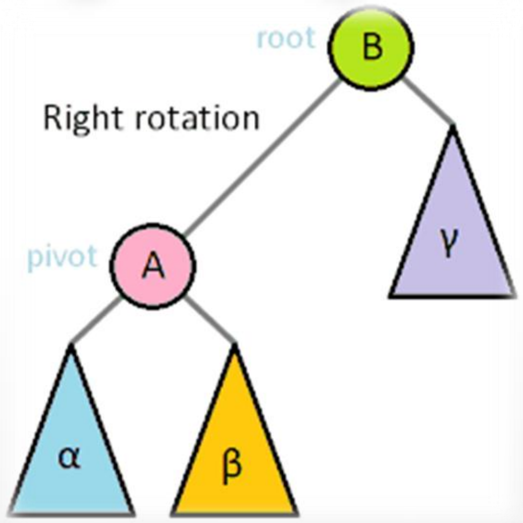
BIRZEIT UNIVERSITY



AVL Trees

Dr. Abdallah Karakra

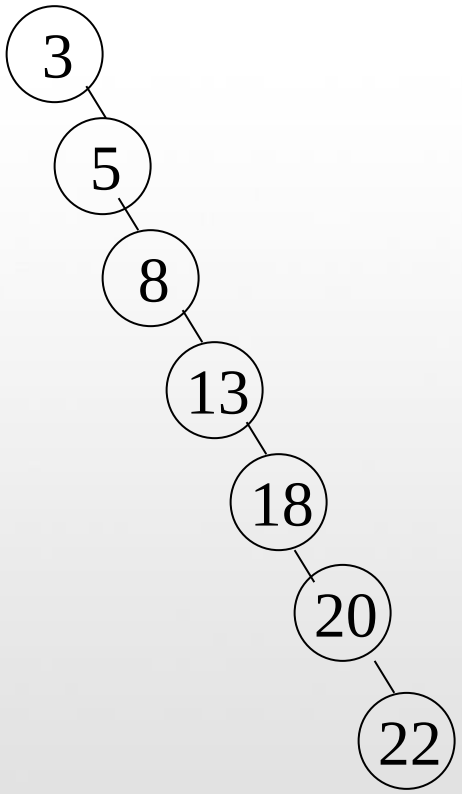
Computer Science Department

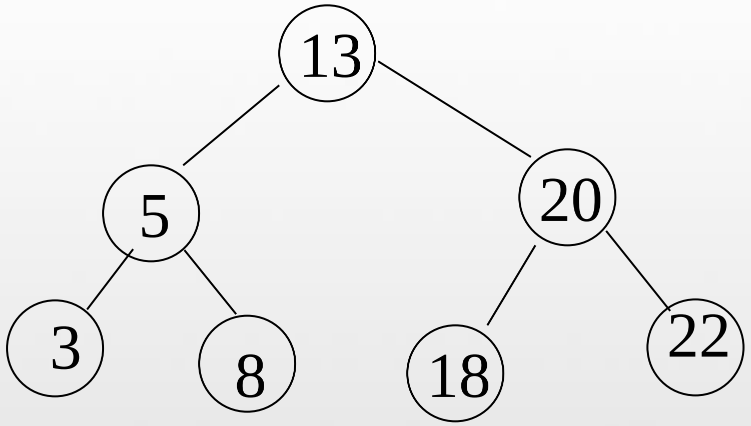
COMP242

Monday, May 29, 2023

Motivation

- When building a binary search tree, what type of trees would we like? Example: 3, 5, 8, 13, 18, 20, 22



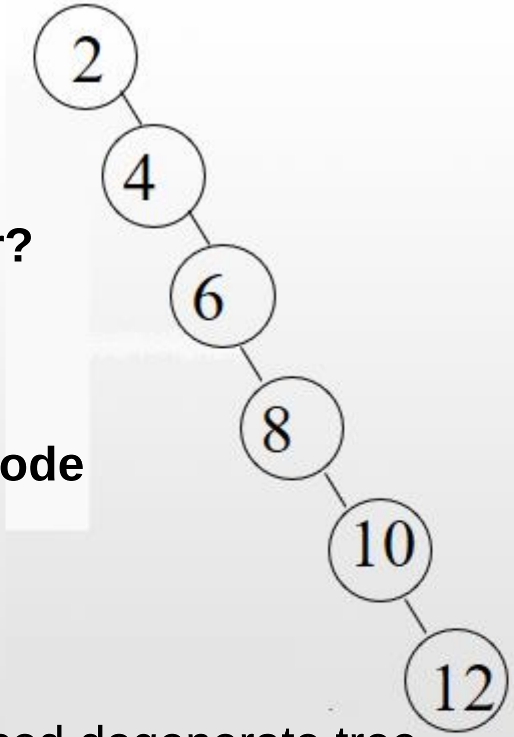


Recall

- ❑ Best case running time of BST operations is O(log N)
- ❑ Worst case running time is O(N)

