BINARY SEARCH TREE

Sutherland Programming Club

By Yizuo Chen, Yiyou Chen

BINARY TREE

A tree has n nodes and n – 1 edges

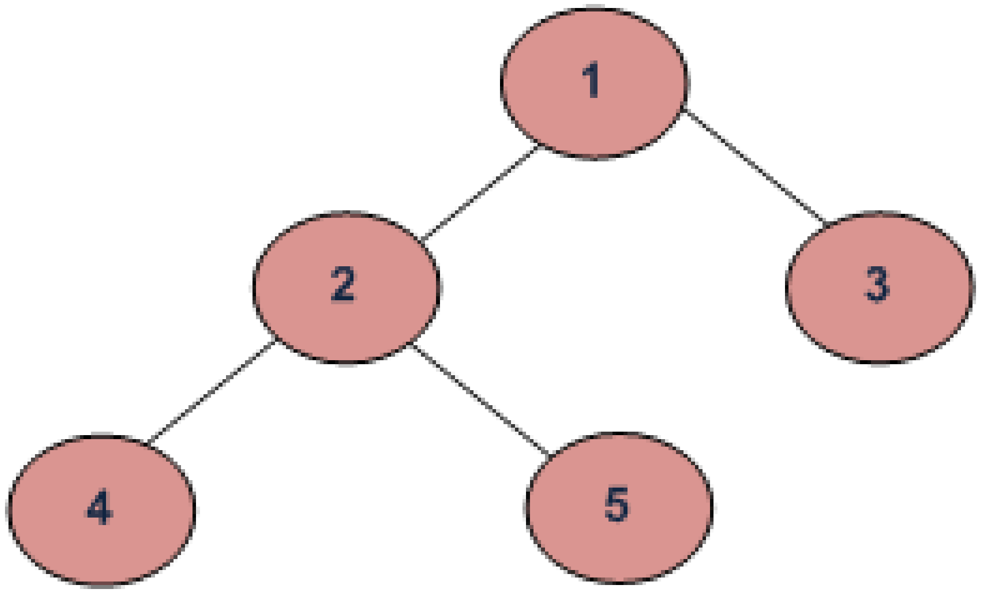
Each node has 0 – 2 children

A node has no child is called a leaf

Parent.

The nodes which have the same parent is called siblings.

TREE TRAVERSALS



Depth First Traversals:
(a) Inorder (Left, Root, Right) : 4 2 5 1 3
(b) Preorder (Root, Left, Right) : 1 2 4 5 3
(c) Postorder (Left, Right, Root) : 4 5 2 3 1

