Trees (Intro)

Instructors: Sam McCauley and Dan Barowy

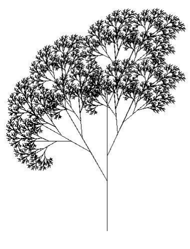
April 15, 2022

Admin

- Sign up to be a TA! Deadline next week.
- Any questions?

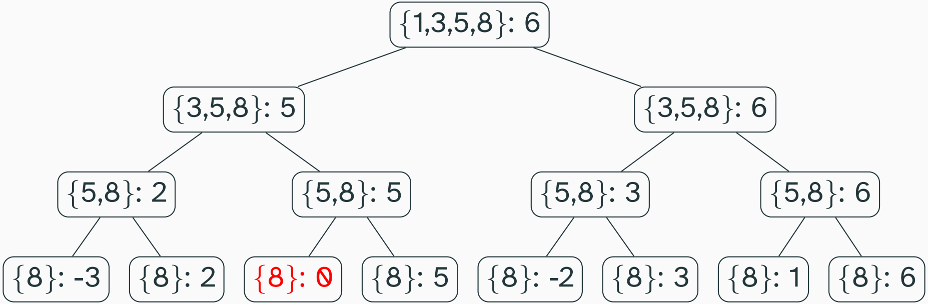
Trees

Trees



- All the ways we've had to store data has been one-dimensional.
  - At the end of the day: every item in our data structure is the *i*th item in the data structure for some *i*
  - All of our access has (indirectly) been through such a one-dimensional mapping
- With trees, we add a second dimension to how we store data
- *Drastic* improvements in what we can store and the performance we can achieve

Trees We've Seen



We can draw the method calls made by a recursive algorithm using a tree! (The above is `canMakeSum()` from lab 3.)

Here: each of the rectangles above (called a *node*) represents a recursive call. We connect each method to the methods it calls.

Trees We've Seen

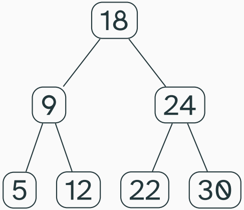
5	9	12	18	22	24	30

Calling back to last lecture: what happens when we do binary search on this array?

Something like: first, we compare our query element to 18. Based on the result, we then compare it to either 9 or 24.

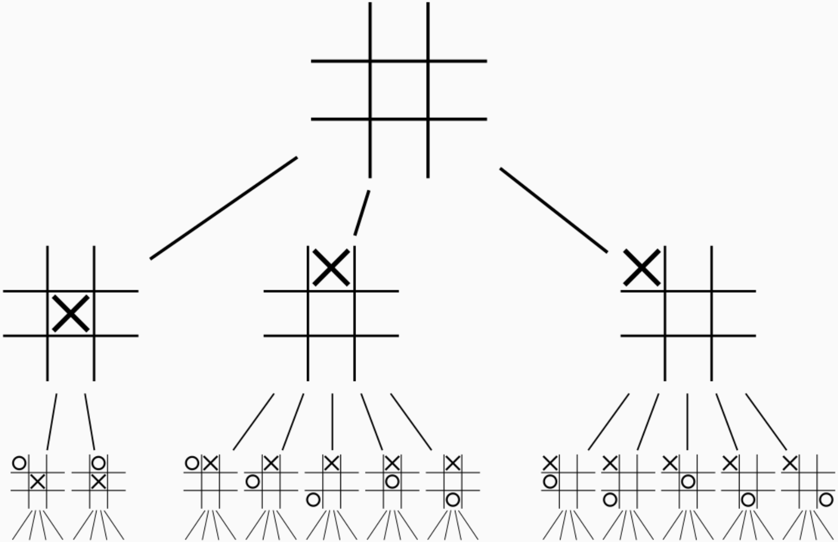
Trees We've Seen

5	9	12	18	22	24	30

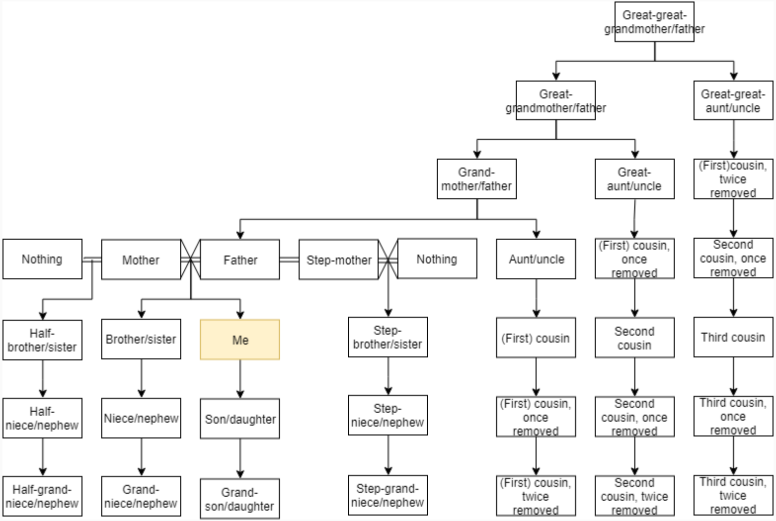


Binary search seems to also have a tree-like structure. We'll see how to store data in a very similar tree very soon.

