Due Tuesday, November 29, 2022 by 11:59pm

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.

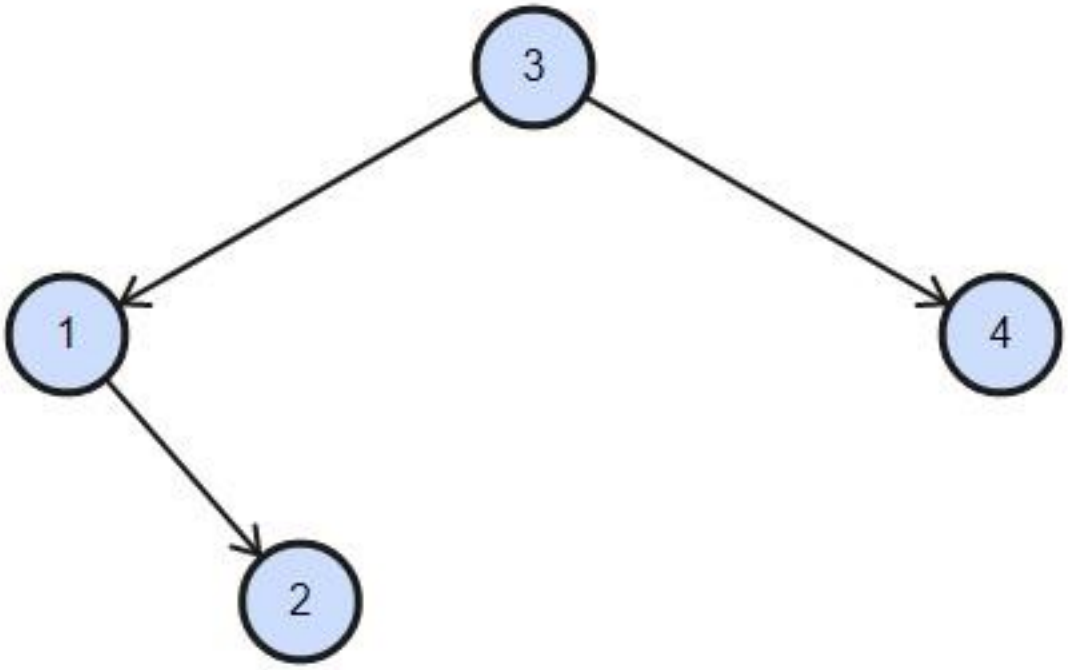
The input file will consist of a single line of numbers separated by spaces. You may assume that there will always be less than 100 numbers and they will all be unique. Your C++ program must read these numbers in order and add them to a binary search tree with the first number becoming the root node. 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:



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.

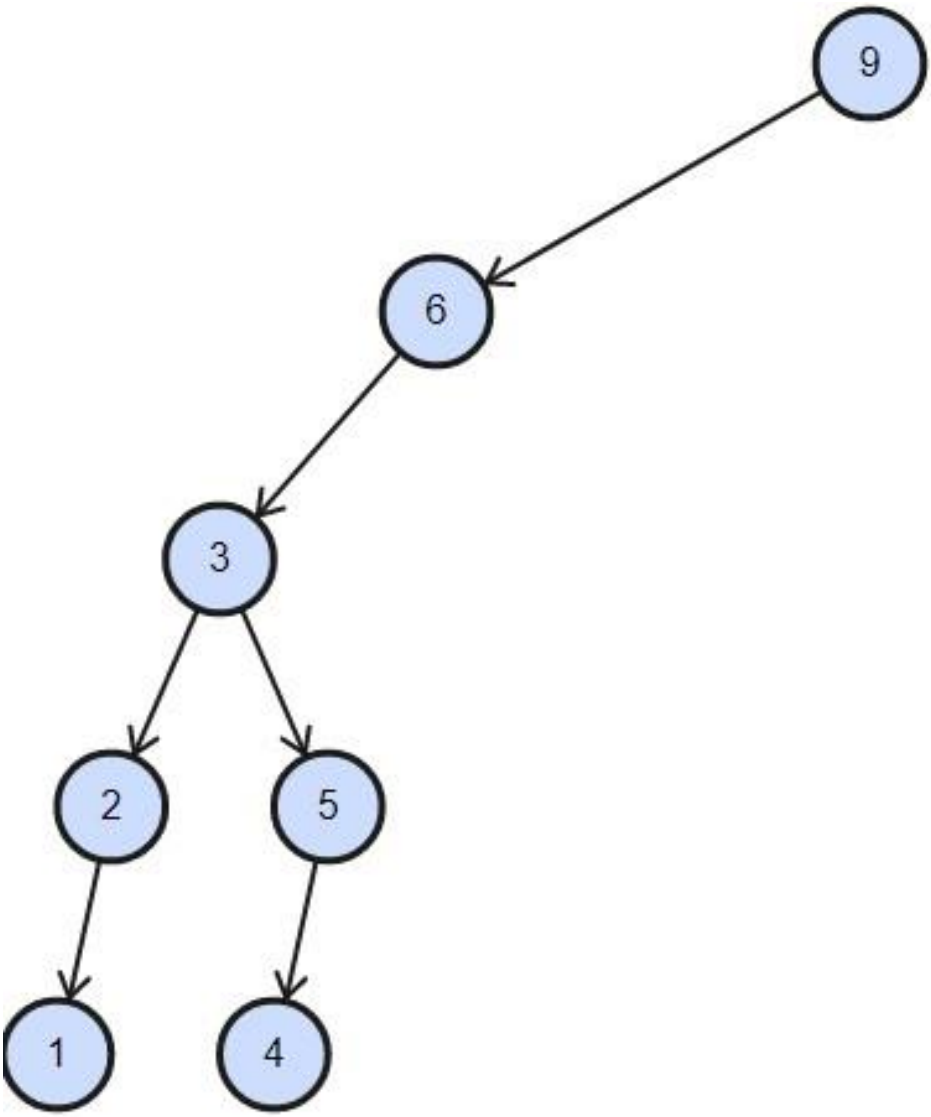
The output file can be read as:

3 is at [x] so it is the 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

input1.txt:

9 6 3 2 5 4 1

BST in program:



Expected output1.txt:

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

Important direction to submit your group assignment 3:

We expect every student of a group will participate to solve the problem and discuss with each other. The purpose of this group assignment is to learn the basic functionality of binary search trees. If someone faces trouble understanding, they should discuss with their group members to get a clear understanding of the binary search trees.

Every student of each group needs to submit the same copy of the solution. Group assignment 3 needs to be turned into the Linux server for this class. Make sure to create a folder under your root directory called **ga3** (folder name must be in lower case and exactly written as ga3). Only copy your .cpp files, .h files, and ArgumentManager.h file to this folder, no test case files or other files are needed. **Group assignments will be graded only once.** You will not get the chance of resubmission after grading, so make sure you are submitting the correct solution before the due date.

**Note: This document may contain typos. If anything seems illogical, please contact one of the TAs for clarification.