Announcements

1. Want to look at your exam? Go to Exam Exclusive OH!
2. Lab Quacks due Sunday
3. MP1 (Lists) Extra Credit due Monday

1117

Join Code: **225**

Warm Up Question: What does the prefix in the tree traversals represent?

Post-order, **In**-order, **Pre**-order…

when the root is called relative to T_L & T_R

Post	In	Pre
T_L, T_R, r	T_L, r, T_R	r, T_L, T_R

Try out the warm up if you're early!

ILLINOIS

Tree Reconstruction, Traversals, and Search

Join Code: **225**

Learning Objectives

1. Reconstruct a tree from traversals
2. Implement a level order traversal
3. Analyze the runtime of traversals
4. Analyze the space and time complexities of searches

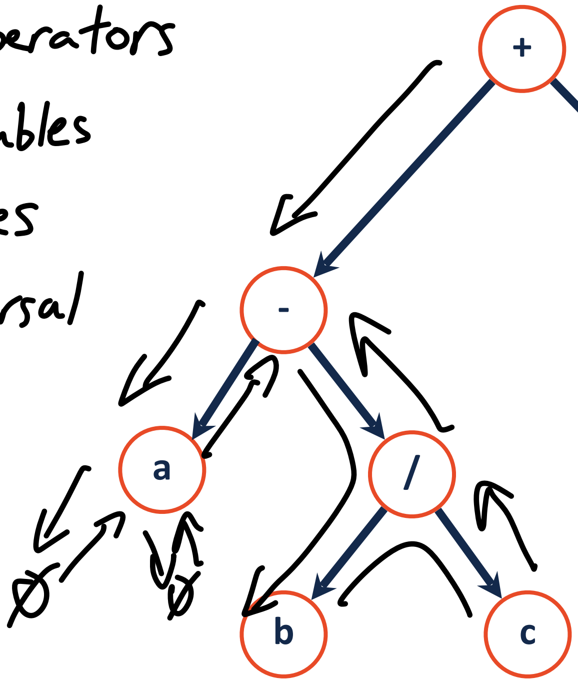
Accessing all nodes

Internal - operators

