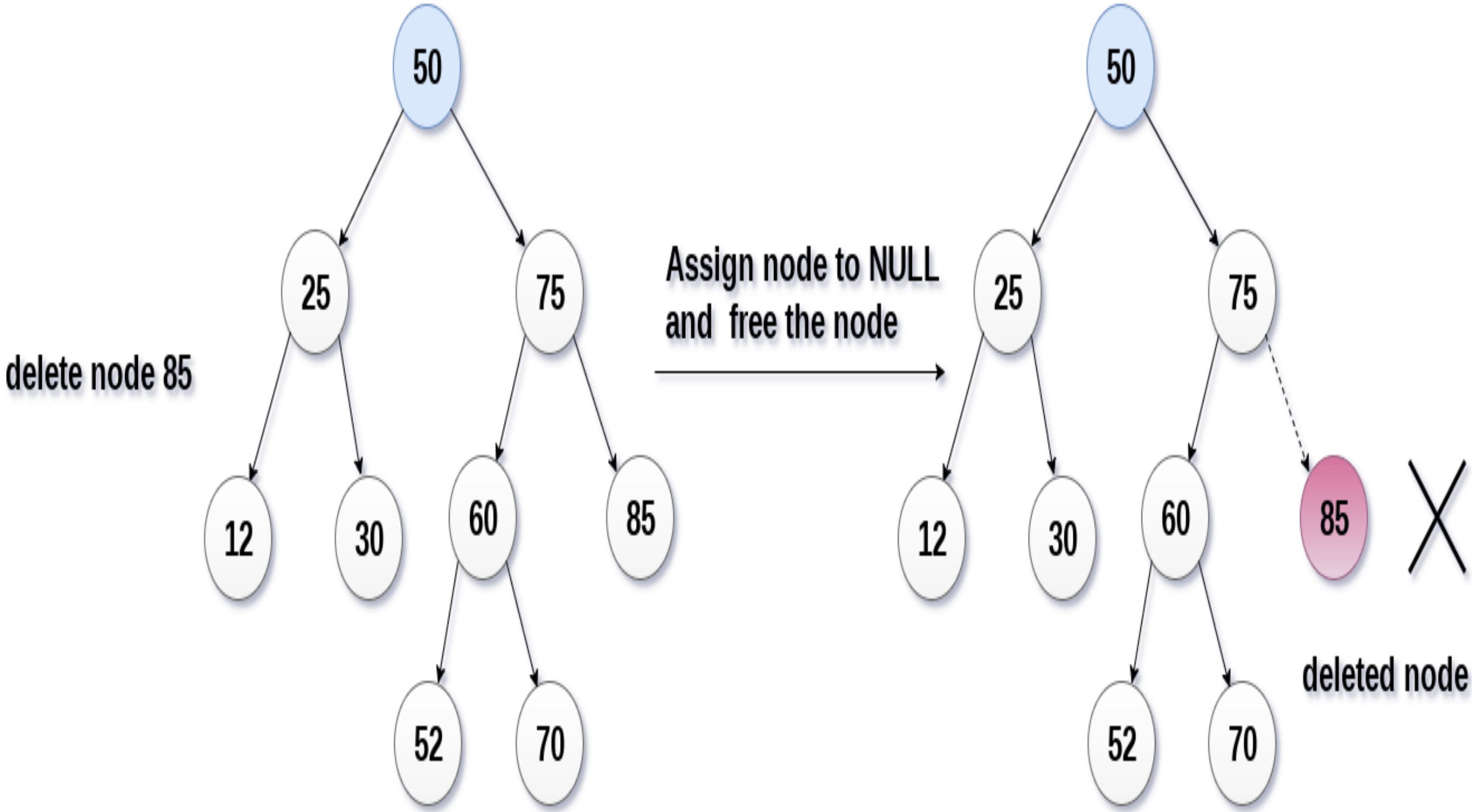
AVL Trees

By
Aditya Tiwari
Assistant Professor
CSVTU Bhilai

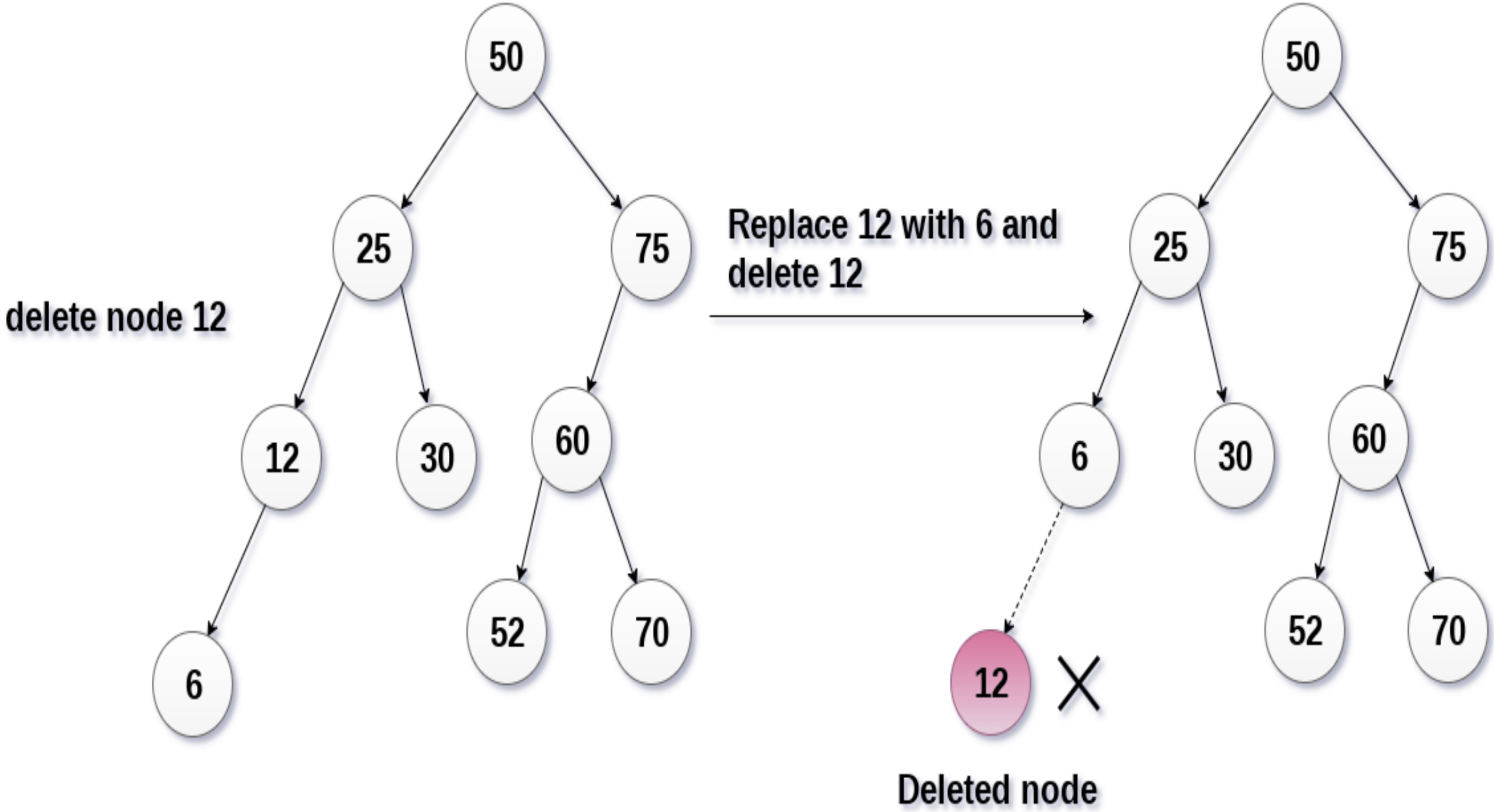
AVL Deletion

- Understand the concept of BINARY SEARCH TREES deletion.
- Delete function is used to delete the specified node from a binary search tree. However, we must delete a node from a binary search tree in such a way, that the property of binary search tree doesn't violate. There