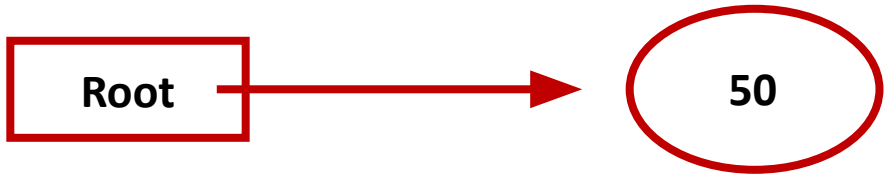
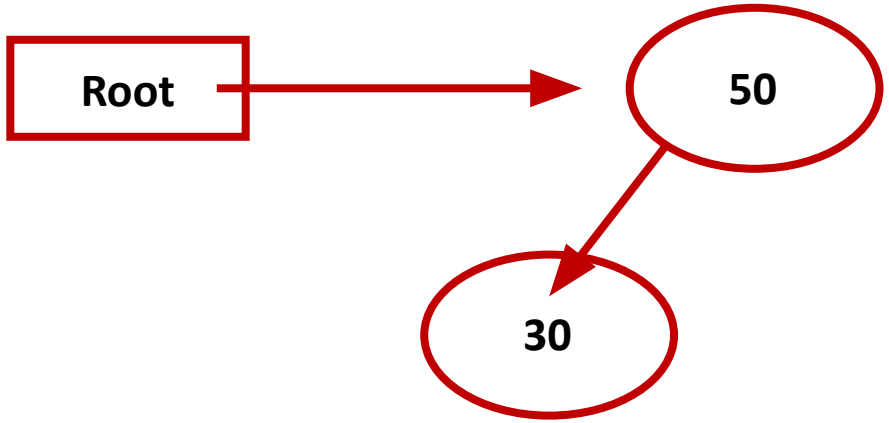
# Binary Search Tree