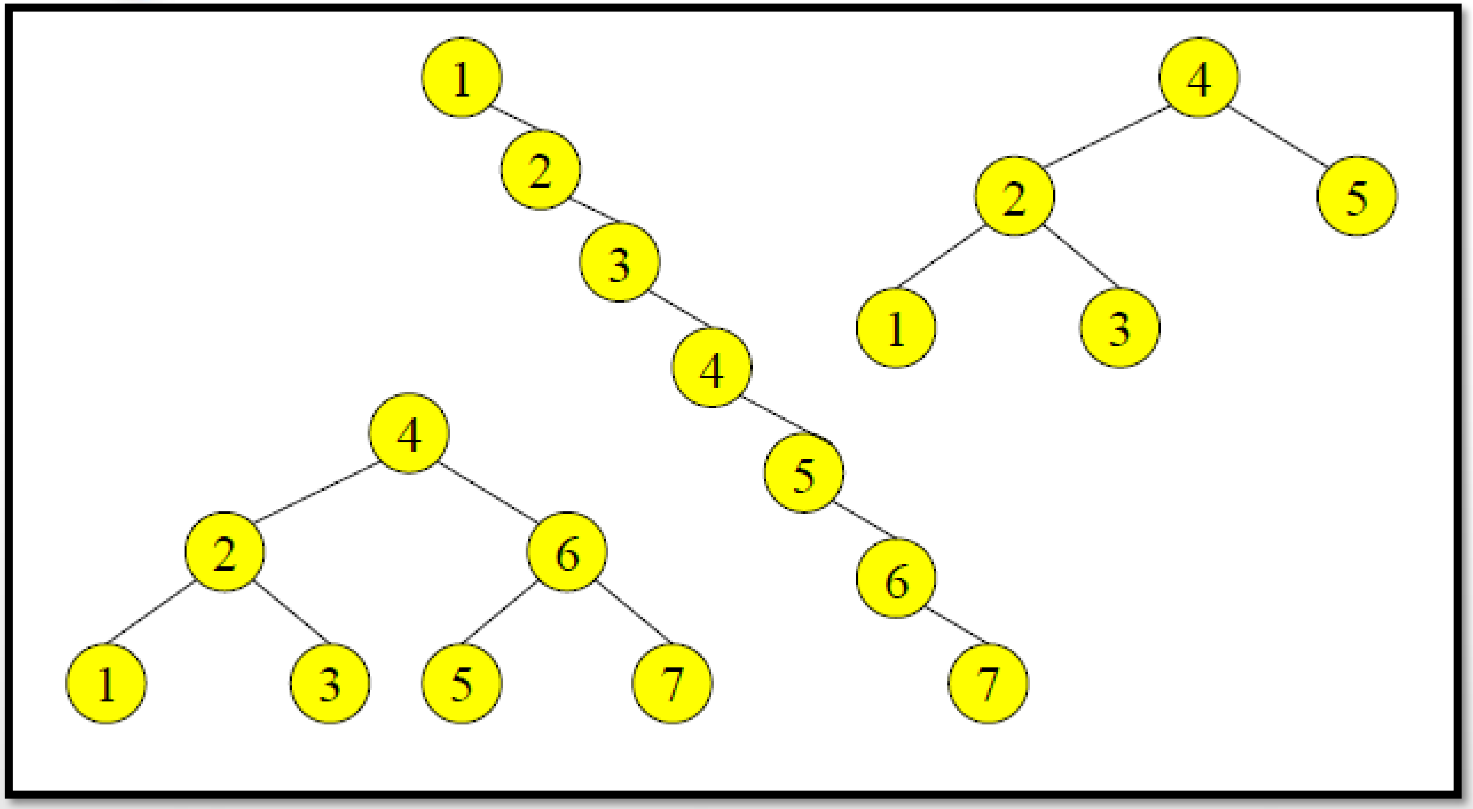
What happens when you Insert elements in ascending order?

- ❖ **Insert: 2, 4, 6, 8, 10, 12 into an empty BST**
- ❖ **Problem: Lack of “balance”**
- ❖ **compare the height of left and right subtrees for each node**



