The Assignment:

Implement the Priority Queue class, using a heap to store the items, as described in Section 11.1.

Purposes:

Ensure that you understand and can use a heap which is implemented in an array.

Before Starting:

Read all of Sections 10.2 and 11.1.

Files that you must write:

1. pqueue2.h:
2. Header file for this version of the PriorityQueue class. You don't have to write any of this file. Just copy our version and add your name and other information at the top.
3. pqueue2.cxx:
4. The implementation file for the new PriorityQueue class. This version contains stubs for you to complete.

Other files that you may find helpful:

1. pqtest.cxx
2. A simple interactive test program.
3. pqexam2.cxx
4. A non-interactive test program that will be used to grade the correctness of your Priority Queue class.

The PriorityQueue Class Using a Heap

Discussion of the Assignment

Your PriorityQueue class for this assignment will use a heap that is stored in an array. The component type of the array is called OneItemInfo, and is defined as part of the PriorityQueue class in pqueue2.h. In order for you to further learn how to use private member functions, I would like for you to write and use all the private member functions

that are listed in pqueue2.h. The precondition/postcondition contracts for these functions are given in the queue.cxx file.

Dynamic Memory (Not used in this assignment)

This version of the priority queue does not use dynamic memory. What does that tell you about the need for a copy constructor, an assignment operator, and a destructor?

Do the Assignment in Two Parts

For easier debugging, carry out this assignment in two parts. Part One should include everything except the public get_front ( ) member function and the private functions that are needed for get_front (i.e., is_leaf, big_child_index, and big_child_priority). While you are debugging Part One, you should add a new option to the interactive test program. The new option makes this call to print the current state of the tree of the test PriorityQueue: test.print_tree ("STATE OF THE TREE:"). (**Note that print_tree is a public member function that I wrote for you. You should remove it before you submit your final work.**)

Once Part One is working and tested, you may complete the assignment by implementing the get_front( ) member function and its associated private functions.

Use the interactive test program and the debugger to track down errors in your implementation. If you have an error, *do not start making changes until you have identified the cause of the error.* If you come to me for help, I will always ask you to do the following:

1. Show us the invariant that describes how your private member variables implement the List class.
2. Use the debugger to show me the problem.