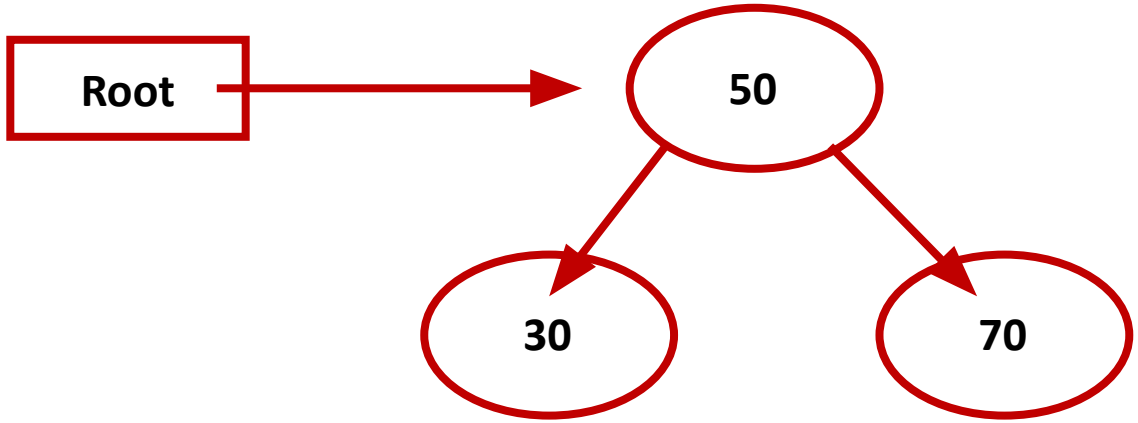
# Insert 50 in BST

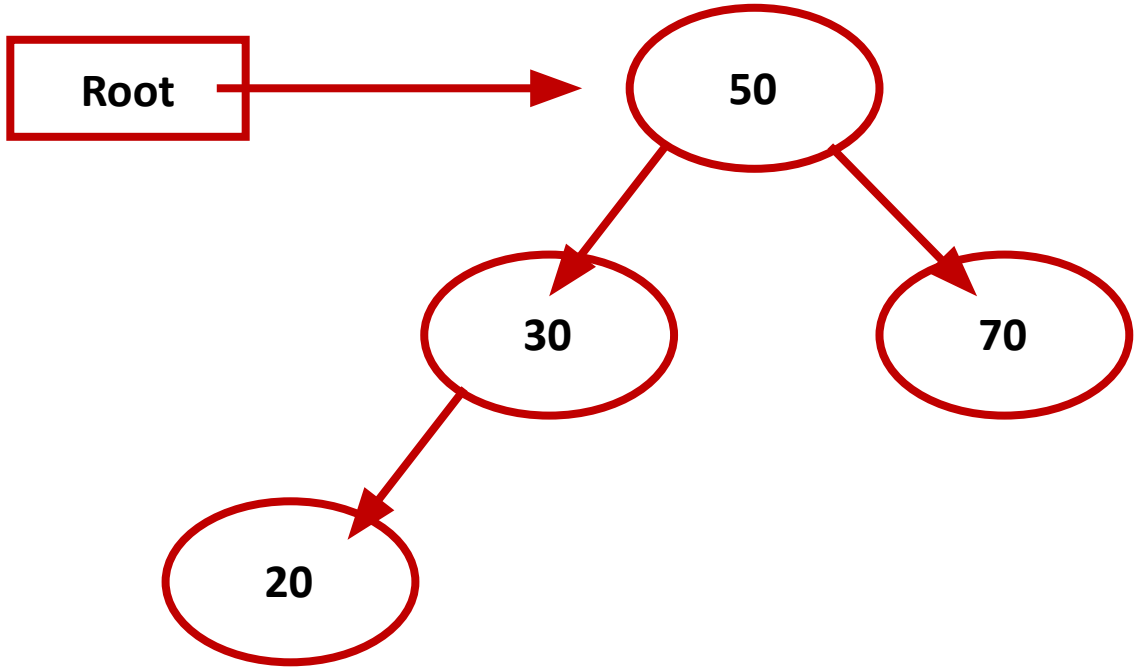
# Insert 30 in BST