are three situations of deleting a node from binary search tree.

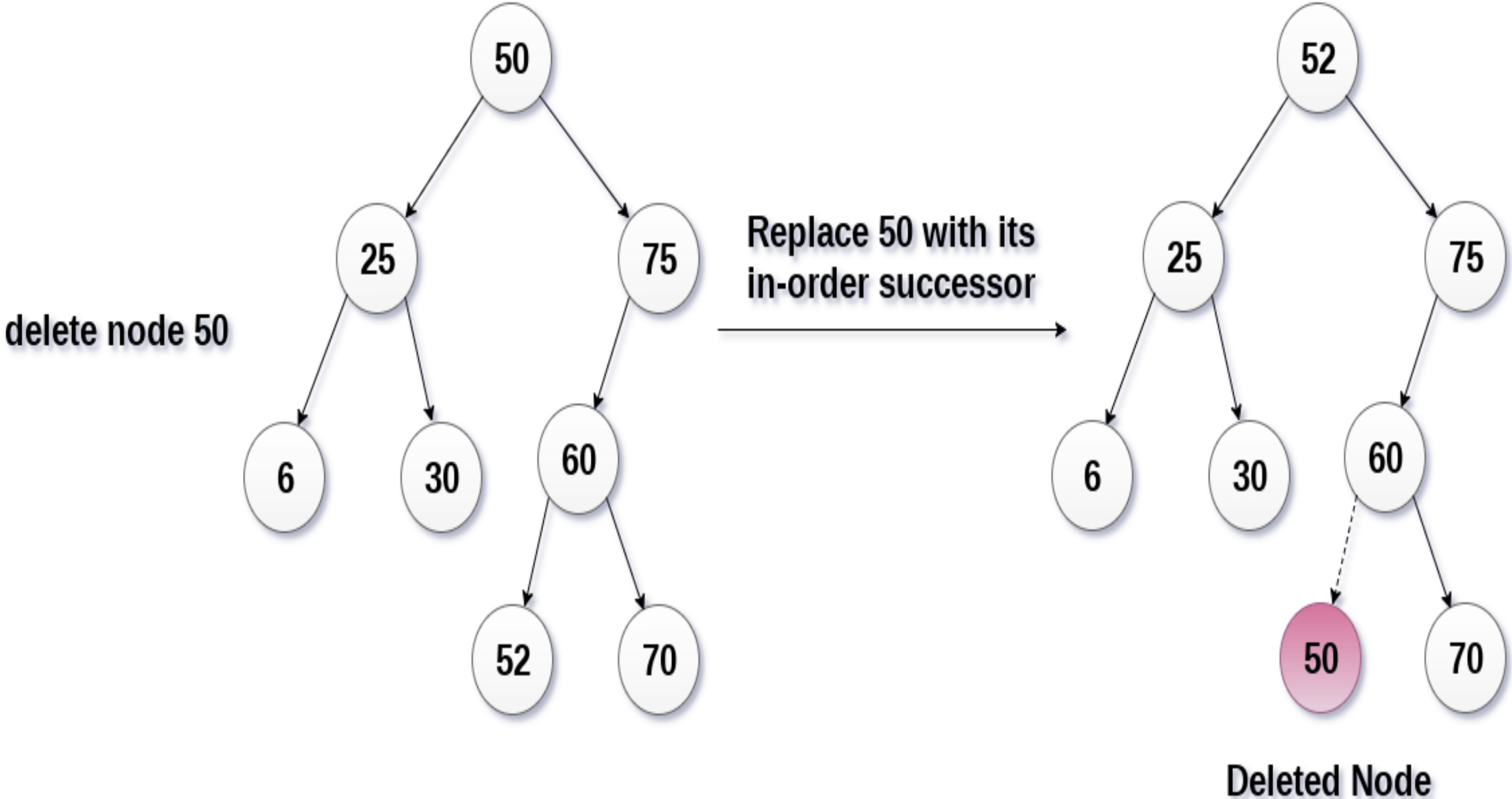
- The node to be deleted is a leaf node
- It is the simplest case; in this case, replace the leaf node with the NULL and simple free the allocated space.
- In the following image, we are deleting the node 85, since the node is a leaf node, therefore the node will be replaced with NULL and allocated space will be freed.


