Binary Search Trees

Hitesh Boinpally
Summer 2023

Agenda

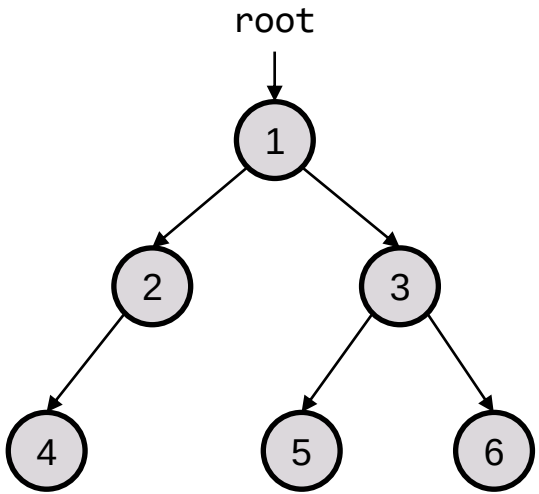
- Review of Trees
- Binary Search Trees
- Modifying a Tree

Agenda

- Review of Trees 
- Binary Search Trees
- Modifying a Tree

Trees Defined

- **Tree:** Nodes linked together in some hierarchical fashion
- **Binary Tree:** A tree where each node has at most 2 children



Recursive Definition:

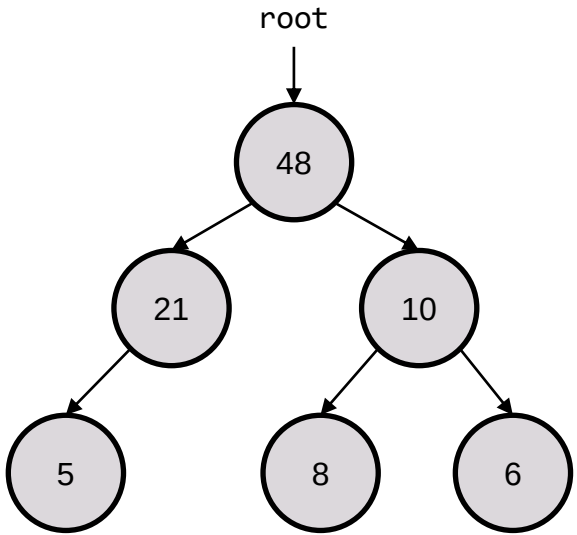
- A tree is either:
    1. Empty
    2. A node with data, and a left and right subtree

Printing Trees

- Want to print out the contents of the tree

Different ways to do so:

Pre-order	`48 21 5 10 8 6`
In-order	`5 21 48 8 10 6`
Post-order	`5 21 8 6 10 48`



Agenda

- Review of Trees
- Binary Search Trees 
- Modifying a Tree

Aside: Searching

- Given an array of numbers, check if a certain number is in it.
- How many elements do we have to go through in the worst case?

Aside: Searching

- Given an array of numbers, check if a certain number is in it.
- How many elements do we have to go through in the worst case?
    - All of them if the element isn't in the list ☹

Aside: Searching

- Given an array of numbers, check if a certain number is in it.
- How many elements do we have to go through in the worst case?
    - All of them if the element isn't in the list ☹

- What if the array was sorted?

Aside: Searching

- Given an array of numbers, check if a certain number is in it.
- How many elements do we have to go through in the worst case?
  - All of them if the element isn't in the list ☹

- What if the array was sorted?
  - Find the element 27

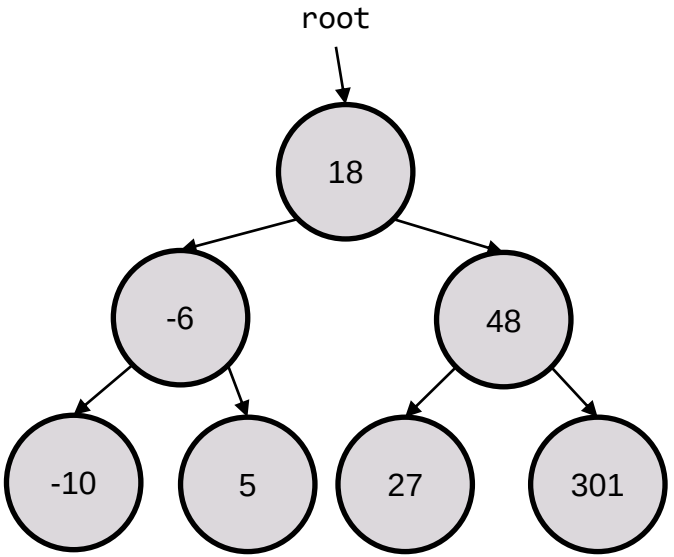
-10	-6	5	18	27	48	301

Binary Search Trees

- We can apply these properties to trees to speed up our searching ability
    - Think: `TreeSets`

Binary Search Tree (BST): A binary tree with the following property:

- For every non-empty node `root`:
    - elements of root's left subtree contain data "less than" root's data
    - elements of root's right subtree contain data "greater than" root's data
    - root's left and right subtrees are also binary search trees



Agenda

- Review of Trees
- Binary Search Trees
- Modifying a Tree 

What's the output of this program?

```
public static void main(String[] arg) {
    Point p = new Point( x: 1,  y: 2);
    change1(p);
    System.out.println(p);
    change2(p);
    System.out.println(p);
}

1 usage
public static void change1(Point p) {
    p.x = 14;
}

1 usage
public static void change2(Point p) {
    p = new Point( x: 7,  y: 8);
}
```

slido.com
code: #su_cse123

`x = change(x)`

- To modify a binary tree, need to change the `overallRoot` or a `root`'s `.left` / `.right`
  - Similar to the `front` or `.next` of a linked node

- For trees we utilize the `x = change(x)` pattern
- A few components:
  - Pass in the original value of `x`
  - Change the value somehow in the method `change`
  - Return the updated value of `x`
  - Re-assign the updated value to `x`