CSCI 136: Data Structures and Advanced Programming
Lecture 23
Trees, part 2
Instructor: Kelly Shaw
Williams

Your to-dos

1. Read before Wed: Bailey, Ch 14.4.
2. Lab 7 (partner lab), due Tuesday 11/8 by 10pm.

Topics

Tree terminology

Announcements

CSCI 136 final exam
Saturday, December 17 at 1:30pm
Room TBD
A **tree** is a recursive data structure that stores information hierarchically. A tree is either:

- **empty** (i.e., $\emptyset$), or
- a **node** containing a **value** and references to one or more **trees**.

The empty tree: A non-empty **binary** tree:

$$\emptyset$$

Let's implement this together.

**Terminology**

The topmost node is called the **root**.

**Properties of trees**

**Connected**: every node in a tree is **reachable** by following a single unique **path** starting from the **root** node.
Properties of trees

*# edges:* a tree having *n vertices* always has *n-1 edges.*

Terminology

The nodes at the bottom of a tree are called *leaves.*

Terminology

Any node that is not a leaf is an *interior node.*

Terminology

A node may have *children.*
A node may have children.

A node that has children is called the parent of those children.

For a given node, all of the nodes above it are called ancestors.
Properties of trees

**Single ancestor:** every node in a tree has at most one ancestor.

![Diagram of a tree with a single ancestor]

Terminology

For a **given node**, all of the nodes below it are called **descendants**.

![Diagram of a tree with descendants]

Properties of trees

**Subtrees:** the descendents of every tree (except the empty tree) are also trees.

![Diagram of a tree with subtrees]

Terminology

The **degree** of a tree is the maximum number of **children** had by any node.

Degree of this tree: 2

Degree 2 trees are common: we call them **binary trees**.
A tree that is missing no leaves is **full**.

A **path** is a sequence of edges between **two nodes**.

**Cycle-free**: no path will ever revisit the same node.

The **length** of a path is the **number of edges** in the path.

Length = 2
Terminology

The height of node n is the length of the longest path between n and any leaf.

Height of n = 1

Terminology

The height of a tree is the length of the longest path between the root and any leaf.

Height of tree = 2

Terminology

The depth of node n is the length of the longest path between the root and n.

Depth of n = 1

Terminology

The level of any node is its depth.

Level 0

Level 1

Level 2
Terminology

The depth of \( n \) + the height of \( n \) ≤ the height of the tree.

\[
\text{depth of } n: 1 + \text{height of } n: 0 \leq \text{height of tree: } 2
\]

Properties of trees

**Directed** or **undirected**: trees can be either directed, meaning that traversals can only happen in one direction, or undirected, meaning that traversals can happen in any direction.

The tree shown here is directed. We can represent an undirected tree using back edges.

Is a list a tree?

Yes, a list is a tree whose nodes have **degree** 1. We call such trees **degenerate**.
Activity

Encode this binary tree using BinaryTree<T>

Recap & Next Class

Today:

Tree terminology

Next class:

Tree traversals