:

name                    leaf?  index
----                    -----  -----
Goble, Colin            leaf       7
Jones, Doug                        3
Kay, Alan               leaf      17
Knuth, Donald                      8
Liang, Li                          1
Ralston, Anthony                   0
Trigoboff, Michael                 5
Turing, Alan            leaf      12
Von Neumann, John                  2
(items printed)                      (9)

>>> postorder:

name                    leaf?  index
----                    -----  -----
Goble, Colin            leaf       7
```

```
Kay, Alan               leaf      17
Knuth, Donald                      8
Jones, Doug                        3
Liang, Li                          1
Turing, Alan            leaf      12
Trigoboff, Michael                 5
Von Neumann, John                  2
Ralston, Anthony                   0
(items printed)                      (9)

>>> retrieve Trigoboff, Michael

Trigoboff, Michael

>>> retrieve Norman, Donald

not found

>>> remove Ralston, Anthony

Ralston, Anthony removed

>>> array order:

name                    leaf?  index
----                    -----  -----
Trigoboff, Michael                 0
Liang, Li                          1
Von Neumann, John                  2
Jones, Doug                        3
Turing, Alan            leaf       5
Goble, Colin            leaf       7
Knuth, Donald                      8
Kay, Alan               leaf      17
(items printed)                      (8)

>>> preorder:

name                    leaf?  index
----                    -----  -----
Trigoboff, Michael                 0
Liang, Li                          1
Jones, Doug                        3
Goble, Colin            leaf       7
Knuth, Donald                      8
Kay, Alan               leaf      17
Von Neumann, John                  2
Turing, Alan            leaf       5
(items printed)                      (8)

>>> inorder:

name                    leaf?  index
----                    -----  -----
Goble, Colin            leaf       7
Jones, Doug                        3
Kay, Alan               leaf      17
Knuth, Donald                      8
Liang, Li                          1
Trigoboff, Michael                 0
Turing, Alan            leaf       5
Von Neumann, John                  2
(items printed)                      (8)

>>> postorder:
```

```
name                    leaf?  index
----                    -----  -----
Goble, Colin            leaf       7
Kay, Alan               leaf      17
Knuth, Donald                      8
Jones, Doug                        3
Liang, Li                          1
Turing, Alan            leaf       5
Von Neumann, John                  2
Trigoboff, Michael                 0
(items printed)                      (8)

>>> remove Jones, Doug

Jones, Doug removed

>>> array order:

name                    leaf?  index
----                    -----  -----
Trigoboff, Michael                 0
Liang, Li                          1
Von Neumann, John                  2
Kay, Alan                          3
Turing, Alan            leaf       5
Goble, Colin            leaf       7
Knuth, Donald           leaf       8
(items printed)                      (7)

>>> preorder:

name                    leaf?  index
----                    -----  -----
Trigoboff, Michael                 0
Liang, Li                          1
Kay, Alan                          3
Goble, Colin            leaf       7
Knuth, Donald           leaf       8
Von Neumann, John                  2
Turing, Alan            leaf       5
(items printed)                      (7)

>>> inorder:

name                    leaf?  index
----                    -----  -----
Goble, Colin            leaf       7
Kay, Alan                          3
Knuth, Donald           leaf       8
Liang, Li                          1
Trigoboff, Michael                 0
Turing, Alan            leaf       5
Von Neumann, John                  2
(items printed)                      (7)

>>> postorder:

name                    leaf?  index
----                    -----  -----
Goble, Colin            leaf       7
Knuth, Donald           leaf       8
Kay, Alan                          3
Liang, Li                          1
Turing, Alan            leaf       5
```

```
Von Neumann, John                 2
Trigoboff, Michael                0
(items printed)                     (7)

>>> remove Kay, Alan

Kay, Alan removed

>>> array order:

name                    leaf?  index
----                    -----  -----
Trigoboff, Michael                 0
Liang, Li                          1
Von Neumann, John                  2
Knuth, Donald                      3
Turing, Alan            leaf       5
Goble, Colin            leaf       7
(items printed)                     (6)

>>> preorder:

name                    leaf?  index
----                    -----  -----
Trigoboff, Michael                 0
Liang, Li                          1
Knuth, Donald                      3
Goble, Colin            leaf       7
Von Neumann, John                  2
Turing, Alan            leaf       5
(items printed)                     (6)

>>> inorder:

name                    leaf?  index
----                    -----  -----
Goble, Colin            leaf       7
Knuth, Donald                      3
Liang, Li                          1
Trigoboff, Michael                 0
Turing, Alan            leaf       5
Von Neumann, John                  2
(items printed)                     (6)

>>> postorder:

name                    leaf?  index
----                    -----  -----
Goble, Colin            leaf       7
Knuth, Donald                      3
Liang, Li                          1
Turing, Alan            leaf       5
Von Neumann, John                  2
Trigoboff, Michael                 0
(items printed)                     (6)

>>> remove Kay, Alan

Kay, Alan not found

>>> remove Von Neumann, John

Von Neumann, John removed

>>> array order:
```

```
name                    leaf?  index
----                    -----  -----
Trigoboff, Michael                 0
Liang, Li                          1
Turing, Alan            leaf       2
Knuth, Donald                      3
Goble, Colin            leaf       7
(items printed)                      (5)

