CS 225

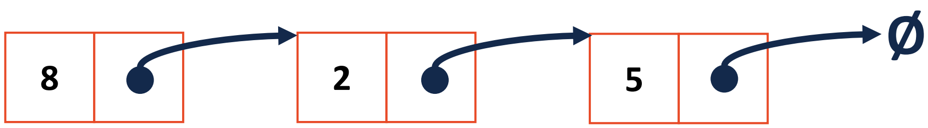
Data Structures

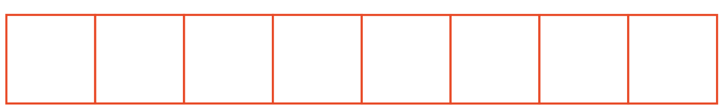
October 14 – Disjoint Sets and Iterators

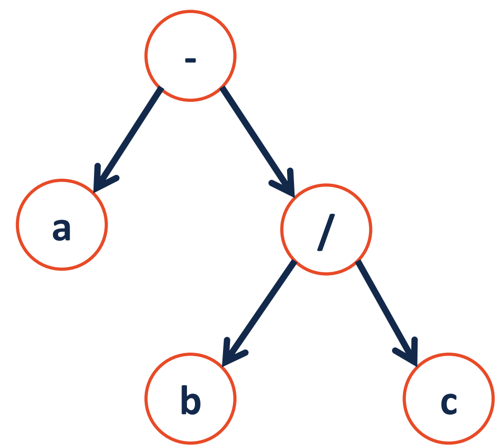
G Carl Evans

Iterators

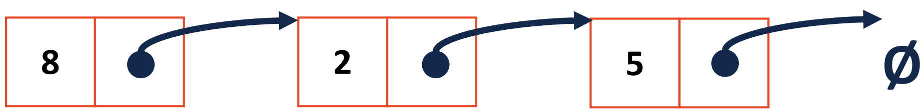
Suppose we want to look through every element in our data structure:

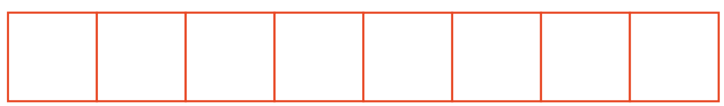


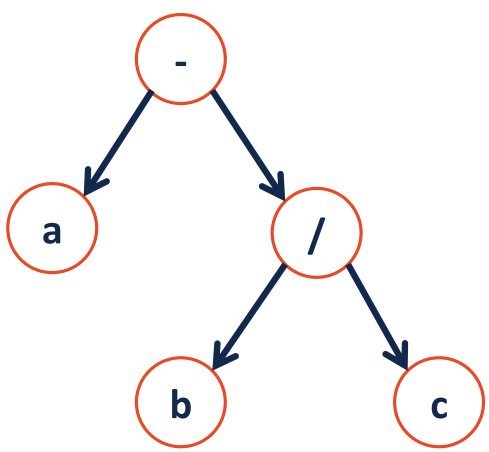




Iterators encapsulated access to our data:

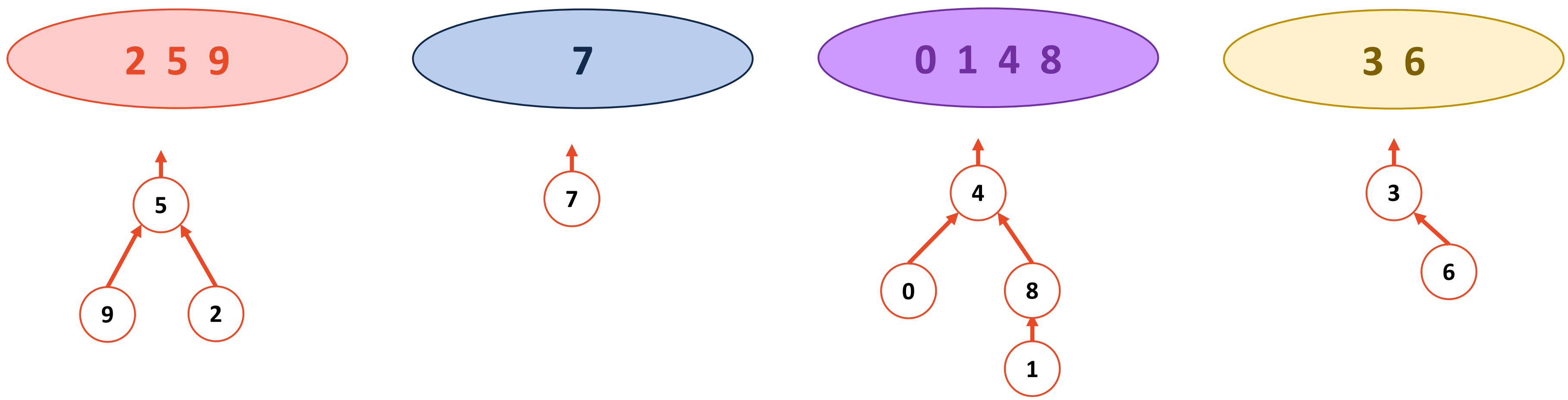






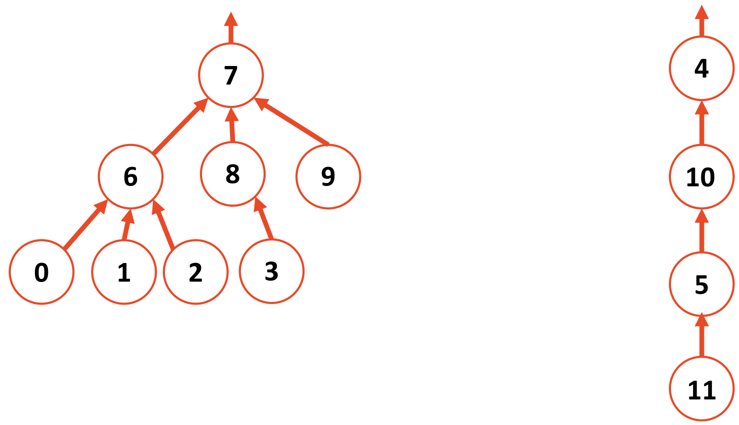
Cur. Location	Cur. Data	Next
`ListNode *`		
`size_t`		
`stack<Node *>`		

Disjoint Sets



0	1	2	3	4	5	6	7	8	9
4	8	5	-1	-1	-1	3	-1	4	5

Disjoint Sets – Smart Union



Union by height

0	1	2	3	4	5	6	7	8	9	10	11
6	6	6	8	-4	10	7	-3	7	7	4	5

***Idea**: Keep the height of the tree as small as possible.*

Union by size

0	1	2	3	4	5	6	7	8	9	10	11
6	6	6	8	-8	10	7	-4	7	7	4	5

***Idea**: Minimize the number of nodes that increase in height*

We will show the height of the tree is: log(n).

Union by Size

To show that every tree in a disjoint set data structure using union by size has a height of at most O(log n) we will show that the inverse.

Base Case

Inductive Hypothesis

Union by Size

Case 1

Union by Size

Case 2

Union by Height - Rank

Base

New UpTrees have Rank =

When you join two UpTrees

Union by Rank

1. For all non-root nodes ***x, rank(x) < rank(parent(x))***

2. Rank only changes for roots and only up

Union by Rank

Much like before we will show the min(nodes) in a tree with a root of rank $k \geq 2^k$

Base Case

IH

Union by Rank

For any integer $r \geq 0$, there are $\leq n/2^r$ nodes with rank r.

Why?

Path Compression