delete node 85
50
25
75
12
30
60
85
52
70
Assign node to NULL
and free the node
50
25
75
12
30
60
85
52
70
deleted node

- The node to be deleted has only one child.
- In this case, replace the node with its child and delete the child node, which now contains the value which is to be deleted. Simply replace it with the NULL and free the allocated space.
- In the following image, the node 12 is to be deleted. It has only one child. The node will be replaced with its child node and the replaced node 12 (which is now leaf node) will simply be deleted.


delete node 12
50
25
75
12
30
60
6
52
70
Replace 12 with 6 and
delete 12
50
25
75
6
30
60
12
52
70
Deleted node

- he node to be deleted has two children.
- It is a bit complexes case compares to other two cases. However, the node which is to be deleted is replaced with its in-order successor or predecessor recursively until the node value (to be deleted) is placed on the leaf of the tree. After the procedure, replace the node with NULL and free the allocated space.
- In the following image, the node 50 is to be deleted which is the root node of the tree.
- The in-order traversal of the tree given below.
- 6, 25, 30, 50, 52, 60, 70, 75.
- replace 50 with its in-order successor 52. Now, 50 will be moved to the leaf of the tree, which will simply be deleted.


delete node 50
50
25
75
6
30
60
52
70
Replace 50 with its
in-order successor
52
25
75
6
30
60
50
70
Deleted Node

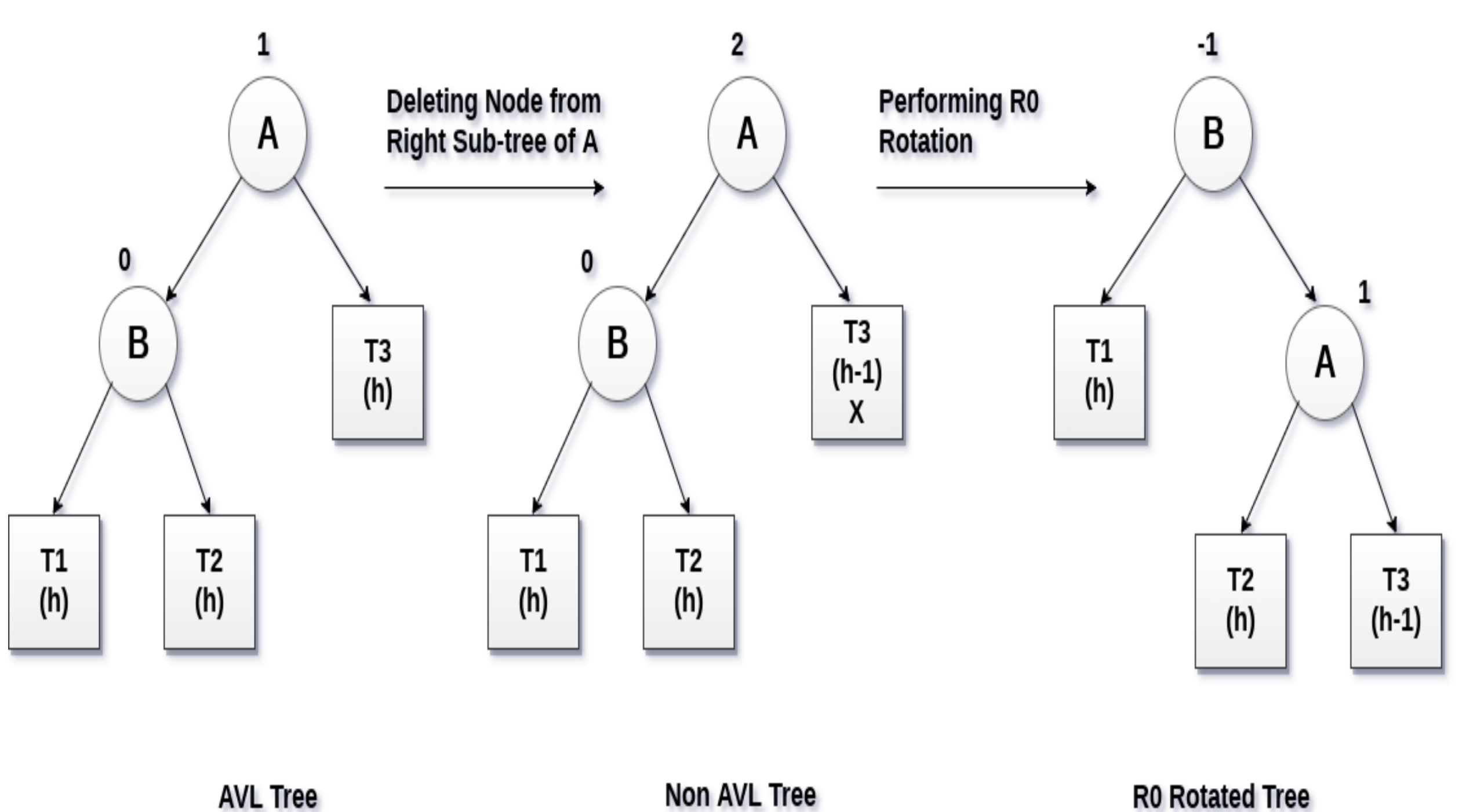
Deletion in AVL Tree

- Deleting a node from an AVL tree is similar to that in a binary search tree. Deletion may disturb the balance factor of an AVL tree and therefore the tree needs to be rebalanced in order to maintain the AVLness. For this purpose, we need to perform rotations. The two types of rotations are L rotation and R rotation. Here, we will discuss R rotations. L rotations are the mirror images of them.
- If the node which is to be deleted is present in the left sub-tree of the critical node, then L rotation needs to be applied else if, the node which is to be deleted is present in the right sub-tree of the critical node, the R rotation will be applied.

