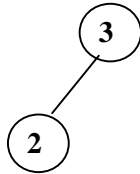


## AVL Trees Example

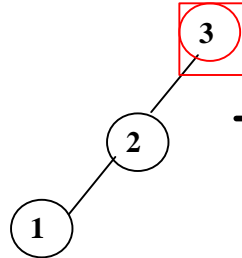
**Insert 3**



**Insert 2**

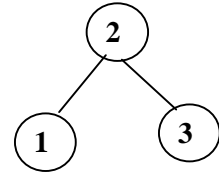


**Insert 1 (non-AVL)**

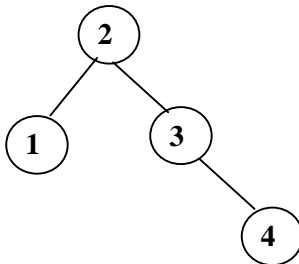


Single rotation

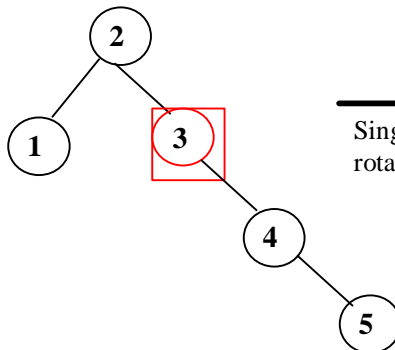
**AVL**



**Insert 4**

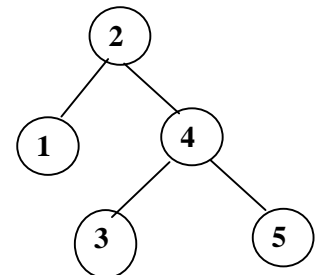


**Insert 5 (non-AVL)**

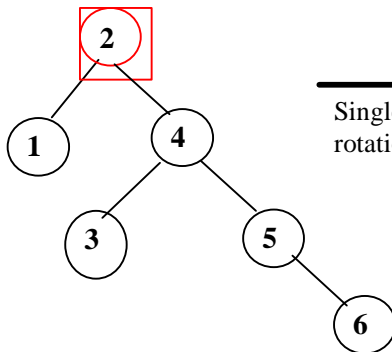


Single rotation

**AVL**

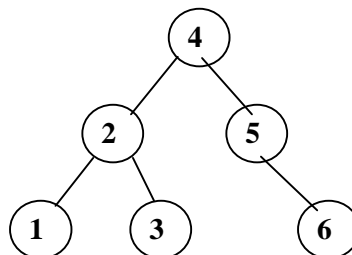


**Insert 6 (non-AVL)**

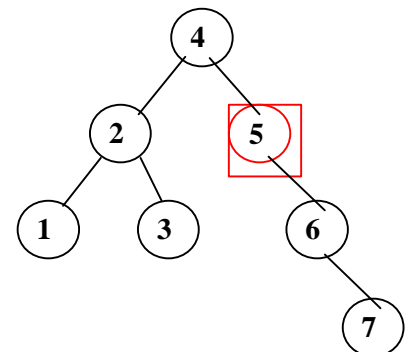


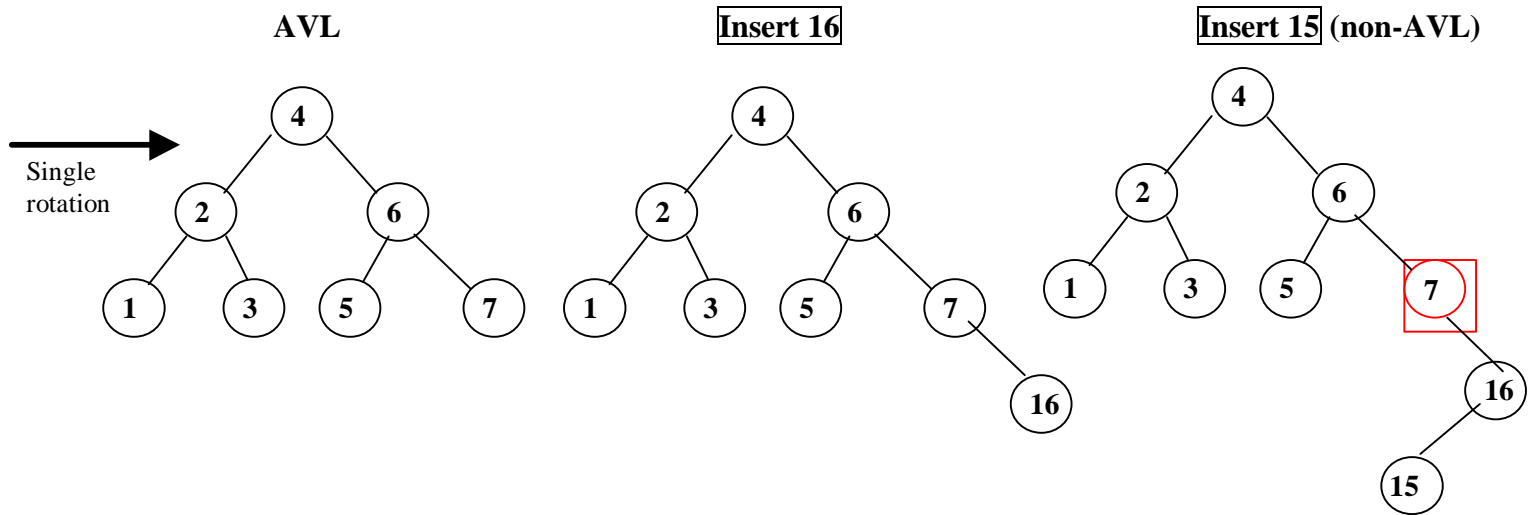
Single rotation

**AVL**



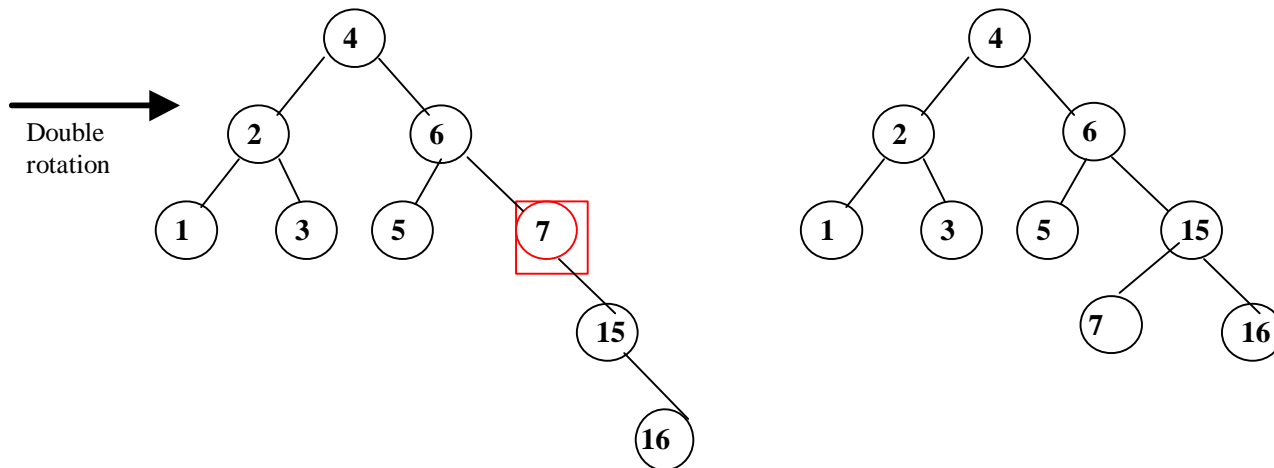