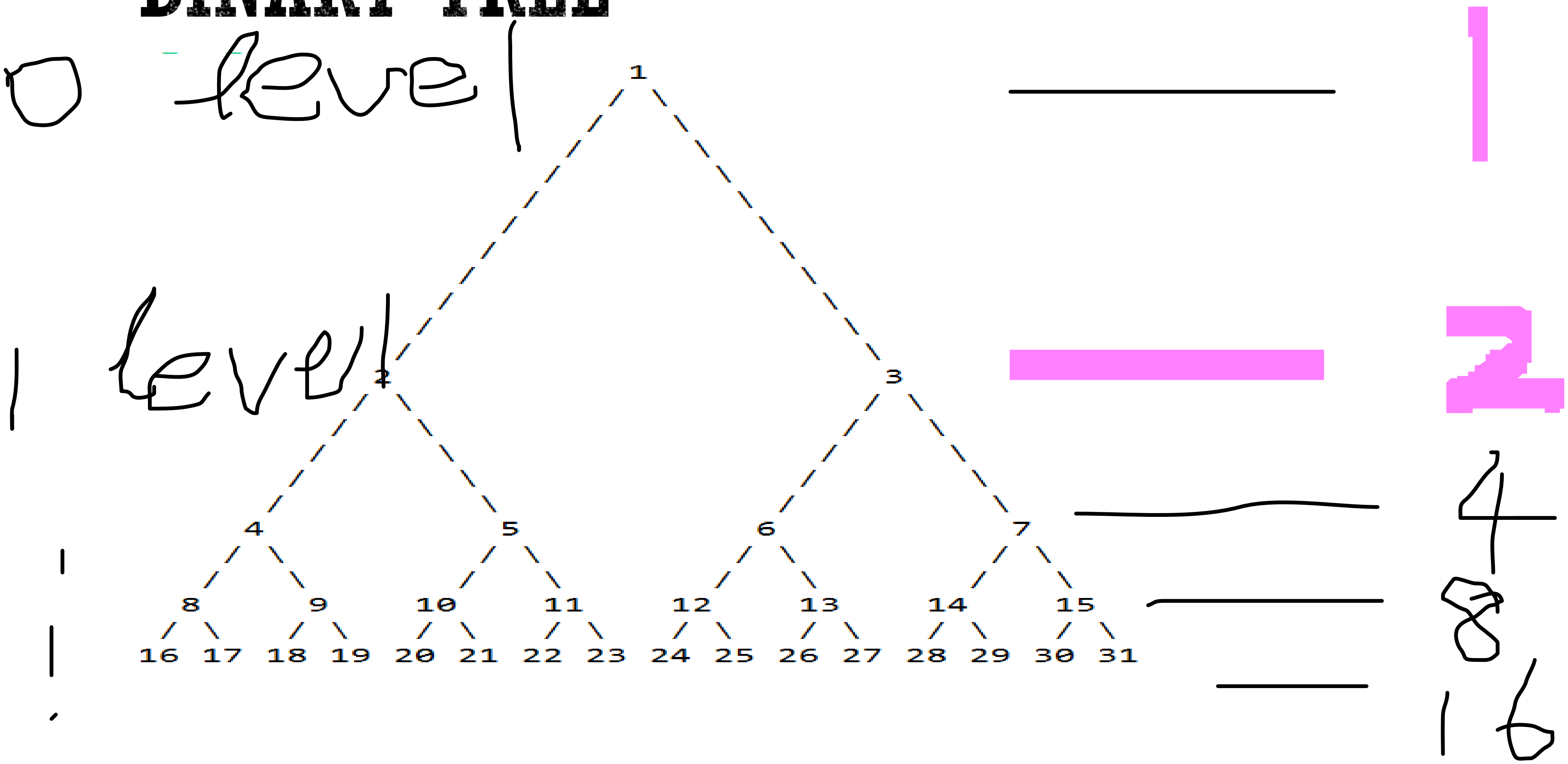
Only change the orders!

Breadth First or Level Order Traversal : 1 2 3 4 5

MORE ABOUT BINARY SEARCH TREE

- The upper most node is called root.
- The depth of a node n: the length of the unique path from the root to n.
- The height of a node n: the length of the longest path from n to a leaf.

