CS 225
Data Structures

February 26 – Trees
G Carl Evans
Trees

“The most important non-linear data structure in computer science.”
- David Knuth, The Art of Programming, Vol. 1

A tree is:

•

•

•
Binary Tree – Defined

A *binary tree* $T$ is either:

- OR

-
Tree Property: height

$\text{height}(T)$: length of the longest path from the root to a leaf

Given a binary tree $T$:

$\text{height}(T) =$
Tree Property: full

A tree $F$ is **full** if and only if:

1.

2.
Tree Property: perfect

A perfect tree $P$ is defined in terms of the tree’s height.

Let $P_h$ be a perfect tree of height $h$, and:

1.

2.
Tree Property: complete

**Conceptually:** A perfect tree for every level except the last, where the last level is “pushed to the left”.

**Slightly more formal:** For all levels $k$ in $[0, h-1]$, $k$ has $2^k$ nodes. For level $h$, all nodes are “pushed to the left”.

![Diagram of a tree with nodes labeled A, S, C, X, 2, 2, 5, Y, Z. The root is labeled C, with children labeled S and X. S has children labeled A and 2, X has children labeled 2 and 5, A has children labeled Y and Z.](image-url)
Tree Property: complete

A **complete** tree $C$ of height $h$, $C_h$:
1. $C_{-1} = \{\}$
2. $C_h$ (*where* $h > 0$) = $\{r, T_L, T_R\}$ and either:

   $T_L$ is __________ and $T_R$ is __________

   OR

   $T_L$ is __________ and $T_R$ is __________
Tree Property: complete

Is every **full** tree **complete**?

If every **complete** tree **full**?
Tree ADT
Tree ADT

*insert*, inserts an element to the tree.

*remove*, removes an element from the tree.

*traverse*,
#pragma once

template <class T>
class BinaryTree {
    public:
        /* ... */
    
    private:

};
Trees aren’t new:
Trees aren’t new:
How many NULLs?

**Theorem:** If there are $n$ data items in our representation of a binary tree, then there are ___________ NULL pointers.
How many NULLs?

Base Cases:

n = 0:

n = 1:

n = 2:
How many NULLs?

Induction Hypothesis:
How many NULLs?

Consider an arbitrary tree $T$ containing $n$ data elements:
Traversals
Traversals

```
template<class T>
void BinaryTree<T>::__Order(TreeNode * root)
{
    if (root != NULL) {
        ______________________;
        __Order(root->left);
        ______________________;
        ______________________;
        __Order(root->right);
    }
}
```
Traversals

```
template<class T>
void BinaryTree<T>::__Order(TreeNode * root)
{
    if (root != NULL) {
        ______________________;
        __Order(root->left);
        ______________________;
        ______________________;
        __Order(root->right);
        ______________________;
    }
}
```
Traversals

template<class T>
void BinaryTree<T>::__Order(TreeNode * root) {
    if (root != NULL) {
        ______________________;
        ___Order(root->left);
        ______________________;
        ______________________;
        ___Order(root->right);
        ______________________;
    }
}