LEC 13

CSE 123

Binary Trees: Modification

Questions during Class?
Raise hand or send here

sli.do #cse123

BEFORE WE START

Talk to your neighbors:

Red button or blue button?

Instructor: Nathan Brunelle

TAs:

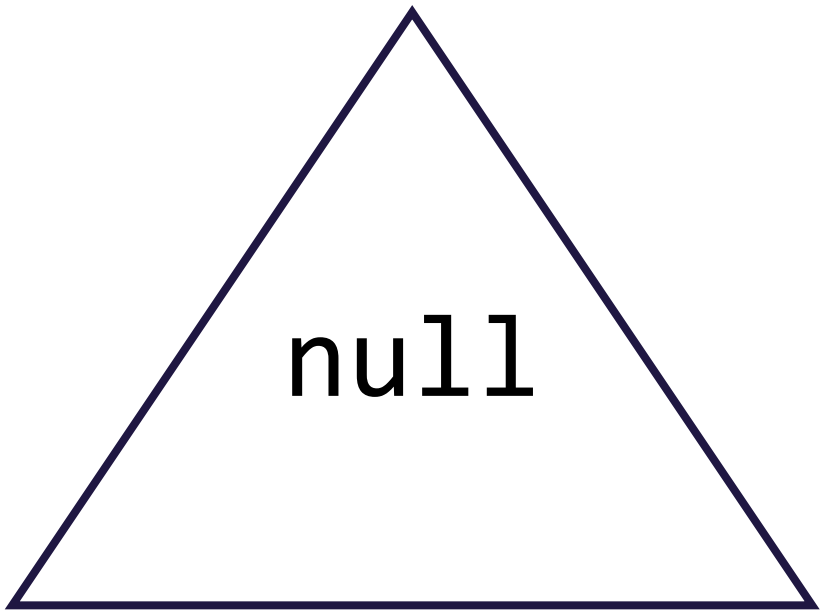
Cora	Chris	Crystal	Anya	Evan
Jonah	Nhan	Saachi	Ishita	Sahana
Chloe	Shiven	Nate	Renee	Yuntong
Amiya	Shashmit	Akshaya	Vihan	Shreedhaarini
Roshan	Adiyan	Nancy	Rachel	Ananya
Ryuta	Mimi	Isaac	Megan	

Announcements

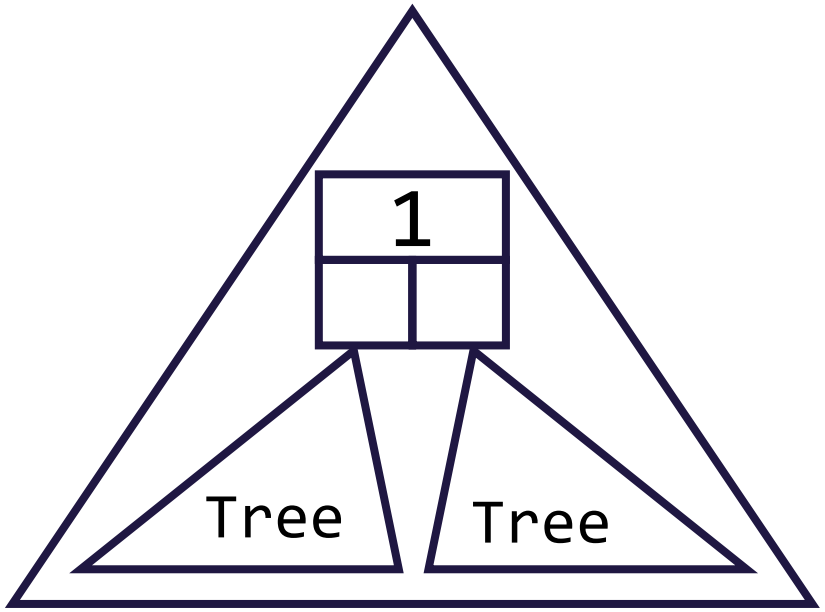
- Programming Project 2 Due Tonight @ 11:59pm
- Computer Quiz 2 Begins Tomorrow
- Programming Project 3 Released Tomorrow

Review: Binary Trees

- A Binary Tree is either:



Empty tree



Node w/ two subtrees

This is a recursive definition!
A tree is either empty or a node with two more trees!

Review: Binary Tree Programming

- Programs look very similar to Recursive LinkedList!

- Guaranteed base case: empty tree
  - Simplest possible input, should immediately know the return
- Guaranteed public / private pair
  - Need to know which subtree you're currently processing
- If modifying, we use `x = change(x)`
  - Don't stop early, return updated subtree (`IntTreeNode`)

Review: Binary Tree Traversals

- 3 different primary traversals
  - All concerned with when you process your current root

- Pre-order traversal:
  - Process **root**, left subtree, right subtree
- In-order traversal
  - Process left subtree, **root**, right subtree
- Post-order traversal:
  - Process left subtree, right subtree, **root**

Sometimes different traversals yield different results

Modifying Binary Trees

- Like linked lists, cannot modify nodes
  - Because data field is `final` (there are good reasons for this)
- Will need to create and insert new nodes
- Use `x = change(x)`, usually ***3 times***
  - overall root (in public method)
  - left subtree
  - right subtree

- Order might matter!
  - Does operation on root depend on children?