

Due: Thursday, July 28

Emphasis: Binary Trees and Searching

In this assignment you will add inorder, preorder and postorder traversals to a binary search tree class. All three implementations of accessing the binary tree will be implemented in a similar manner, simply the order of output of a node and recursively visiting the two child nodes will differ. You should implement the actual traversal as a recursive function that visits the nodes of the tree. You have been given an implementation of a binary search tree, in BST.h and BST.cpp, as well as a test driver for the class. There are places in the test driver commented out for you which indicates the correct prototype of the function you should create for your traversals. For example, to invoke an inorder traversal, the call looks something like:

```
intBST.inorder(cout);
```

That is to say, the public function to do an inorder traversal takes a single parameter, an ostream variable, and displays the contents of the tree to the indicated output stream. You will need to define a different, private method, to do the actual recursion and traversal of the nodes.

In addition to implementing the inorder, preorder and postorder traversals, you should also fill in the destructor for the BST. The implementation of the destructor was left blank. Basically this was because the destructor amounts to doing an inorder traversal of the binary tree, but instead of displaying the items as they are visited inorder, you need to delete them. Using your implementation of inorder traversal, correctly create a destructor to do an inorder traversal of the nodes, deleting the nodes as they are visited.

Requirements

- 1) Use the BST.h and BST.cpp classes provides as your starting point
- 2) Add inorder, postorder and preorder methods to the class, prototypes should match those as required in the BSTtester.cpp test driver.
- 3) Implement your traversal methods using recursion. You will need to define a private method to perform the actual recursive traversal which will be called by the public interface method.
- 4) Also implement the destructor for the BST as an inorder traversal that causes the nodes in the tree to be deleted during the inorder traversal.