Programming Assignment #6

Binary Search Tree and Heap implementations

CS 3358.253, Spring 2015
Instructor: Jill Seaman

Due: Wednesday, 4/29/2015 (upload electronic copy by 1:30pm)

Part 1:

You will implement the BST_3358 interface, which is a templated binary search tree.

bst_3358.h the interface and incomplete implementation (on the website)

In the file bst_3358.h, you will find 10 undefined (or incompletely defined) functions. You will need to provide the implementation for these functions.

Note that there are 10 public functions. There are 8 private functions. Many of the public functions call a private function to do their work. The private functions take pointers to TreeNodes as parameters so that they can be called recursively (the public functions cannot take TreeNodes as arguments, because TreeNodes are private). The public and private functions are overloaded (have the same names, different param lists). You should not use loops to implement the functions, instead use recursion.

You will implement the following functions:

- constructor
- destructor
- makeEmpty (private version)
- findItem (private version)
- findMin (private version)
- findMax (private version)
- insertItem (private version)
- countNodes (private version)
- inOrderTraversal (private version)
- preOrderTraversal (private version) (note the special requirements for this one)

Part 2:

You will implement the Heap_3358 interface, which is a templated binary heap.

heap_3358.h the interface and incomplete implementation (on the website)

In the file heap_3358.h, you will find 7 undefined (or incompletely defined) functions. You will need to provide the implementation for these functions.

Some of the functions contain comments to help you write the code. Some of the functions should call other functions (as indicated in the comments). As with part 1, you should not use loops to implement the functions, instead use recursion.

NOTES:

- Implement one function at a time, compile, test, and fix. Write your own test drivers.
- For the BST, I recommend implementing the constructor, insertItem, and then findItem and inOrderTraversal, so that you can test your insert. Everything depends on getting insert implemented correctly.
- For the heap, implement the constructor, insertItem and percolateUp. After that you can use the provided display function to see if the tree was constructed properly.
- Most of the definitions will be short, one or two recursive calls with a base case or two. Don't try to make it more complicated than it needs to be.

Style: See the Style Guidelines document on the course website.

Logistics:

Please submit the following files in a single zip file. You can call it assign6_xxxxxx.zip, where xxxxx is your TX State NetID (your txstate.edu email id).

```
bst_3358.h heap_3358.h
```

Submit: an electronic copy only, using the Assignments tool on the TRACS website for this class.