Unbalanced degenerate tree

Balanced and unbalanced BST



❖ **If the BST is not balanced**

May end up with some nodes very deep

AVL Trees

- Adelson-Velskii and Landis
- AVL trees are height-balanced binary search trees

- **Balance factor** of a node

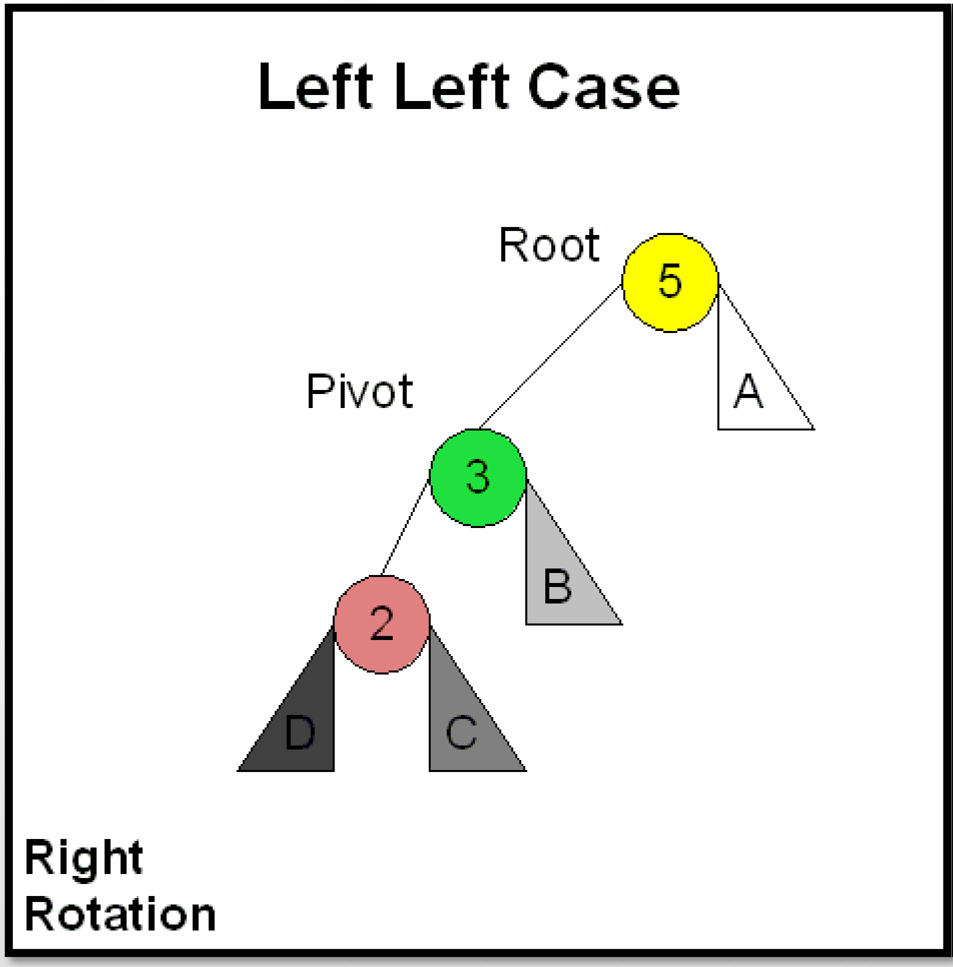
  |Height(left subtree) - Height(right subtree)|

- An AVL tree has balance factor calculated at every node

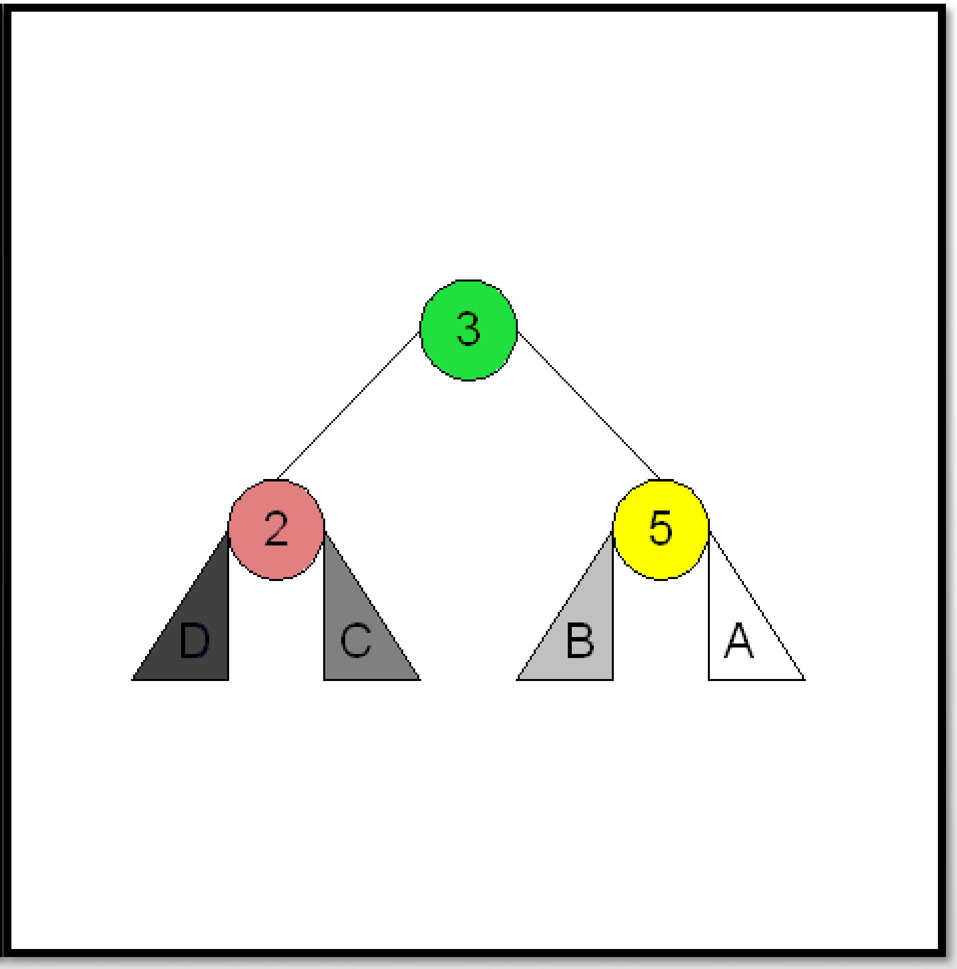
- For every node, heights of left and right subtree can differ by no more than 1
  - |height (left) – height (right) | ≤ 1