CALCULATE THE MAXIMUM HEIGHT OF A BINARY TREE

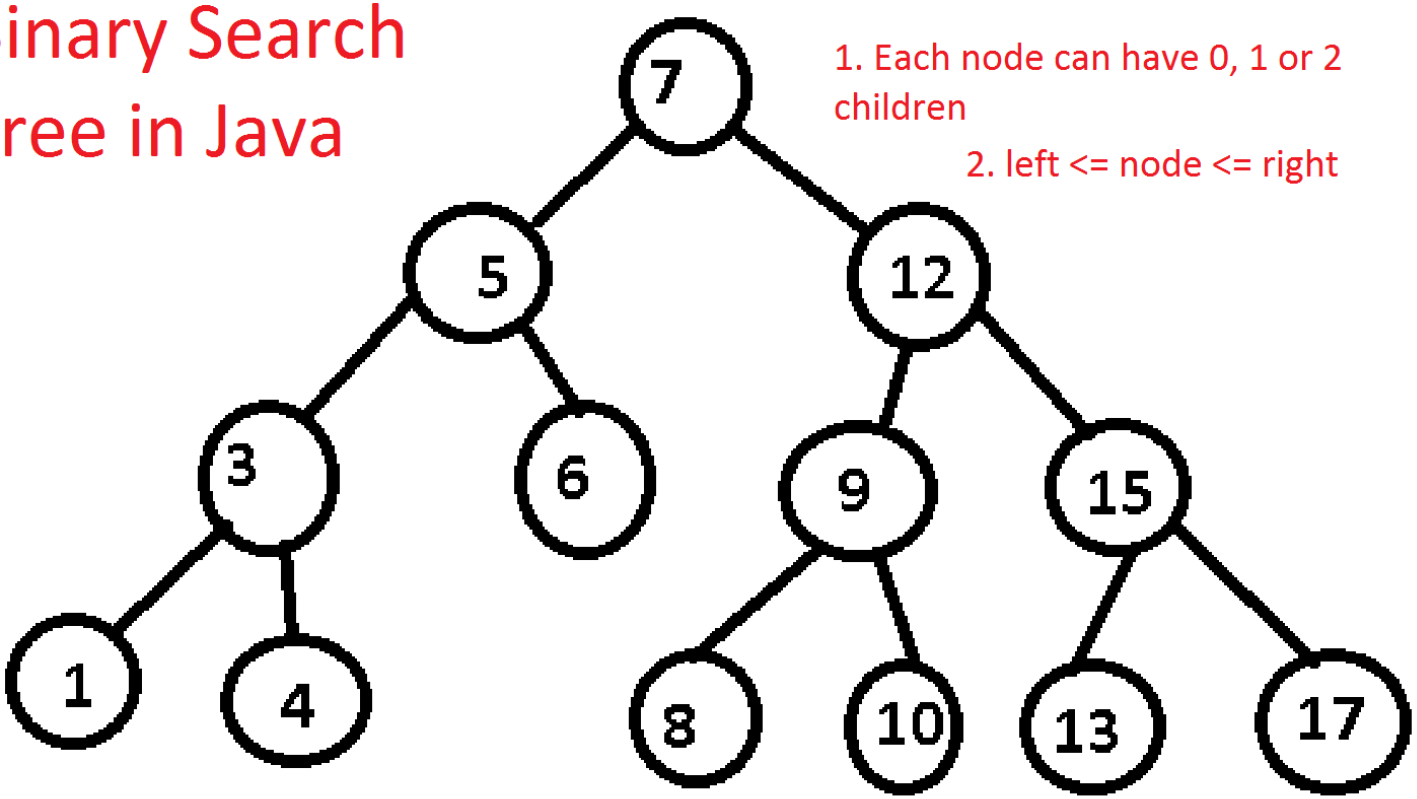


PATTERNS

- For k-th level of a binary tree, there're at most 2^n nodes.
- Total nodes to k-th level: 2^0 + 2 ^ 1 + 2^2 + .. + 2 ^ k = 2 ^ (k + 1) - 1.
- # nodes = 2 ^ (height) – 1.
- Height = log_2_(#nodes + 1).