>>> preorder:

name                    leaf?  index
----                    -----  -----
Trigoboff, Michael                 0
Liang, Li                          1
Knuth, Donald                      3
Goble, Colin            leaf       7
Turing, Alan            leaf       2
(items printed)                      (5)

>>> inorder:

name                    leaf?  index
----                    -----  -----
Goble, Colin            leaf       7
Knuth, Donald                      3
Liang, Li                          1
Trigoboff, Michael                 0
Turing, Alan            leaf       2
(items printed)                      (5)

>>> postorder:

name                    leaf?  index
----                    -----  -----
Goble, Colin            leaf       7
Knuth, Donald                      3
Liang, Li                          1
Turing, Alan            leaf       2
Trigoboff, Michael                 0
(items printed)                      (5)

>>> remove Turing, Alan

Turing, Alan removed

>>> array order:

name                    leaf?  index
----                    -----  -----
Trigoboff, Michael                 0
Liang, Li                          1
Knuth, Donald                      3
Goble, Colin            leaf       7
(items printed)                      (4)

>>> preorder:

name                    leaf?  index
----                    -----  -----
Trigoboff, Michael                 0
Liang, Li                          1
Knuth, Donald                      3
Goble, Colin            leaf       7
```

```
(items printed)                    (4)

>>> inorder:

name                    leaf?  index
----                    -----  -----
Goble, Colin            leaf       7
Knuth, Donald                      3
Liang, Li                          1
Trigoboff, Michael                 0
(items printed)                    (4)

>>> postorder:

name                    leaf?  index
----                    -----  -----
Goble, Colin            leaf       7
Knuth, Donald                      3
Liang, Li                          1
Trigoboff, Michael                 0
(items printed)                    (4)

>>> remove Goble, Colin

Goble, Colin removed

>>> array order:

name                    leaf?  index
----                    -----  -----
Trigoboff, Michael                 0
Liang, Li                          1
Knuth, Donald           leaf       3
(items printed)                    (3)

>>> preorder:

name                    leaf?  index
----                    -----  -----
Trigoboff, Michael                 0
Liang, Li                          1
Knuth, Donald           leaf       3
(items printed)                    (3)

>>> inorder:

name                    leaf?  index
----                    -----  -----
Knuth, Donald           leaf       3
Liang, Li                          1
Trigoboff, Michael                 0
(items printed)                    (3)

>>> postorder:

name                    leaf?  index
----                    -----  -----
Knuth, Donald           leaf       3
Liang, Li                          1
Trigoboff, Michael                 0
(items printed)                    (3)
```

Code To Start With

The code for you to start with is provided as a Microsoft Visual C++ 2008 Express project. To get this project, download and expand **lab4.zip**.

Add code to the definition and implementation files as needed. You can add public and private members, but *do not remove any of the public members that are already there*. Also, *make no changes* to **lab4driver.cpp** other than to have it print your name instead of "Your Name," and to show whether you have done the Grade B version or the Grade A version (in the same line where you typed in your name).

There may be additional instructions in comments in the code files, which may contradict the instructions above. In that case, do what those instructions say you should do.

I encourage you to be creative in designing your algorithms and code. I *do not* encourage "creativity" in the area of file names, project names, solution names, etc. Please *do not* change these things within the project that has been provided to you.

Printed Output

Your code is required to produce printed output *exactly as shown above*. The output shown above is indented to the right in this description for clarity. Do not indent your actual output. Start each line of your output in the leftmost column of the console window.

Memory Leaks

Your code is required to run in Microsoft Visual C++ 2008 Express without memory leaks. The starter code for this project includes support for memory leak detection in VC++ 2008.

Grading Your Code

I will be testing your code in Microsoft Visual C++ 2008 Express. Your code needs to compile without errors or warnings and run correctly in VC++ 2008. Code that does not compile in VC++ 2008 will receive zero points. Code that crashes in VC++ 2008 will receive zero points.

To Submit This Assignment

First, "clean" your project by doing the following:

- execute **Clean Solution** in the **Build** menu
- close Visual Studio, and then delete the **.ncb** file from your solution's root folder

After cleaning your project, compress it into **lab4.zip**. Make sure you have your Zip application set to *preserve the folder structure* of your project, and make sure that your .zip file *will extract into a single folder*, the way that the .zip files from this course do.

Doing these things will ensure that your .zip file is as small as possible, which will make both your upload and my download quicker.

Submit the .zip file containing your cleaned project to Blackboard. Upload into the assignments section of Blackboard the zip file for your cleaned project. Please be sure the source file prints your name, assignment description and number, as requested.

- Be certain to check that you completed the upload successfully. After you click the Upload File button, you must also click the **SUBMIT ASSIGNMENT** button. This is very easy to forget. You have been warned. I would recommend entering an email address so you can be notified that the uploaded was

completed successfully.

- You may upload as many versions as you wish prior to the due date. I will only see and grade the last one. You won't be able to upload assignments after the due date.