- Store current heights in each node

AVL Tree Rotation :Single Rotation

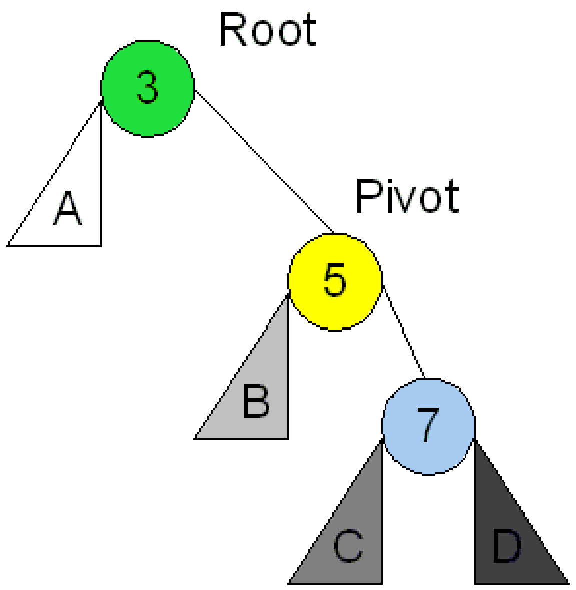


Right Rotation

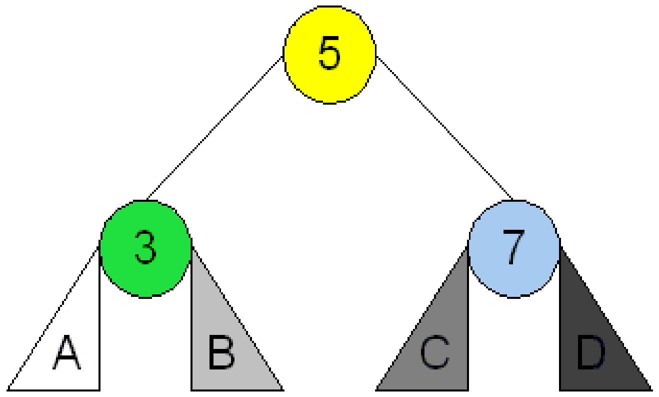


AVL Tree Rotation :Single Rotation

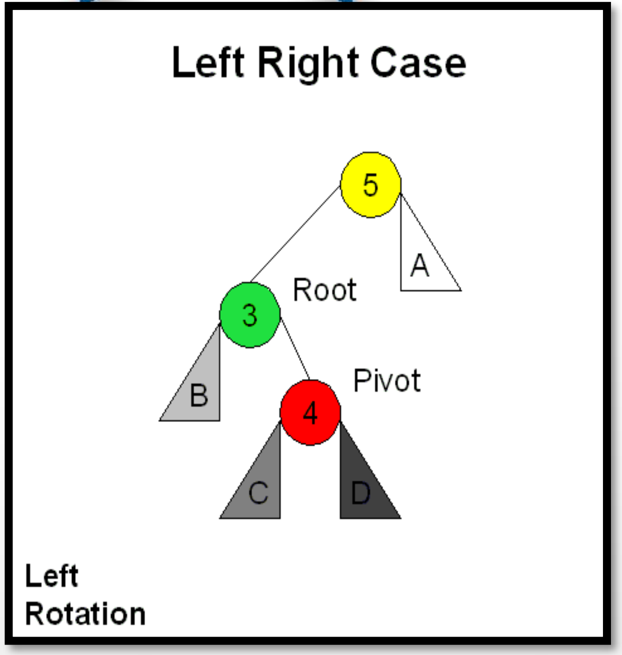
Right Right Case

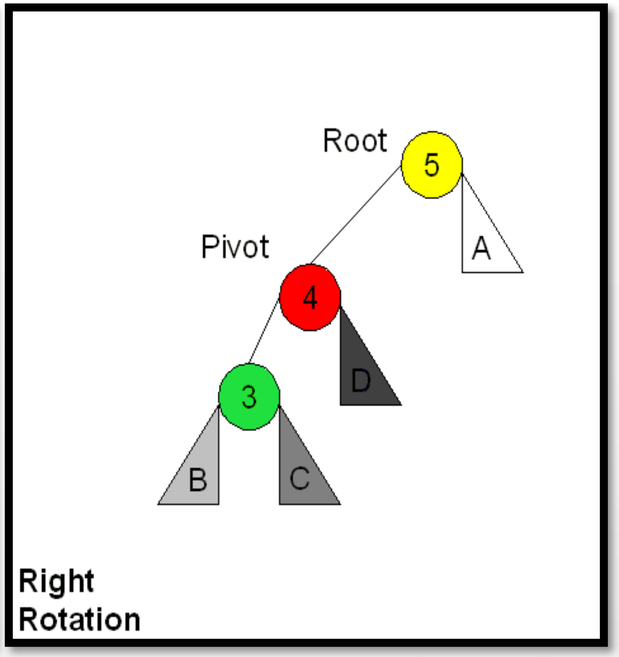


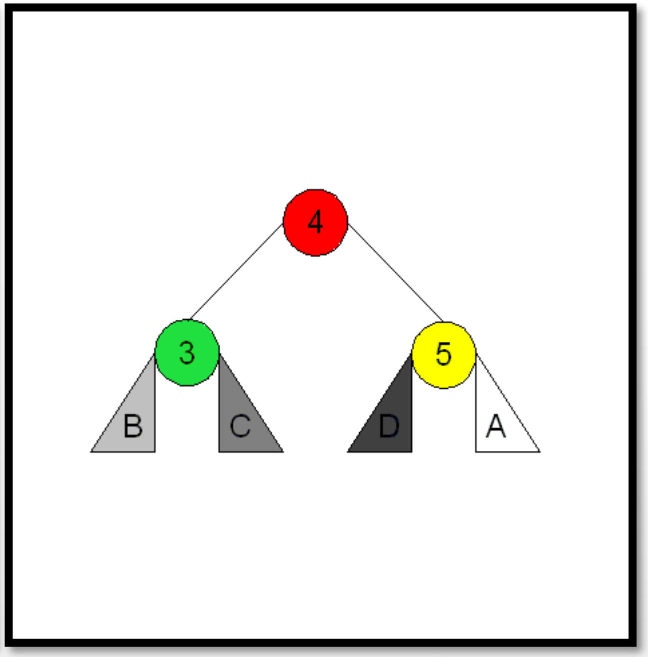
Left Rotation



AVL Tree Rotation :Double Rotation



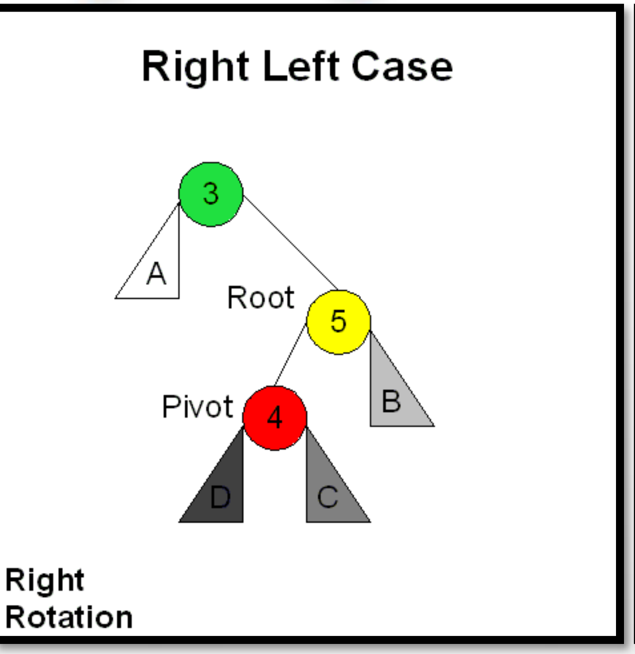


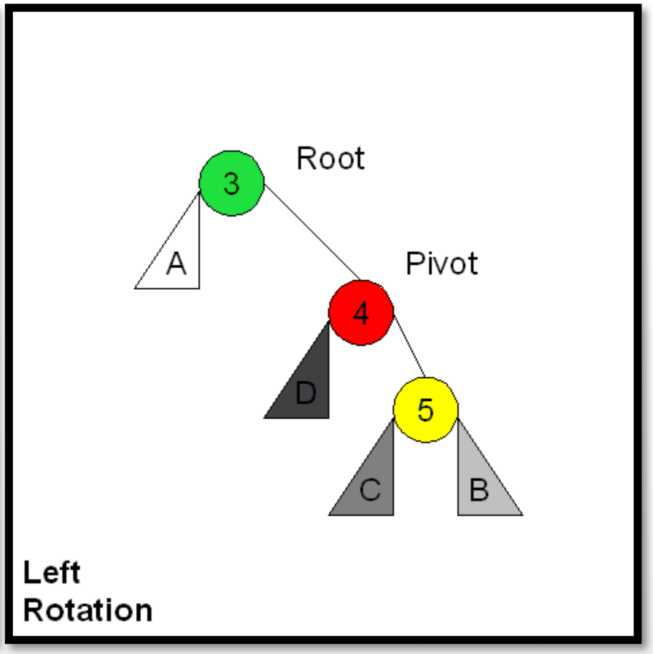


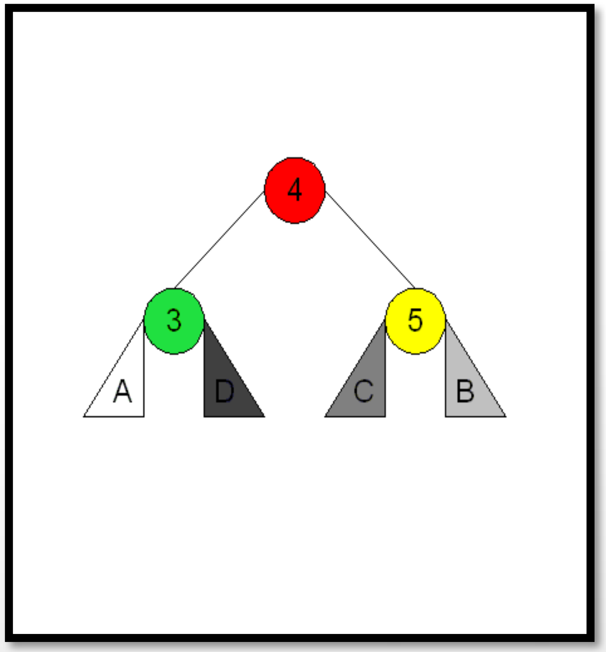
1

2