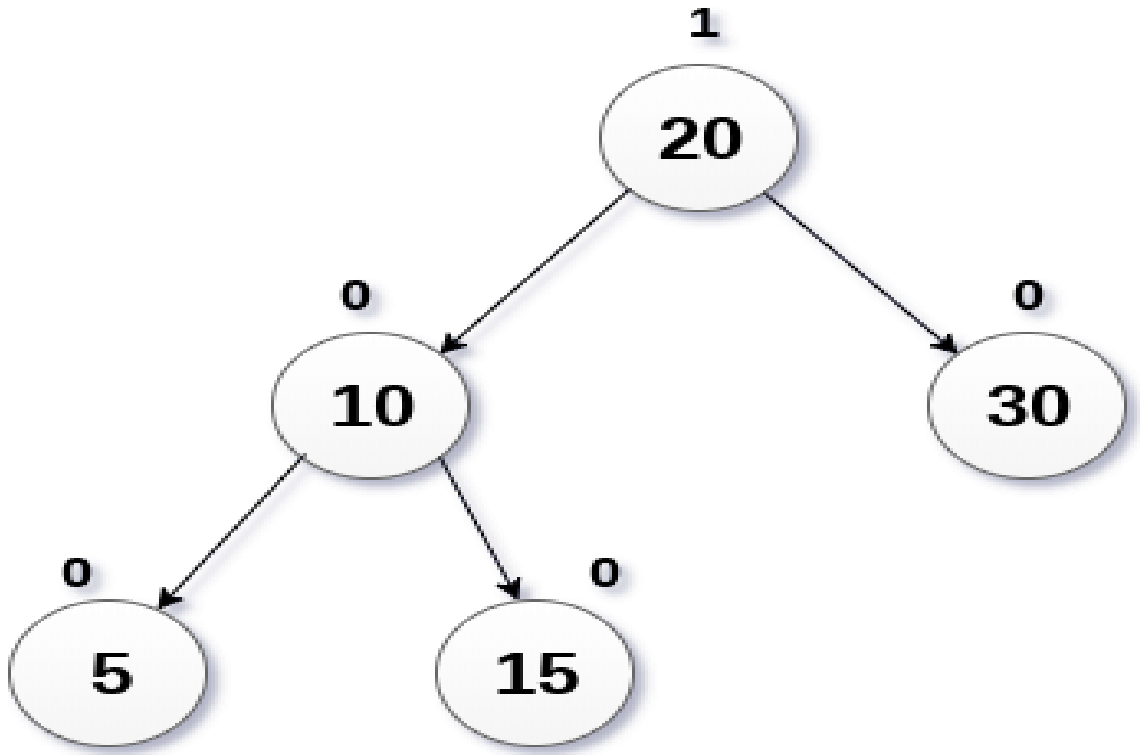
- Let us consider that, A is the critical node and B is the root node of its left sub-tree. If node X, present in the right sub-tree of A, is to be deleted, then there can be three different situations:

R0 rotation (Node B has balance factor 0)

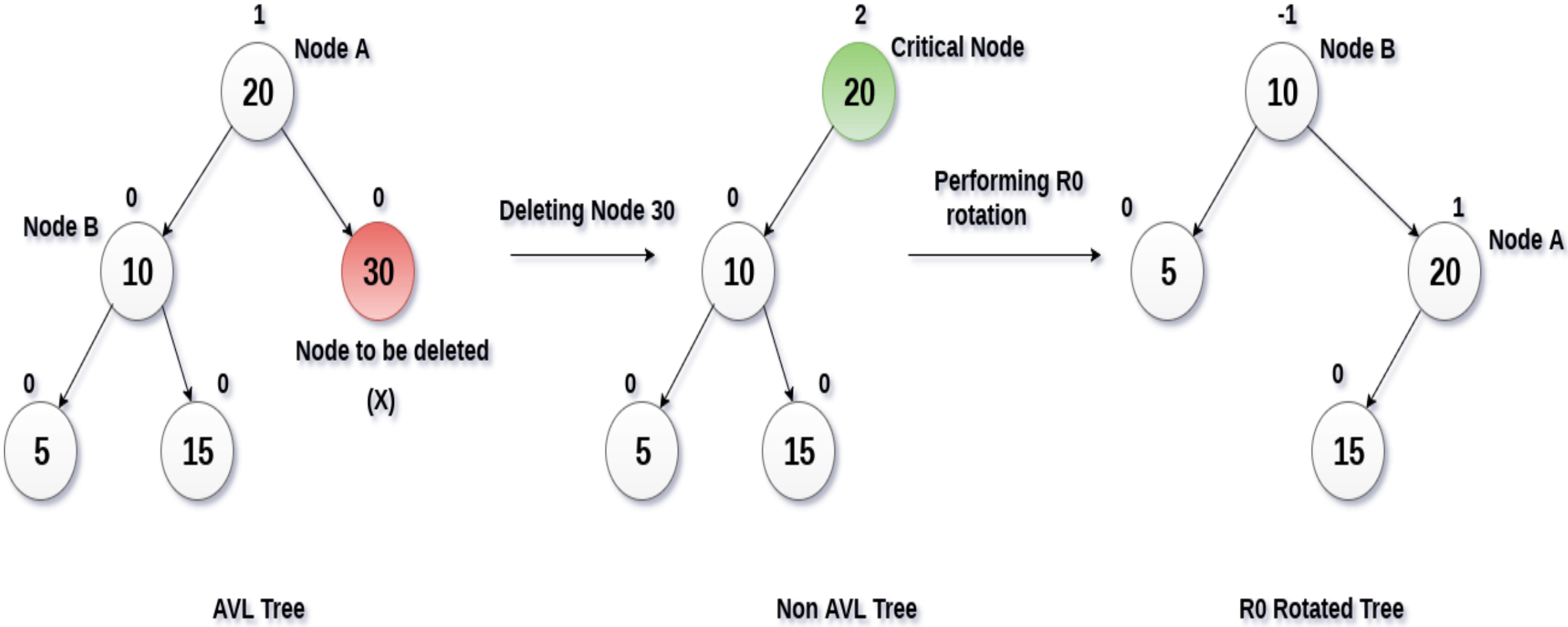
- If the node B has 0 balance factor, and the balance factor of node A disturbed upon deleting the node X, then the tree will be rebalanced by rotating tree using R0 rotation.
- The critical node A is moved to its right and the node B becomes the root of the tree with T1 as its left sub-tree. The sub-trees T2 and T3 becomes the left and right sub-tree of the node A. the process involved in R0 rotation is shown in the following image.


