Questions during Class?
Raise hand or send here

sli.do #cse123

**BEFORE WE START**

*Talk to your neighbors:*

Recap Quiz 2 - how did it go? What will you do differently in preparing for Quiz 3?

Music: [123 24su Lecture Tunes](#)

**Instructor:** Joe Spaniac

**TAs:** Andras Daniel, Eric Nicole, Sahej Trien, Zach
Lecture Outline

• Announcements

• Binary Trees
  - Terminology
  - Recursive Definition
  - Tree Traversal

• Programming Binary Trees
Announcements

• Quiz 2 grades out!
  - Please check your grades before the next quiz, practice **metacognition**

• Quiz 3 in section on Tuesday, 7/30
  - Topics: Runtime; Recursion
  - Same policies as last time: One sheet of 8.5” x 11” notes (double-sided, printed or handwritten), 50mins, etc.

• Creative project 3 released last night, due 7/31 at 11:59pm
  - Last creative project of the quarter!

• Resubmission Period 4 closes tonight, 7/26 at 11:59pm

• Resubmission Period 5 opens tonight, due next Friday 7/19 at 11:59pm
Lecture Outline

• Announcements

• Binary Trees
  - Terminology
  - Recursive Definition
  - Tree Traversal

• Programming Binary Trees
Binary Trees

- Last data structure of the quarter!
  - Very similar to Linked Lists...
Binary Trees

• Last data structure of the quarter!
  - Very similar to LinkedLists...
Binary Trees

• Last data structure of the quarter!
  - Very similar to Linked Lists...

• Linked TreeNodes w/ 3 fields:
  - int data, TreeNode left, TreeNode right
  - Doubly complicated!

```
overallRoot

1

2

3
```

```
data

left

right

null

null

null
```
Binary Trees

• Last data structure of the quarter!
  - Very similar to LinkedLists...

• Linked TreeNode w/ 3 fields:
  - int data, TreeNode left, TreeNode right
  - Doubly complicated!

• Similar to trees?
  - Close enough!
  - Terminology: root / leaves

• Other terminology as well
Tree Terminology

- **overallRoot**
  - null
  - null
  - null
  - data
  - left
  - right
  - null
  - parent / child
  - null
  - null
  - null
  - null
Tree Terminology

- overallRoot
- null
- data
- left
- right
- siblings
- null
LinkedLists [Review]

- We’ll say that any LinkedList falls into one of the following categories:

  - **Empty list**
    
    ```
    null
    ```
    
    ```
    Empty list
    front == null
    ```

  - **Node w/ another LinkedList**
    
    ```
    [4]
    ```
    
    ```
    Node w/ another LinkedList
    front != null
    front.next = LinkedList
    ```

This is a recursive definition! A sublist is either empty or a node with another sublist!
Binary Trees

- We’ll say that any Binary Tree falls into one of the following categories:

  - **Empty tree**: `root == null`
  - **Node w/ two subtrees**: `root != null` and `root.left / root.right = Tree`

*This is a recursive definition! A tree is either empty or a node with two more trees!*
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 (i.e. root)

• If modifying, we use $x = \text{change}(x)$
  - Don’t stop early, return updated subtree (IntTreeNode)

• Let’s trace through an example together...
Binary Tree Programming

Pre-order traversal

null
null
null
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
Lecture Outline

• Announcements

• Binary Trees
  - Terminology
  - Recursive Definition
  - Tree Traversal

• Programming Binary Trees