BINARY SEARCH TREE

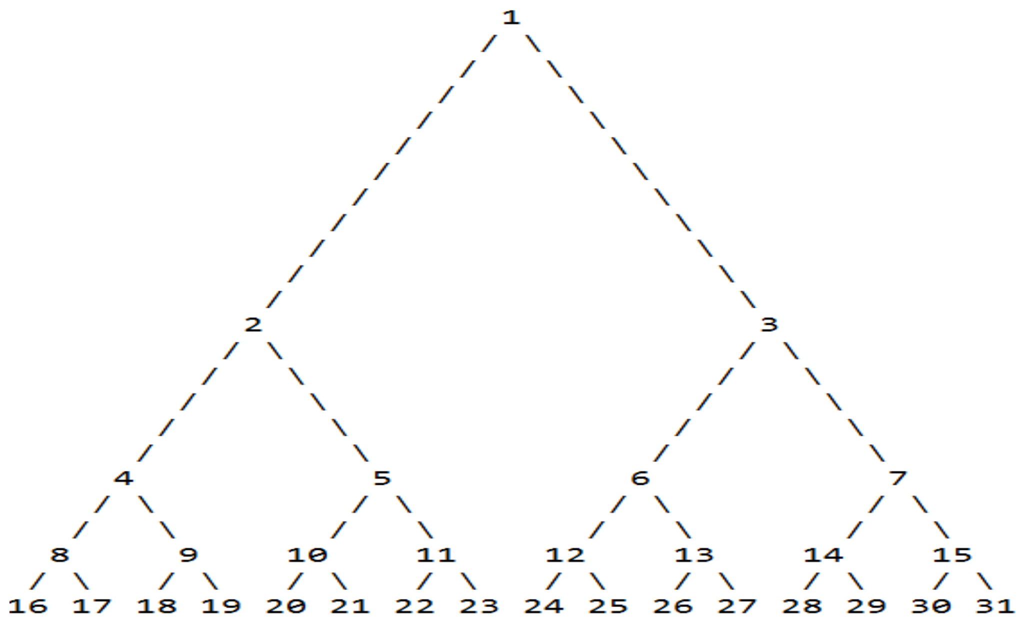
Binary Search
Tree in Java



1. Each node can have 0, 1 or 2 children
2. left <= node <= right

BUILD A BINARY SEARCH TREE

- Struct NODE{
  - NODE left;
  - NODE right;
  - Datatype value;
- }tree[size];



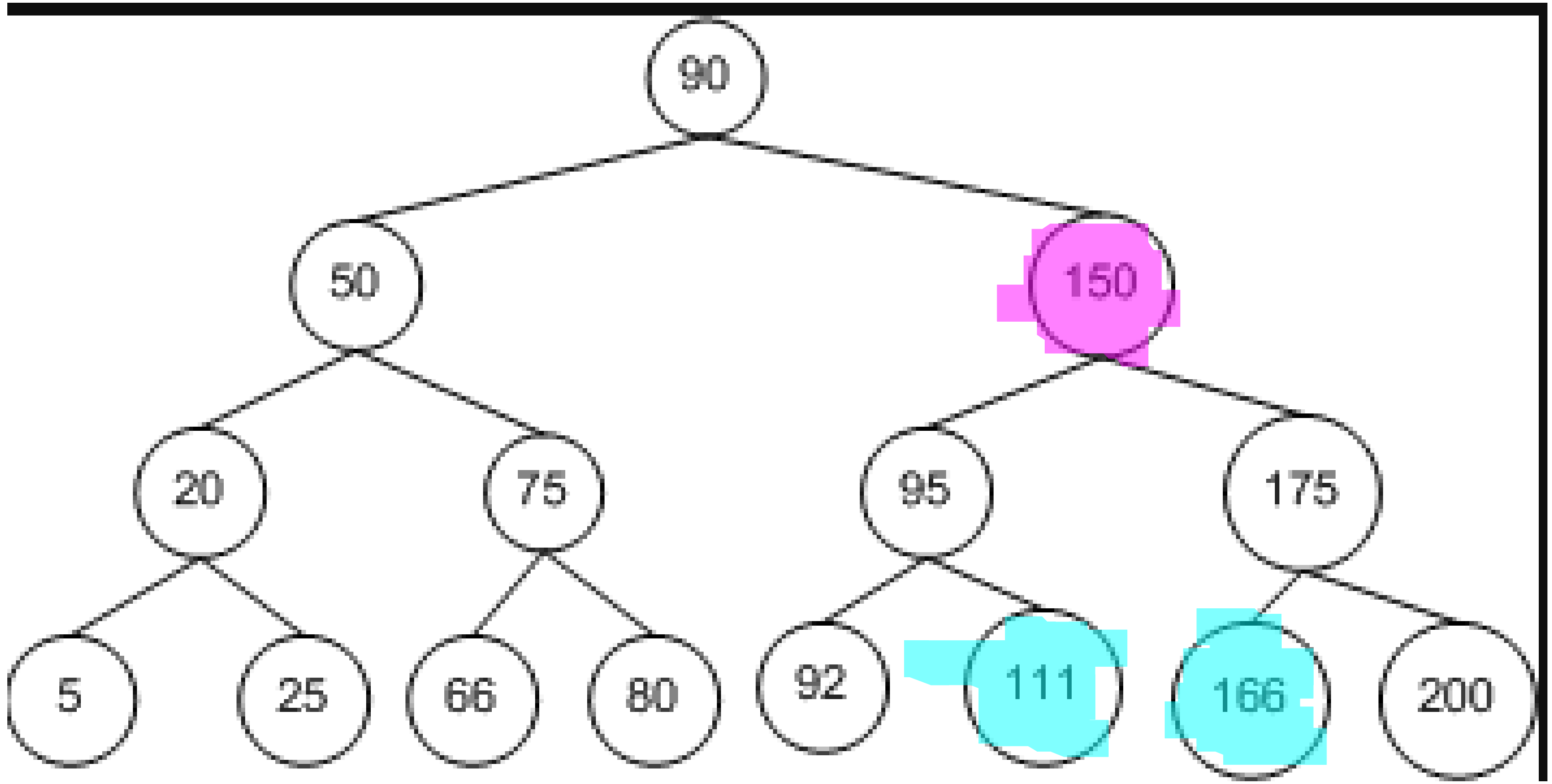
INSERT

```
Void Insert(int pos, int x){
        if(tree[pos] == NULL){
                create_new_node({NULL, NULL, x});
        }
        else{
                if(x < tree[pos].val){
                        Insert(tree[pos].left, x);
                }
                else if (x > tree[pos].val){
                        Insert(tree[pos].right, x);
                }
        }
}
```