1
A
0
B
T3
(h)
T1
(h)
T2
(h)
Deleting Node from
Right Sub-tree of A
2
A
0
B
T3
(h-1)
X
T1
(h)
T2
(h)
Performing R0
Rotation
-1
B
T1
(h)
1
A
T2
(h)
T3
(h-1)
AVL Tree
Non AVL Tree
R0 Rotated Tree

- Example:
- Delete the node 30 from the AVL tree shown in the following image.

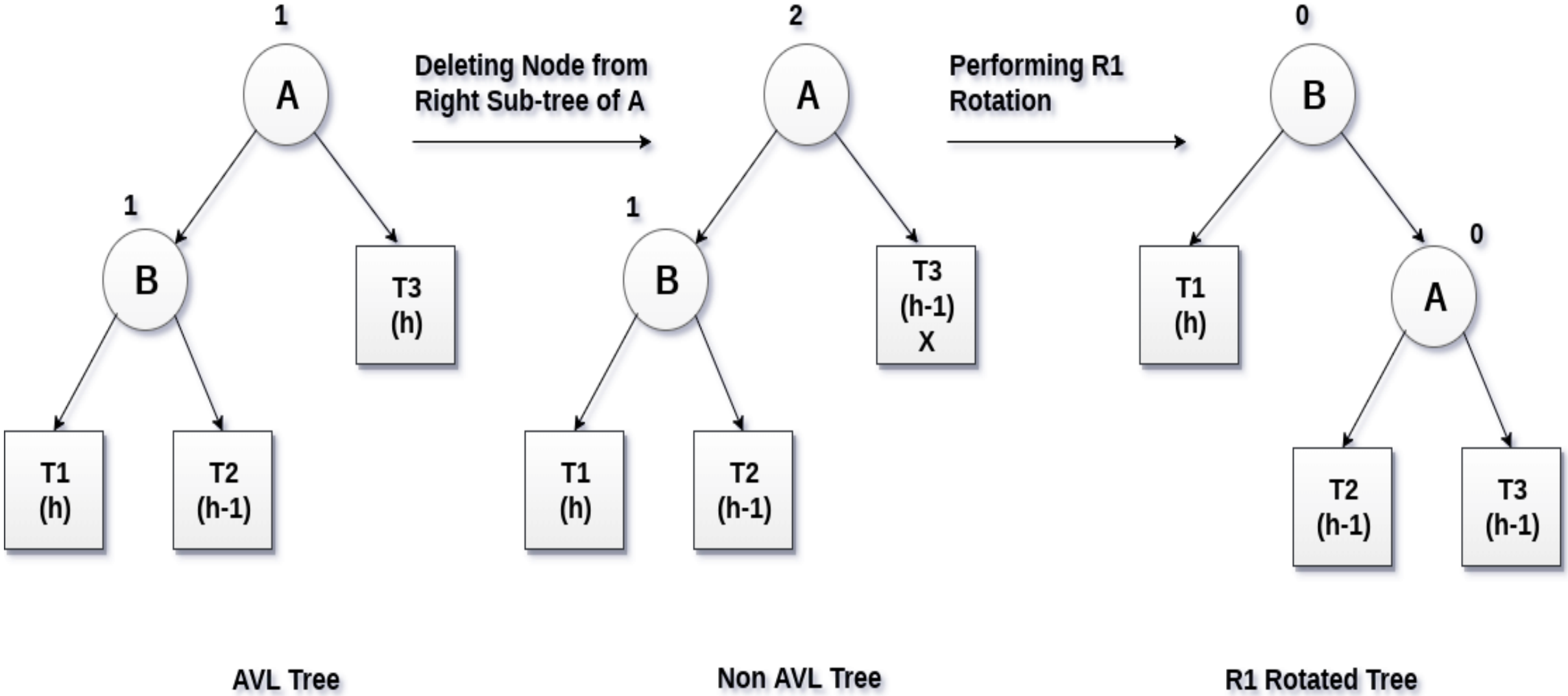


- Solution
- In this case, the node B has balance factor 0, therefore the tree will be rotated by using R0 rotation as shown in the following image. The node B(10) becomes the root, while the node A is moved to its right. The right child of node B will now become the left child of node A.


1
Node A
20
0
Node B
10
0
30
Node to be deleted
(X)
0
5
0
15
AVL Tree
Deleting Node 30
2
Critical Node
20
0
10
0
5
0
15
Non AVL Tree
Performing R0
rotation
-1
Node B
10
0
5
1
Node A
20
0
15
R0 Rotated Tree

R1 Rotation (Node B has balance factor 1)

- R1 Rotation is to be performed if the balance factor of Node B is 1. In R1 rotation, the critical node A is moved to its right having sub-trees T2 and T3 as its left and right child respectively. T1 is to be placed as the left sub-tree of the node B.
- The process involved in R1 rotation is shown in the following image.