Game Tree



Family "Tree"

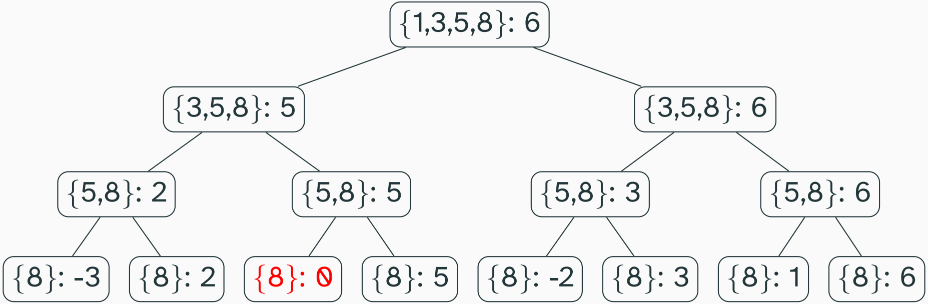


Same basic idea. Though note: not quite a tree by our definition.

Basic Tree Vocabulary

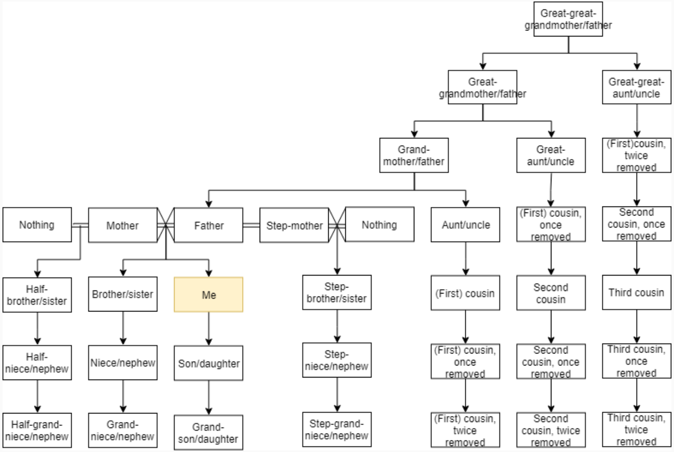
- Tree consists of nodes (the boxes in the images we saw above)
- Nodes are connected by edges (lines in the images we saw above)
- There is one root node that does not have a *parent node*
- Every other node has exactly one *parent node*
- Nodes may have some children.
- A node without a child is called a *leaf*

Labelling nodes



What is the root node in this tree? What are the leaves?

Family "Tree"



Why isn't this a tree?

- Answer: nodes have multiple parents! (Plus there are some extra edges/different types of edges in this image.)

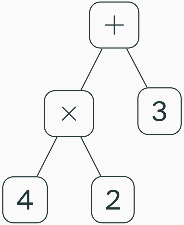
Binary Tree

Binary Tree

- Binary Tree: A tree where each node has at most 2 children
- The degree of a node is the number of children it has. So a binary tree is a tree where all nodes have degree at most 2.
- Let's see an example of a binary tree. Then, we'll discuss the `BinaryTree` class that comes with `structure5`

Expression Tree

$$\boxed{4 \times 2 + 3}$$



We can write arithmetic expressions using a binary tree. (Why is it binary?)

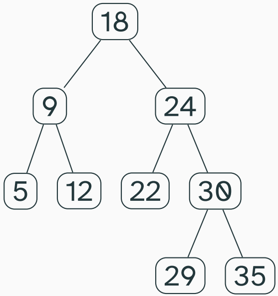
Using a Binary Tree

- *Goal*: store an expression using a binary tree
- Then: evaluate the expression
- Takeaway: practice with binary trees

How to Store a Binary Tree?

- Nodes should probably be objects of some class type.
- Store its children
- In the `SinglyLinkedList`, we had a hidden `Node` class; the `SinglyLinkedList` itself only stored a pointer to the `head`
- `BinaryTree<E>` does not work that way! Just a single recursive class

Visualizing Trees Recursively



Each node in a (binary) tree can be viewed as the root of its own (binary) tree.

BinaryTree plan

- Each node is stored as a `BinaryTree` object
- Stores the value stored at the node
- Stores the parent (of type `BinaryTree`)
  - The root of the tree stores `null` for its parent
- Stores the left and right children (both are of type `BinaryTree`
  - If either doesn't exist, points to an *empty node* (similar to dummy nodes)
  - Children of an empty node *point to the node itself*
  - There are other ways to implement missing children in a binary tree; this is just one
- Let's take a look at the code for `BinaryTree`
- Now, let's look at how we can evaluate a tree of expressions stored in a `BinaryTree`

More Binary Tree Vocabulary: Height

- The *size* of a tree is the number of nodes it contains
- The *depth* of a node n is the number of edges between n and the root.
- The *height* of a tree is the largest height of any node in the tree.

Binary Tree Practice

- How can we calculate the size of a binary tree?
  - Hint: use recursion!
  - Let's look at how the `BinaryTree` class implements this
- How can we calculate the depth of a binary tree?
  - Recursion again!
  - Let's look at how this is implemented