Due Tuesday, April 19

Preorder Binary Search Tree

You will read numbers from a file to construct a binary search tree with them. Then you will output a representation of this BST to an output file using preorder traversal.

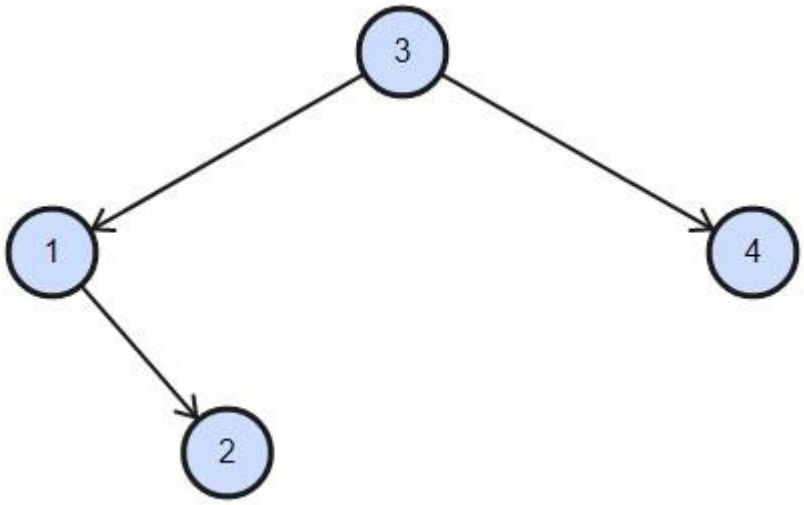
The input file will consist of a single line of numbers separated by spaces. You may assume that there will always be less than 100 numbers and they will all be unique. Your C++ program must read these numbers in order and add them to a binary search tree with the first number becoming the root node. You must implement a node and/or a binary tree class yourself. Implement these however you like but do not use advanced standard library features such as maps. Remember that there is no need to implement node deletion functionality you only need to implement add and traversal functions. For the output, the program will create a file with one line per node in preorder including the node's location in the tree.

For example, if the input file is:
3 1 2 4

Your program will:

- Read 3, making it the root node
- Read 1, go to the left child of 3, since the left child is empty insert 1 as left child of 3
- Read 2, go to the left child of 3, go to the right child of 1, since the right child is empty insert 2 as right child of 1
- Read 4, go to the right child of 3, since the right child is empty insert 4 as right child of 3

Then the binary tree stored in your program should be:



Due Tuesday, April 19

The final step of the program is to output this BST to a file. One way of representing a BST is by outputting its nodes through preorder traversal.

Preorder traversal means:
Access the current node.
Traverse the left subtree by recursively calling the pre-order function.
Traverse the right subtree by recursively calling the pre-order function.

Then doing a preorder traversal on the previous BST will give:
3
1
2
4

To help visualize, lets also add the route you have to take to visit each node where *x* means root node, *l* means going to the left child and *r* means going to the right child.

[x] 3
[xl] 1
[xlr] 2
[xr] 4

This is what the output file of your program should look like. It should contain one node per line outputted in preorder traversal and include the route you take during the traversal.

The output file can be read as:

3 is at [x] so it is root
1 is at [xl] so you have to go root -> left to access it
2 is at [xlr] so you have to go root->left->right to access it

4 is at [xr] so you have to go root->right to access it

Execute your program:

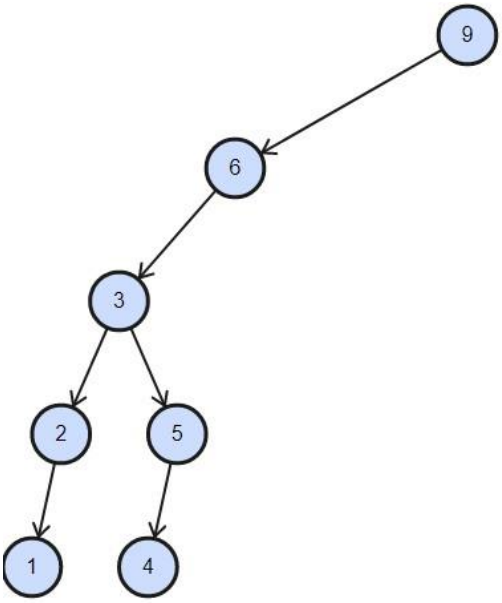
./bstmaker input=input31.txt output=output31.txt

Due Tuesday, April 19

Content of input31.txt:

9 6 3 2 5 4 1

BST in program:



Expected output31.txt:

[x] 9
[xl] 6
[xll] 3
[xlll] 2
[xllll] 1
[xllr] 5
[xllrl] 4

Due Tuesday, April 19

Important direction to submit your ga3:

We expect every student of a group will participate to solve the problem and discuss with each other. The purpose of this GA is to learn the basic functionality of Linked list. If someone faces trouble understanding, they should discuss with their friends to get a clear understanding of the Linked list. It is encouraged to help your friends, but no copy paste. If you help, it will also help you to get a deep understanding.

Every student of each group needs to submit ***the same copy*** of the solution. GA3 needs to be turned in to our Linux server. Make sure to create a folder under your root directory, name it "***ga3"*** (*name must be in lower case and exact*), only copy your .cpp and .h to this folder, no test case or other files needed. If you use ArgumentManager.h, don't forget to turn it in too. ***GAs will be graded only once***. You will not get the chance of resubmission after grading. So, make sure you are submitting correct solution before the due date.