REMOVE

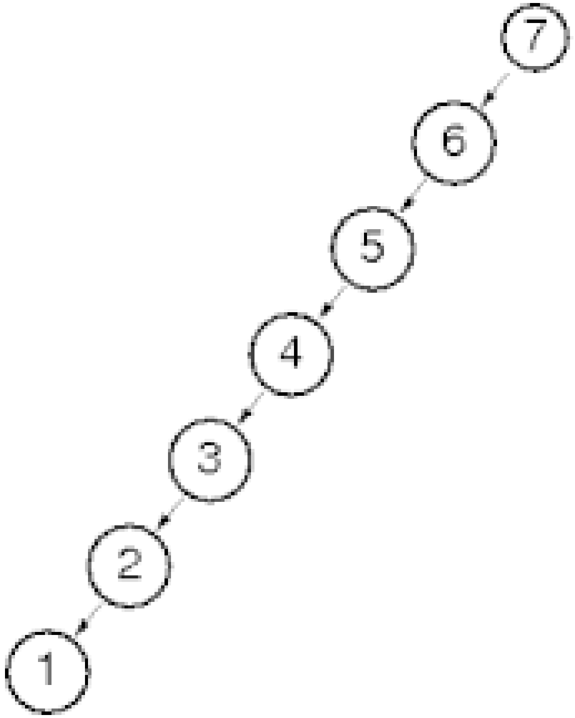
- Same way as Insert, find the node which contains the element.
- If the node is a leaf, directly remove it.
- If it only has on child; replace it with its only child
- Otherwise, replace it with the smallest node in its right subtree or largest node in its left subtree.

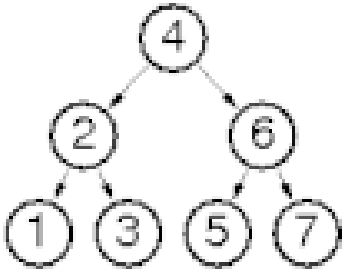
REMOVE



TIME COMPLEXITY (AVERAGE AND WORST)

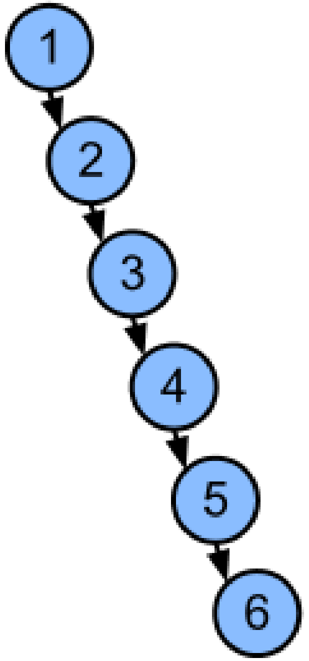
- Average Time Complexity of each operation: O(logn). Only need to search the height of the tree.
- Worst Case: O(n)
- The nodes are in order





Balanced binary tree

Non-balanced



Balanced