Leaves - variables

Expression Trees

Post Order Traversal



Output: abc / - de * +

$a \frac{b}{c} -$ Reverse Polish Notation

$(a - \frac{b}{c})$

Combining information

Preorder Output: + - a / b c * d e

Postorder Output: a b c / - d e * +

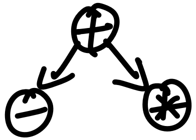
Pre

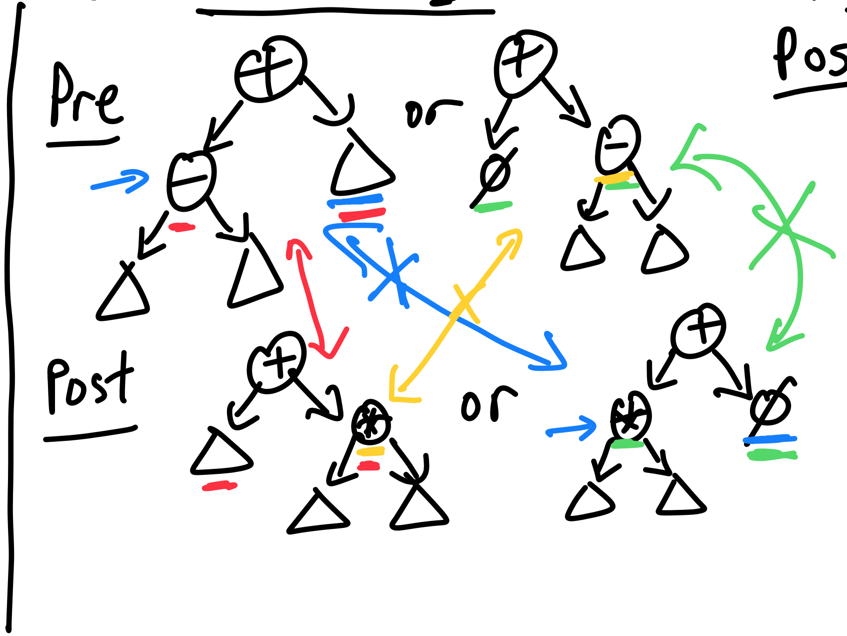
Root
Left
Right

Post

Left
Right
Root

△ - General Tree

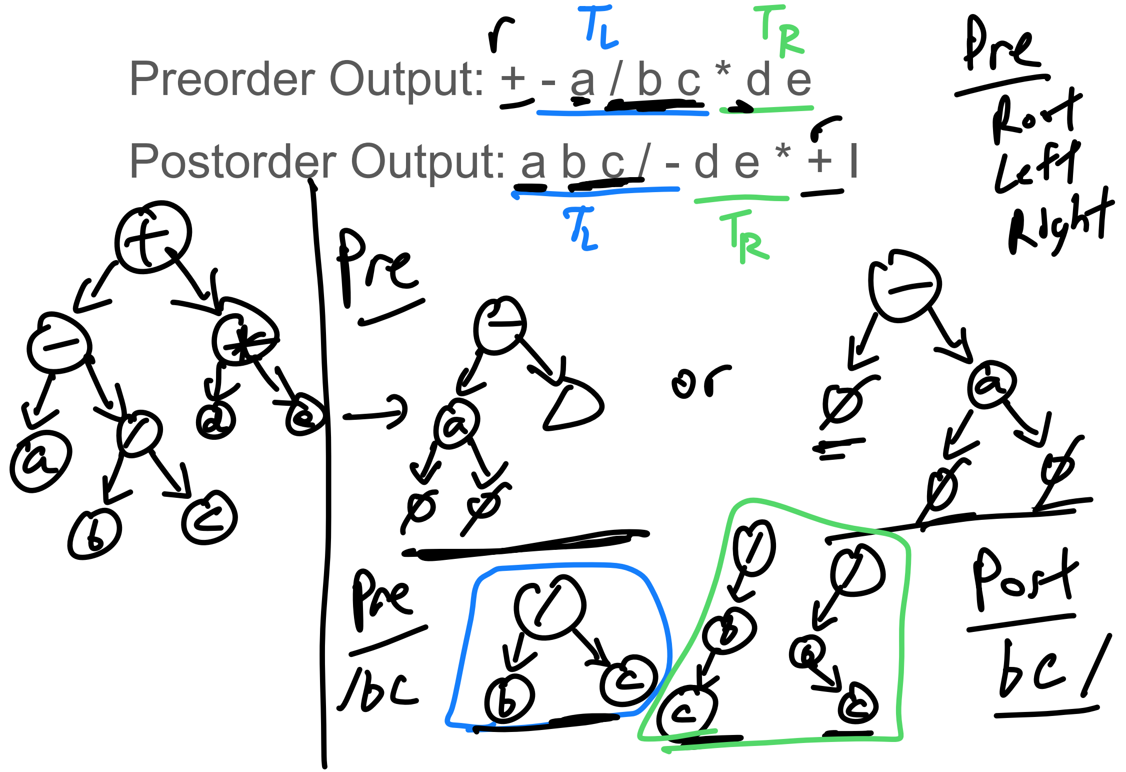




Combining information

Preorder Output: + - a / b c * d e

Postorder Output: a b c / - d e * + l

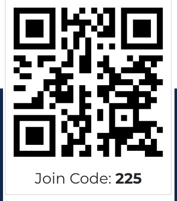


Post - Left
Right
Root

1st output of Post Order Traversal is a leaf

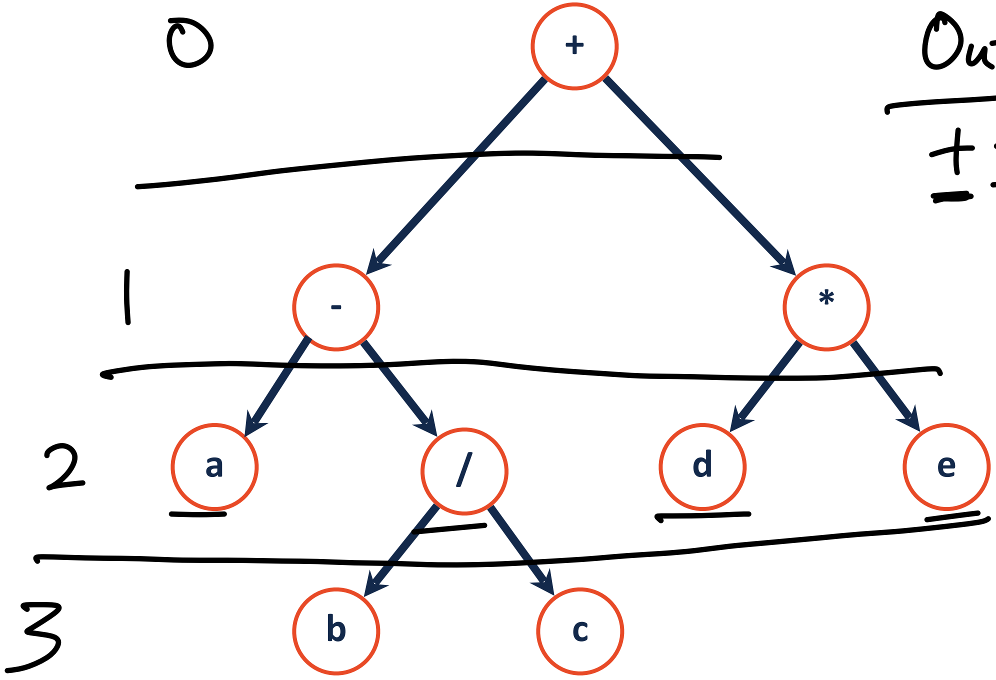
cb/

Accessing all nodes



- In-order
- Pre-order
- Post-order

Level order



Output:

+ - * a / d e b c

```
Level Order(root){
    Queue q;
    If(root) q.enq(root);
    while (q.not_empty()){
        node = q.dequeue();
  →     visit node
        q.enq(node->left)
        q.enq(node->right)
    }
}
```

What data structure can I use to help implement this?

Level Order Traversal

Join Code: **225**

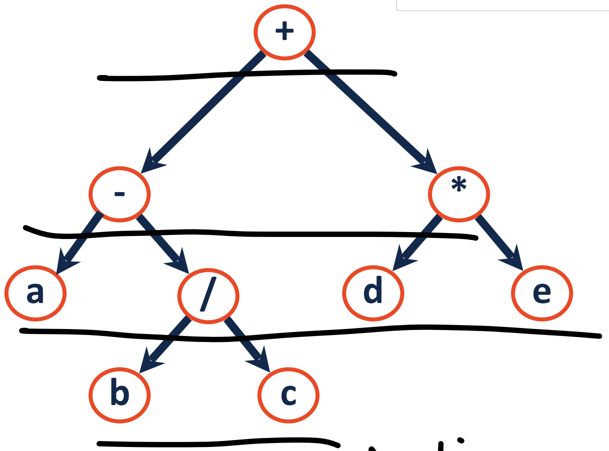
```
template<class T>
void BinaryTree<T>::levelOrder(TreeNode *
root) {
        std::queue<TreeNode *> q;
        q.push(root);
        while(!q.empty()){
            TreeNode* n = q.front();
            q.pop();
            if(n){
                std::cout << n->data;
                q.push(n->left);
                q.push(n->right);
            }
        }
}
```

} dequeue

n != nullptr

$0x0 \rightarrow$ false

$!0x0 \rightarrow$ true



n: # of nodes in binary tree

# of symbols in expression

$O(n)$

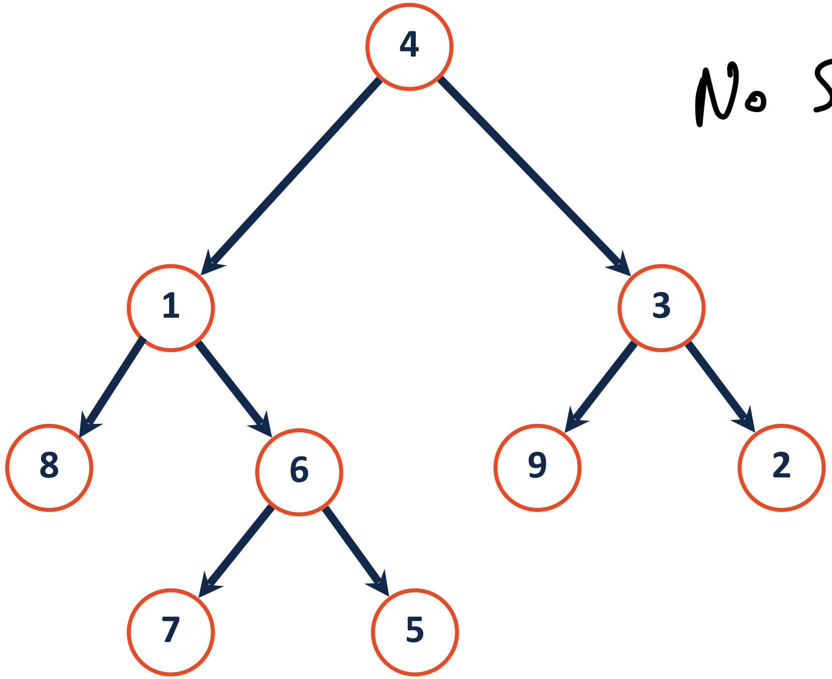
What is the runtime?

Tree Search

Tree Search - find a node of value v in the tree or return that it does not exist

Traversal

No Structure



Tree Search

2:00

Search Type	Traversal Used	Time Complexity	Space Complexity
Depth First Search (DFS)	Pre Post In	$O(n)$	$O(h)$
Breadth First Search (BFS)	Level	$O(n)$	$O(2^h)$

Width

Large Trees

Breadth-First -

Depth-First -