| CSCI 136: |
| :---: |
| Data Structures |
| and |
| Advanced Programming |
| Lecture 24 |
| Trees, part 2 |
| Instructor: Dan Barowy |
| Williams |

## Your to-dos

1. Read before Fri: reading on balanced trees.
2. Lab 8 (solo lab), due Tuesday $4 / 26$ by 10pm.

## Topics

Binary tree implementation
Tree height

## Tree ADT

A tree is a recursive data structure that stores information hierarchically. A tree is either:
-empty (i.e., Ø), or

- a node containing a value and references to one or more trees.
The empty tree:
A non-empty binary tree:




## Binary Tree Height

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


Height of tree $=2$

## Binary Tree Height

Let's think about some corner cases.
What is the height of a tree with just one node?

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

$$
\text { Height of tree }=0
$$

## Binary Tree Height

Here's a more formal definition.

The height of a tree is defined as:

- -1 if the tree is empty, or
- height(left) or height(right), whichever is bigger, + 1

empty tree: -1
- just a root: 0
any other tree: longest path


## Binary Tree Height

Let's think about some corner cases.
What about the empty tree?


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

$$
\text { Height of tree }=-1
$$

## Binary Tree Height

How might we implement getHeight()?

## Binary Tree Height

Let's implement this together.

## Recap \& Next Class

## Today:

Binary tree implementation
Tree height

## Next class:

Binary tree traversals
Binary search trees
Tree balance
Asymptotic analysis

Height

$$
1-2^{4} \times 2
$$