**Insert 7 (non-AVL)**





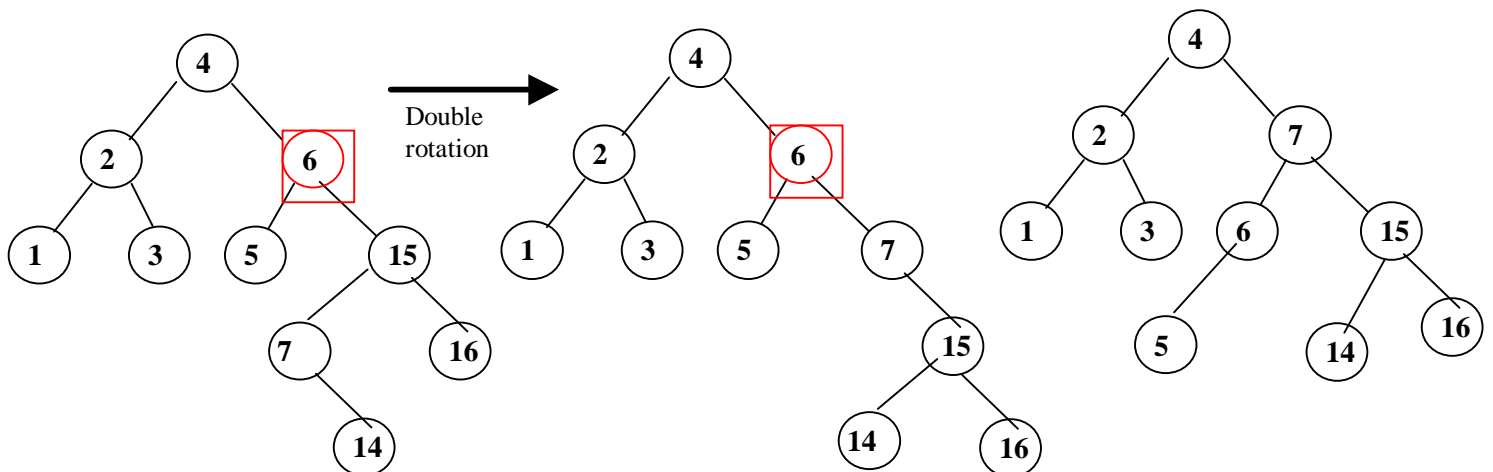
Step 1: Rotate child and grandchild

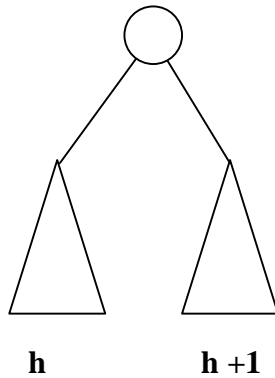
Step 2: Rotate node and new child (AVL)

**Insert 14 (non-AVL)**

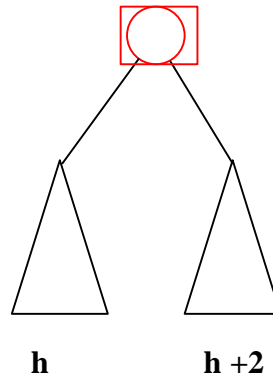
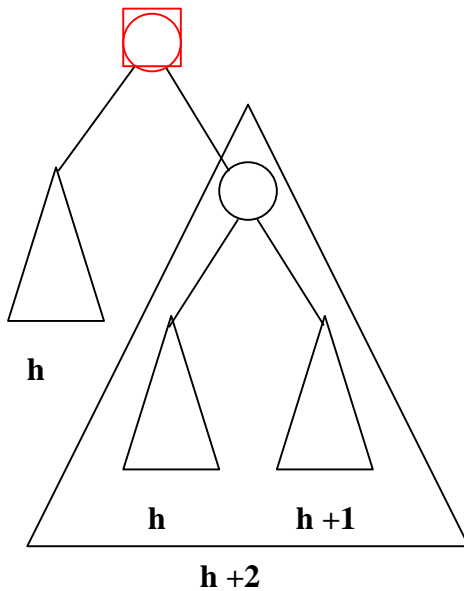
Step 1: Rotate child and grandchild

Step 2: Rotate node and new child (AVL)

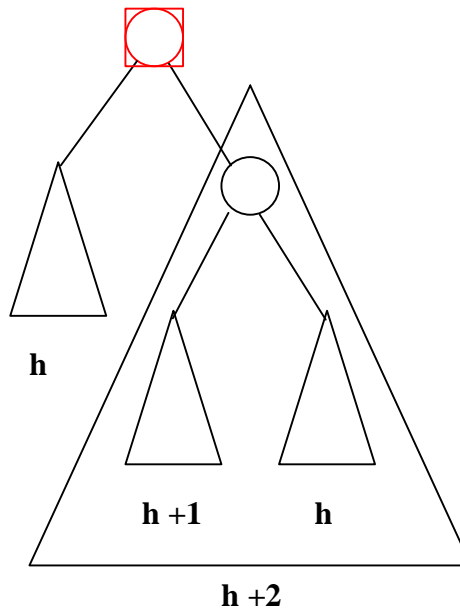


**AVL Tree**

→  
Insert a node

**No longer AVL, must rebalance****Two ways to expand subtree of height  $h+2$ :**

Apply a Single Rotation



Apply a Double Rotation

**Note:** the symmetrical case is handled identically (i.e. mirror image)

Single rotation

Single rotation

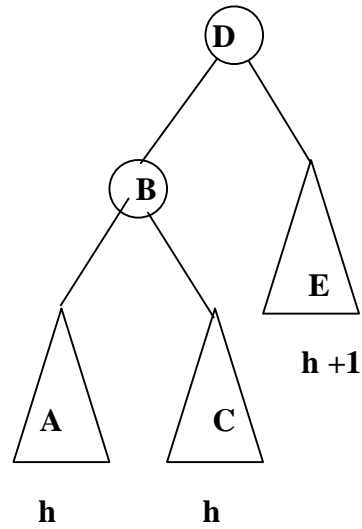


Diagram illustrating a double rotation operation in a binary tree structure. The diagram shows a large triangle containing several smaller triangles and nodes labeled A, B, C, D, E, F, G. Node B is at the top, highlighted with a red square. Node D is in the center. Node F is to the right of D. Node A is to the left of D. Node C is below A. Node E is below D. Node G is to the right of E. An arrow labeled "Double rotation" points to the right. Below the triangles, the text  $h+1$  is centered, and  $h+2$  is centered below that. The text "OR" appears twice, once between  $h+1$  and  $h$ , and once between  $h$  and  $h$ .

Double rotation