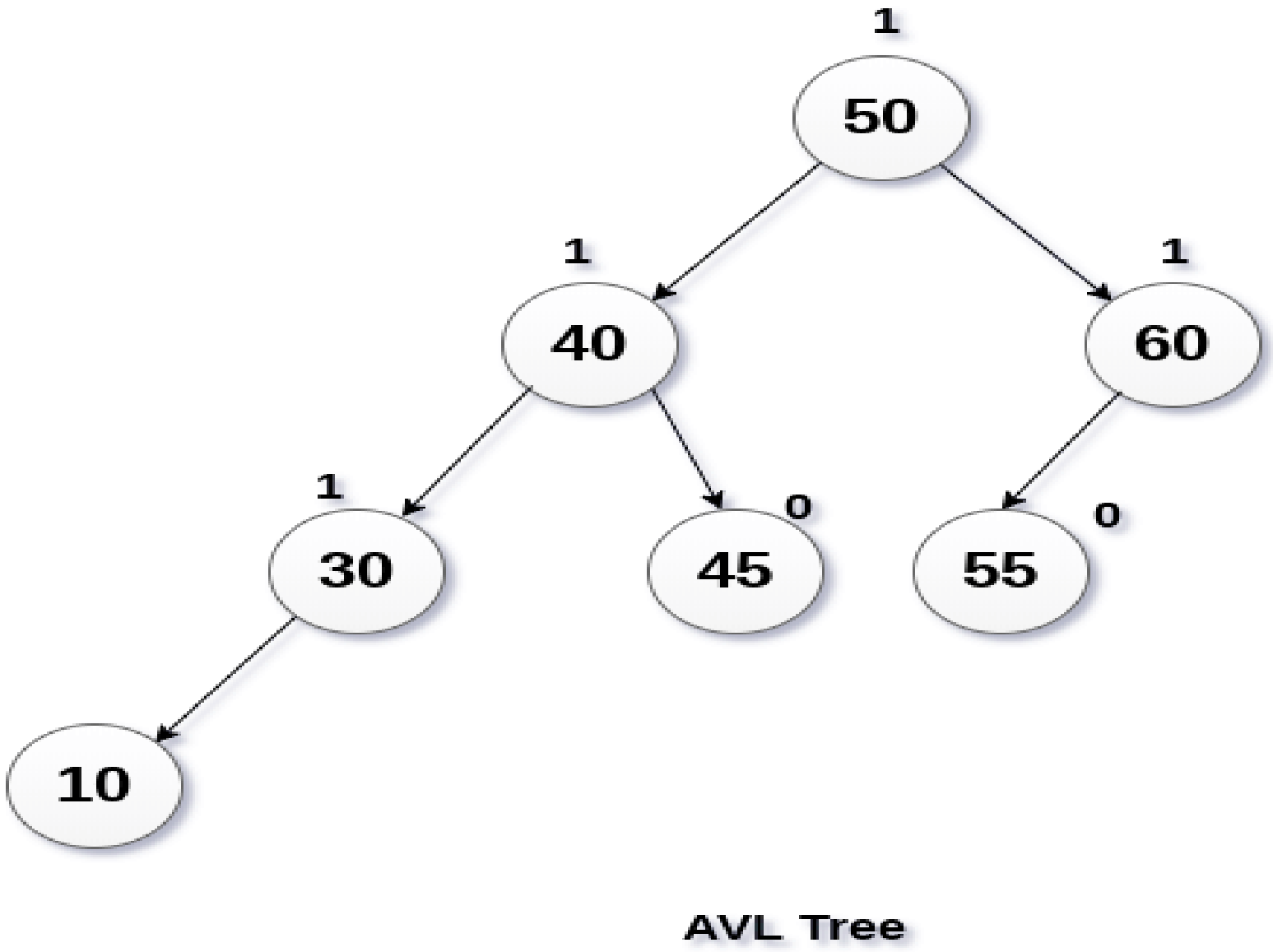


AVL Tree

Non AVL Tree

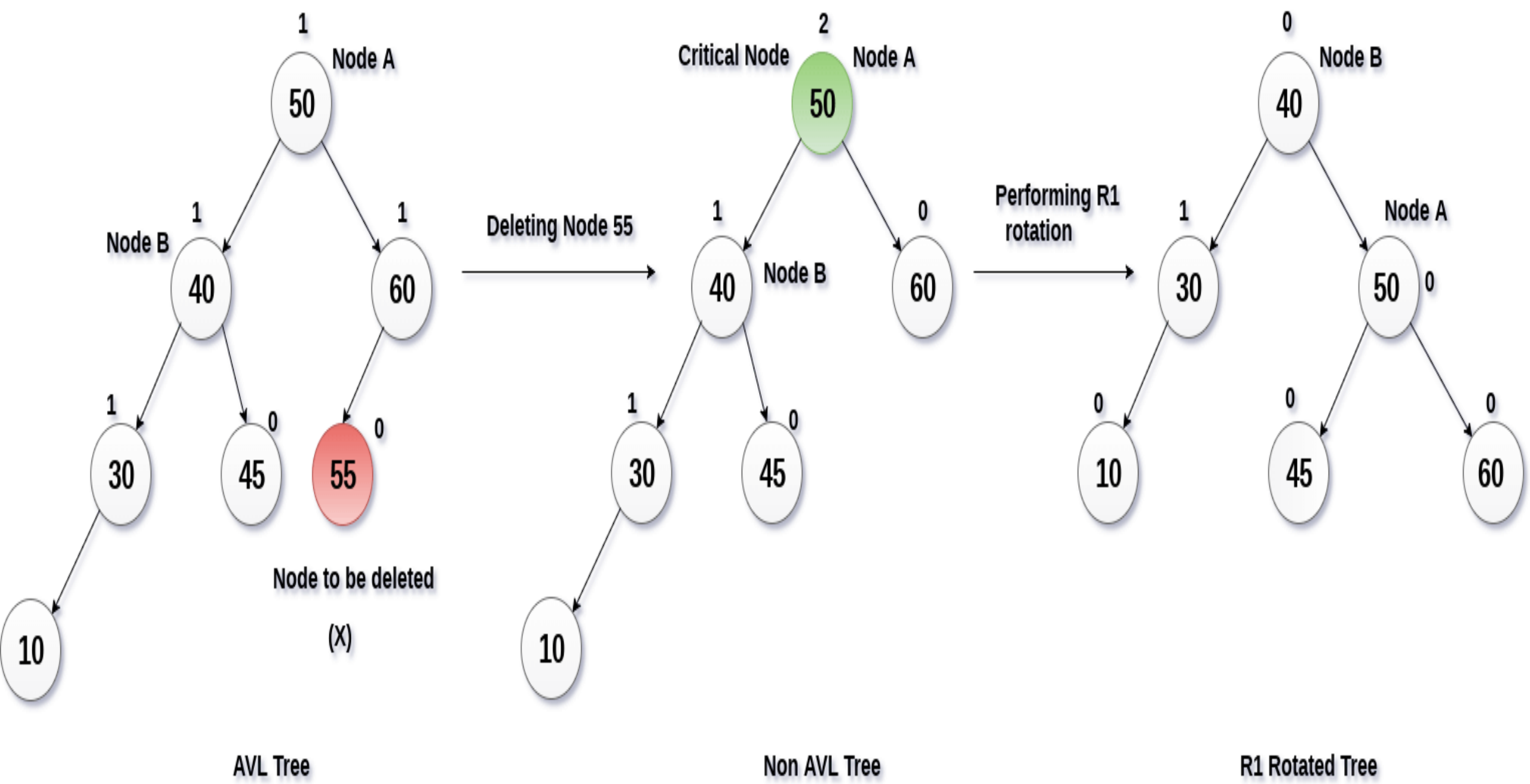
R1 Rotated Tree

Example
Delete Node 55 from the AVL tree shown in the following image.