AVL Tree Rotation :Double Rotation







1

2

AVL Tree Examples

- **Insert 14, 17, 11, 7, 53, 4, 13 into an empty AVL tree**

- **Build an AVL tree with the following values**
  15, 20, 24, 10, 13, 7, 30, 36, 25

Answers

Implementation : AVL Tree

```
//Class Node for the AVL Tree
public class AVLTreeNode {
  int element;          //store data
  AVLTreeNode left;     // left child
  AVLTreeNode right; //right child
  int height;         //Height

  public AVLTreeNode(int element){       // Constructor
    this(element, null, null);
  }

 public AVLTreeNode(int element, AVLTreeNode left, AVLTreeNode right)
  {
    this.element=element;
    this.left=left;
    this.right=right;
    this.height=0;
  }
}
```

Implementation : AVL Tree

```
// AVL Tree Class

public class AVL {

   private AVLTreeNode root;

   public AVL(){
     root=null;
   }

   /* Methods go here */

 }
```

AVL Tree : Height

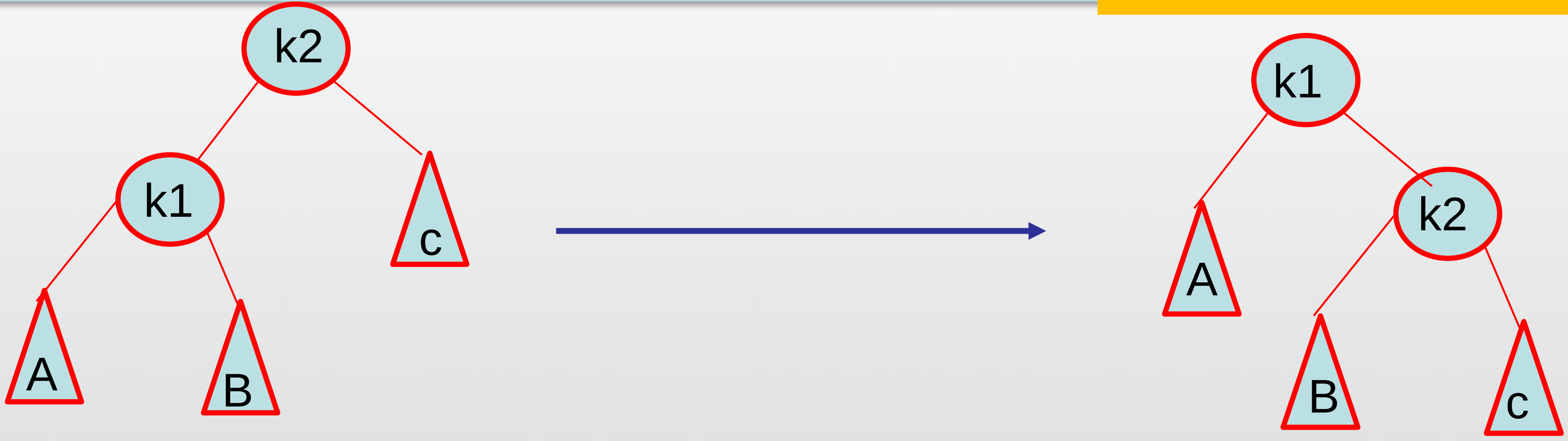
```
// Return the height of a node e
  private int height( AVLTreeNode e ){
   if( e == null )
      return -1;

   return e.height;
}
```

AVL Tree : RotateWithLeftChild

```
// Rotate binary tree node with left child(single rotate to right)
private AVLTreeNode rotateWithLeftChild(AVLTreeNode k2){
 AVLTreeNode k1 = k2.left;
 k2.left = k1.right;
 k1.right = k2;
 k2.height = Math.max(height(k2.left),height( k2.right ))+ 1;
 k1.height = Math.max(height(k1.left),k2.height )+ 1;
 return k1;
}
```

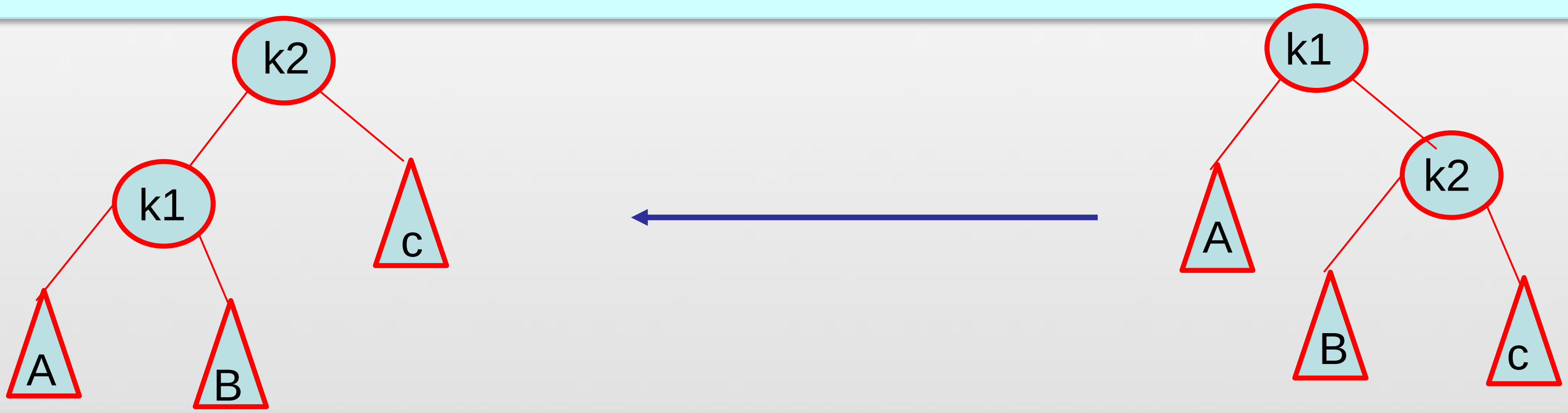
Why Add 1 ?

H.W ???