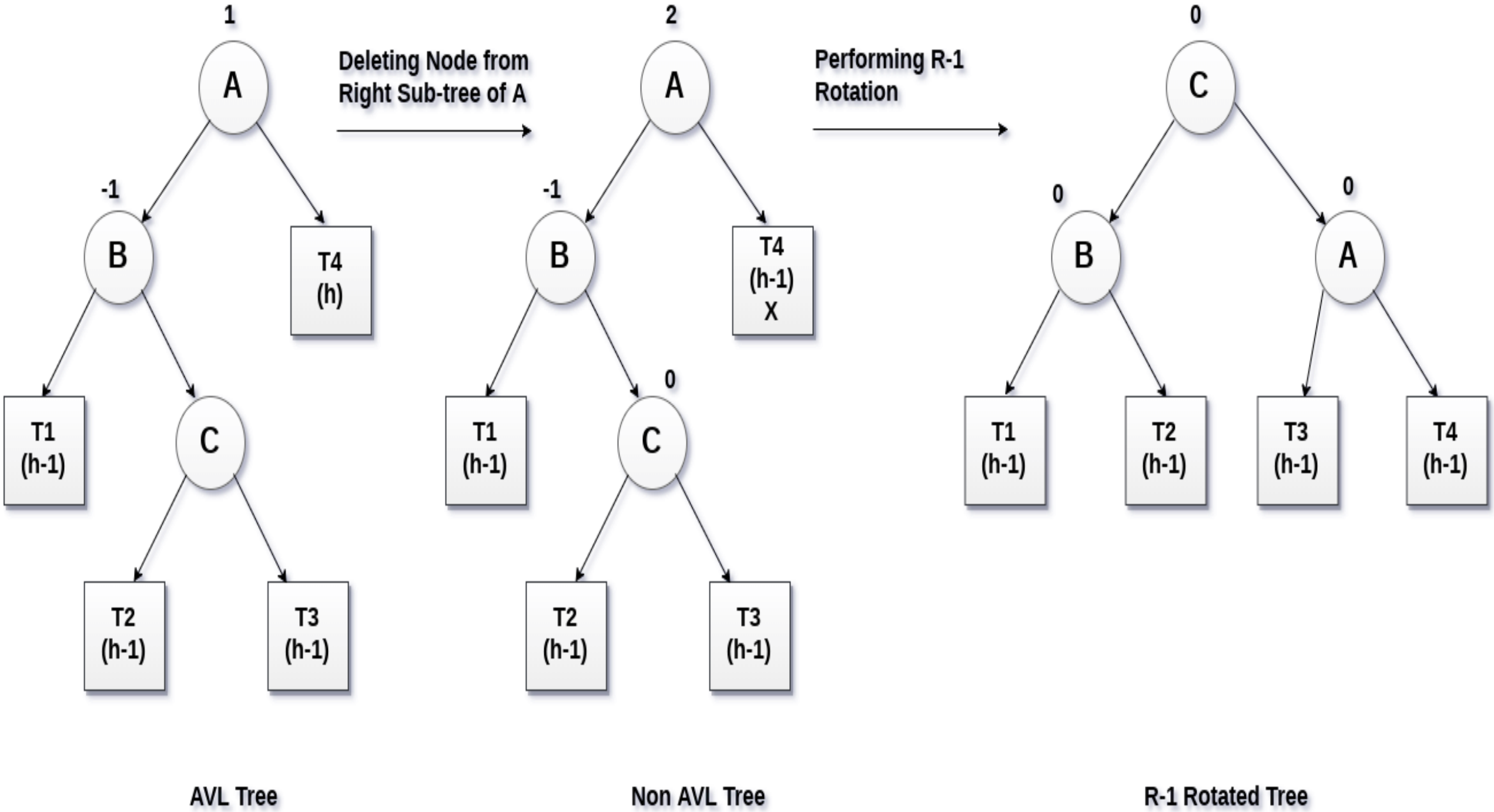
AVL Tree

- Solution :
- Deleting 55 from the AVL Tree disturbs the balance factor of the node 50 i.e. node A which becomes the critical node. This is the condition of R1 rotation in which, the node A will be moved to its right (shown in the image below). The right of B is now become the left of A (i.e. 45).
- The process involved in the solution is shown in the following image.

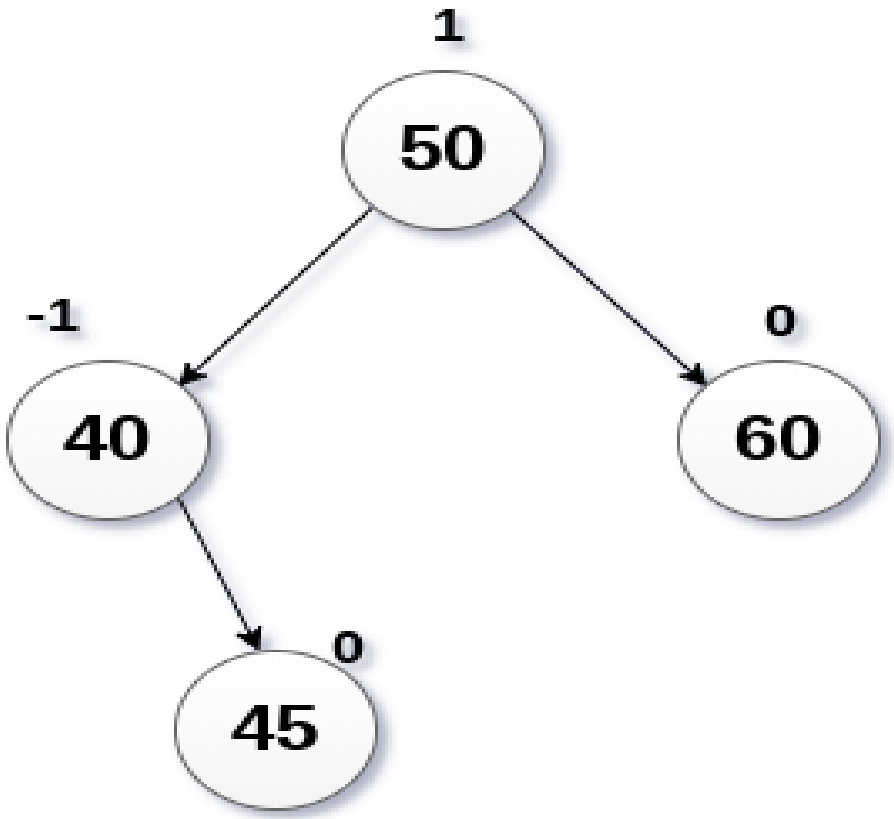

1
Node A
50
1
Node B
40
1
60
1
30
0
45
0
55
10
Node to be deleted
(X)
AVL Tree
Deleting Node 55
2
Critical Node
Node A
50
1
40
Node B
0
60
1
30
0
45
10
Non AVL Tree
Performing R1
rotation
0
Node B
40
1
30
Node A
50
0
0
10
0
45
0
60
R1 Rotated Tree

R-1 Rotation (Node B has balance factor -1)

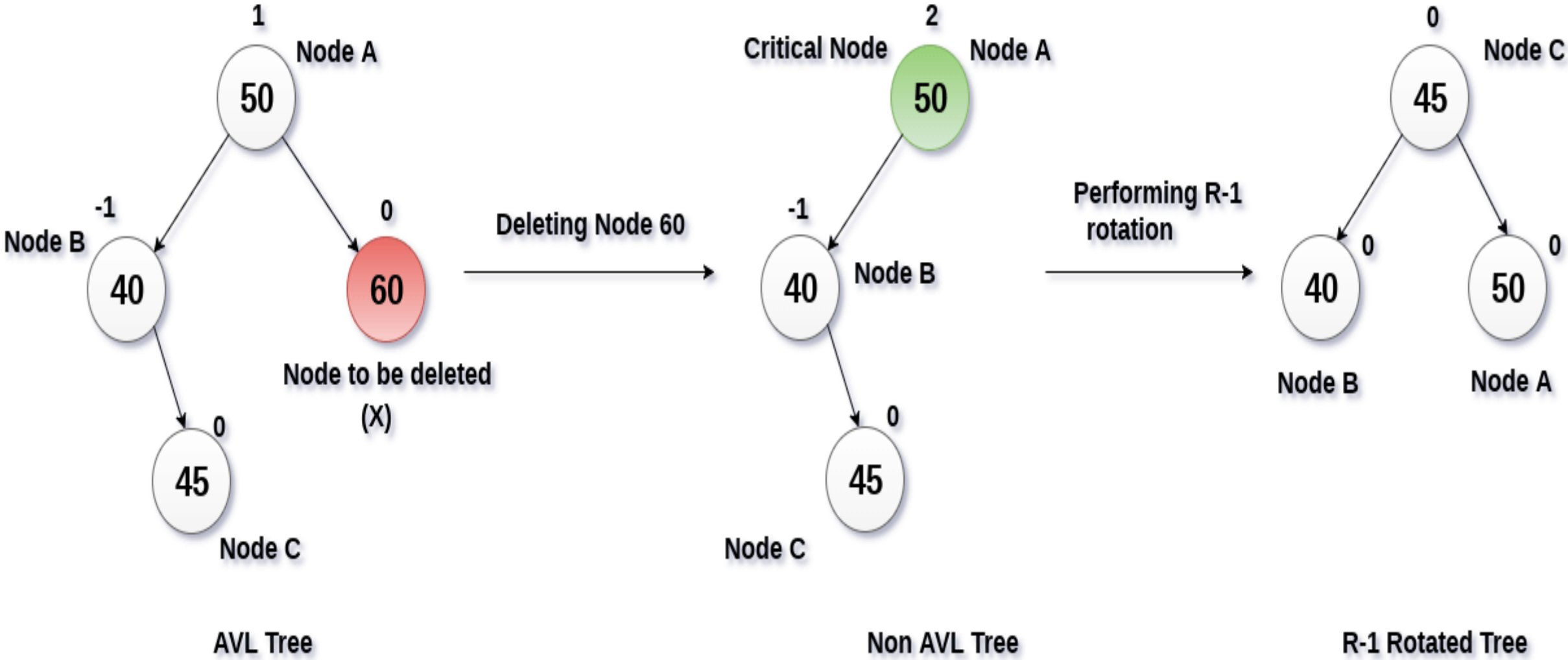
- R-1 rotation is to be performed if the node B has balance factor -1. This case is treated in the same way as LR rotation. In this case, the node C, which is the right child of node B, becomes the root node of the tree with B and A as its left and right children respectively.
- The sub-trees T1, T2 becomes the left and right sub-trees of B whereas, T3, T4 become the left and right sub-trees of A.
- The process involved in R-1 rotation is shown in the following image.


1
A
-1
B
T4
(h)
T1
(h-1)
C
T2
(h-1)
T3
(h-1)
Deleting Node from
Right Sub-tree of A
2
A
-1
B
T4
(h-1)
X
0
T1
(h-1)
C
T2
(h-1)
T3
(h-1)
Performing R-1
Rotation
0
C
0
B
0
A
T1
(h-1)
T2
(h-1)
T3
(h-1)
T4
(h-1)
AVL Tree
Non AVL Tree
R-1 Rotated Tree

Example
Delete the node 60 from the AVL tree shown in the following image.



- Solution:
- in this case, node B has balance factor -1. Deleting the node 60, disturbs the balance factor of the node 50 therefore, it needs to be R-1 rotated. The node C i.e. 45 becomes the root of the tree with the node B(40) and A(50) as its left and right child.


1
Node A
50
-1
Node B
40
0
60
Node to be deleted
(X)
0
45
Node C
AVL Tree
Deleting Node 60
2
Critical Node
Node A
50
-1
40
Node B
0
45
Node C
Non AVL Tree
Performing R-1
rotation
0
Node C
45
0
40
0
50
Node B
Node A
R-1 Rotated Tree

- THANK YOU