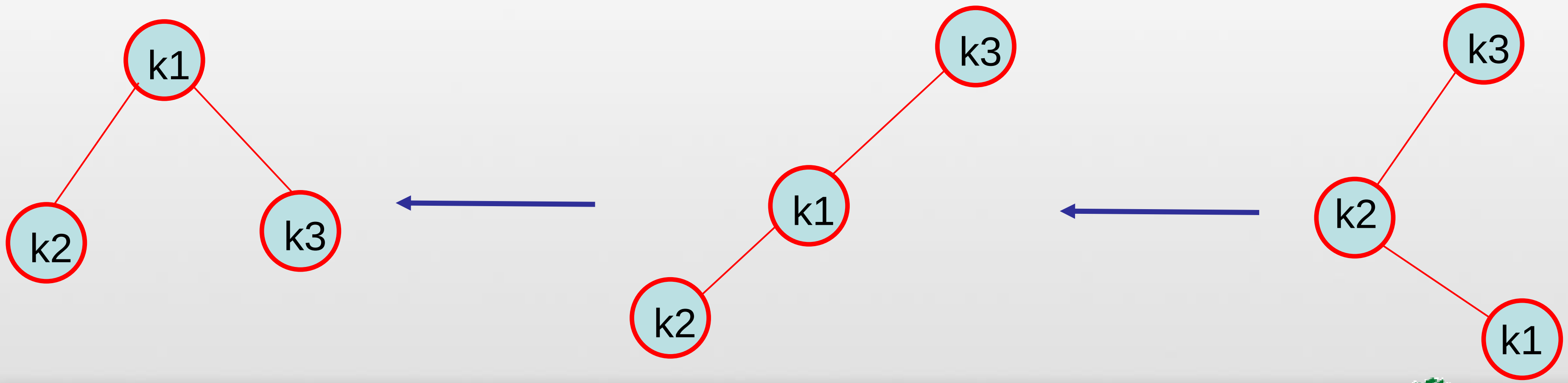
AVL Tree : RotateWithRightChild

```
/ Rotate binary tree node with right child (single rotate to left)

rivate AVLTreeNode rotateWithRightChild(AVLTreeNode k1){
AVLTreeNode k2 = k1.right;
k1.right = k2.left;
k2.left = k1;
k1.height = Math.max(height(k1.left),height( k1.right ))+ 1;
k2.height = Math.max(height(k2.right),k1.height )+ 1;
return k2;
```



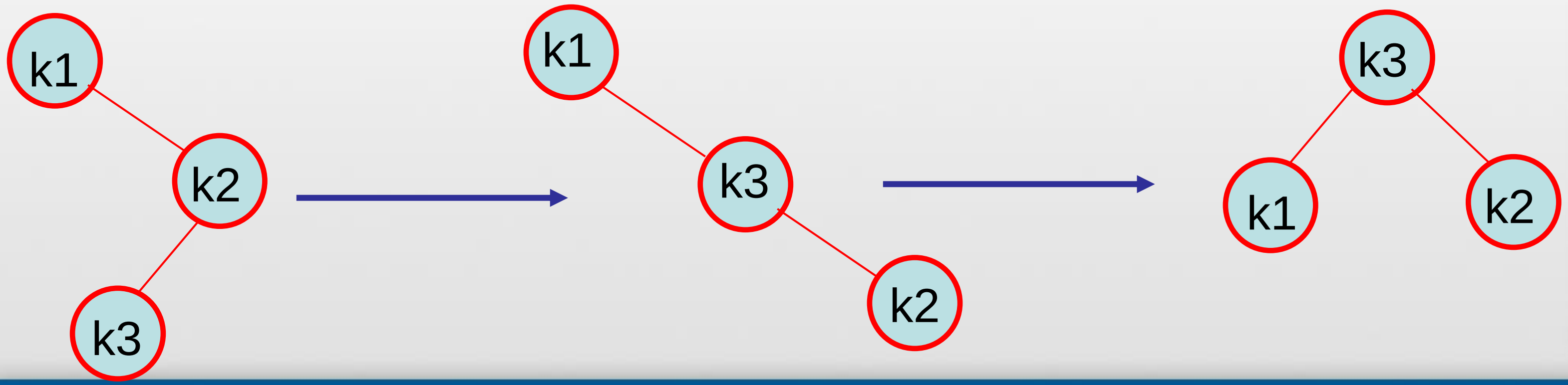
AVL Tree : DoubleWithLeftChild

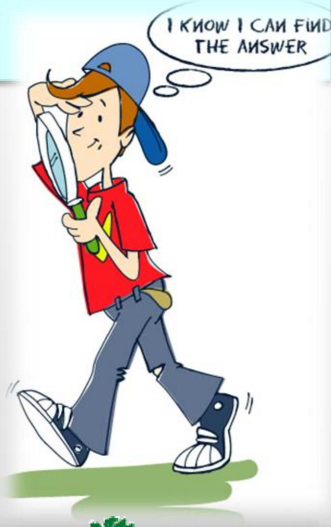
```
/* Double rotate binary tree node: first left child with its
    right child; then node k3 with new left child */

private AVLTreeNode DoubleWithLeftChild(AVLTreeNode k3){
    k3.left = rotateWithRightChild( k3.left );
    return rotateWithLeftChild( k3 );
}
```



AVL Tree : DoubleWithRightChild : H.W

```
/*Double rotate binary tree node: first right child
  with its left child; then node k1 with new right child */

private AVLTreeNode DoubleWithRightChild(AVLTreeNode k1){

      k1.right = rotateWithLeftChild( k1.right );
      return rotateWithRightChild( k1 );

}
```





Extra Exercises

Question 1
Lab 8: AVL Tree

Question?

“Success is the sum of small efforts, repeated day in and day out.”
Robert Collier