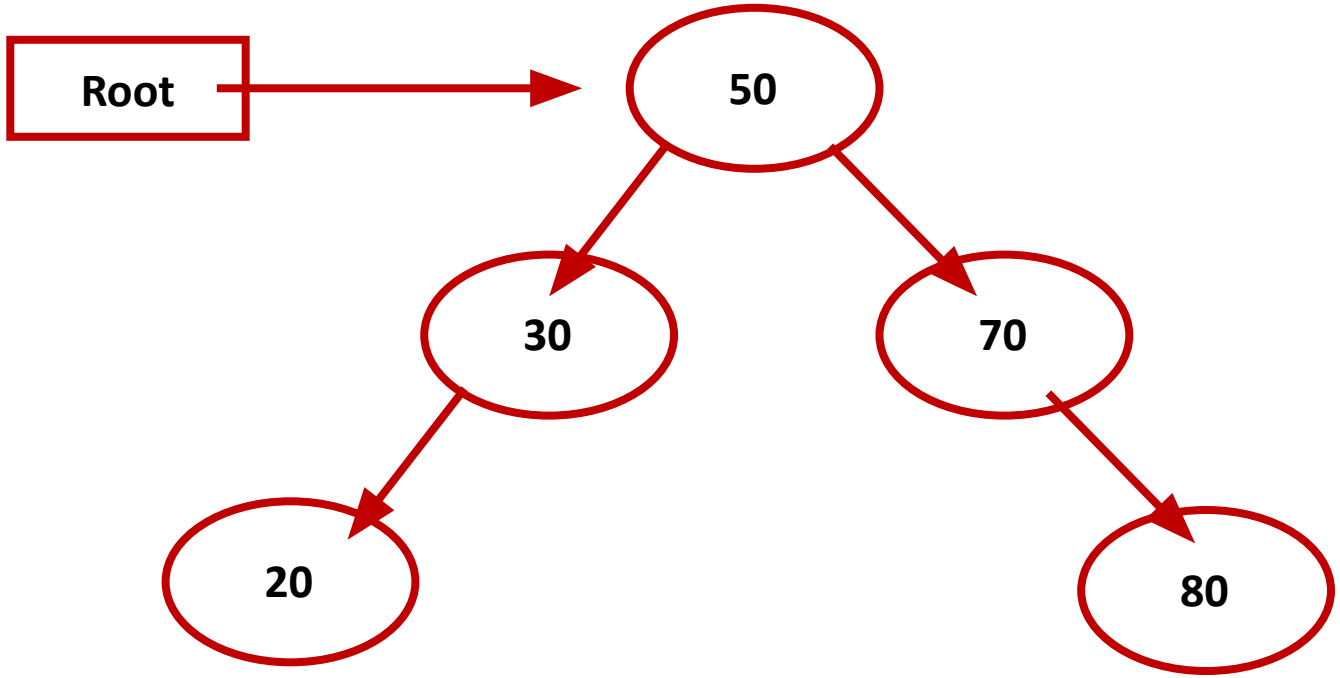
# Insert 70 in BST



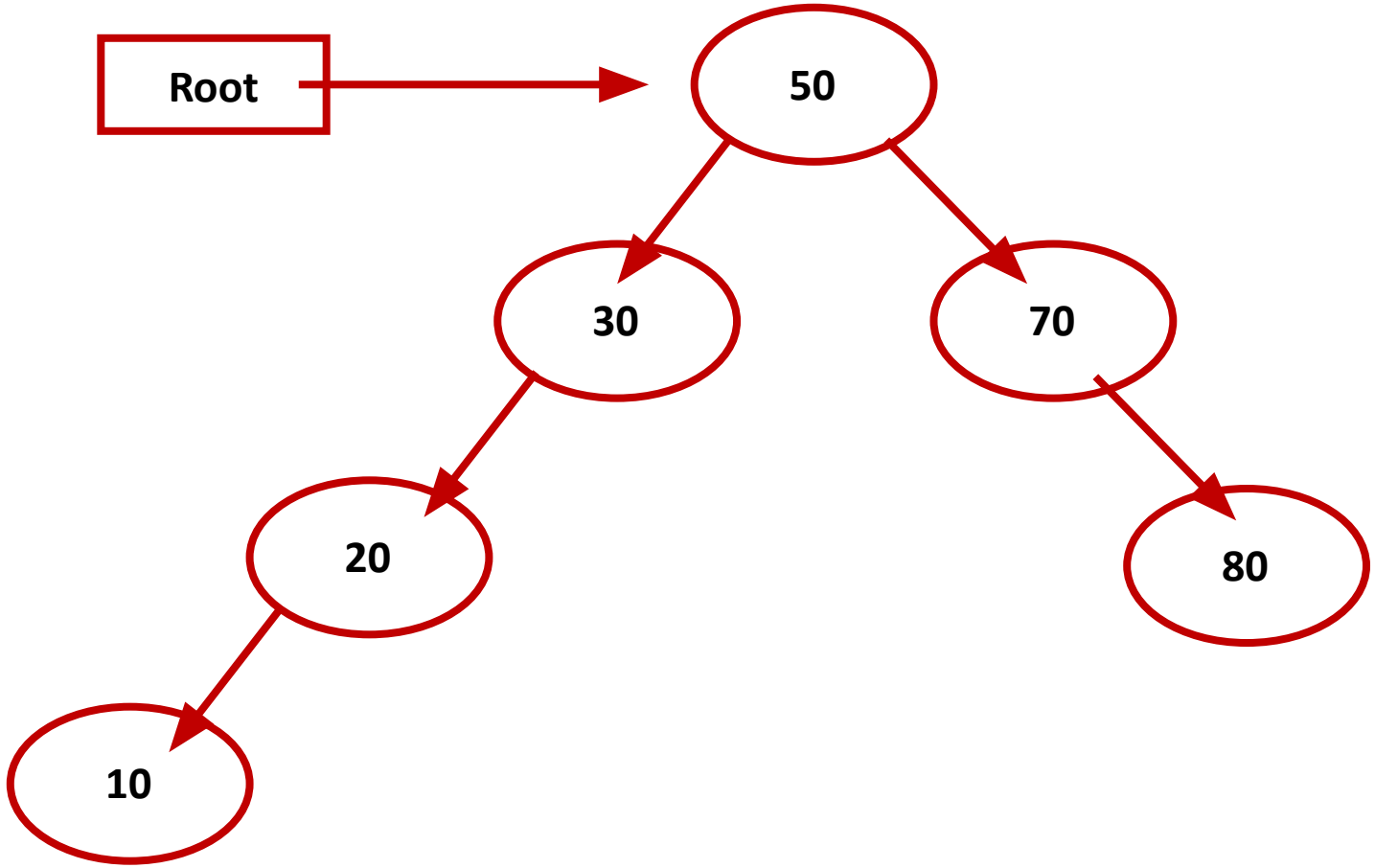
# Insert 20 in BST



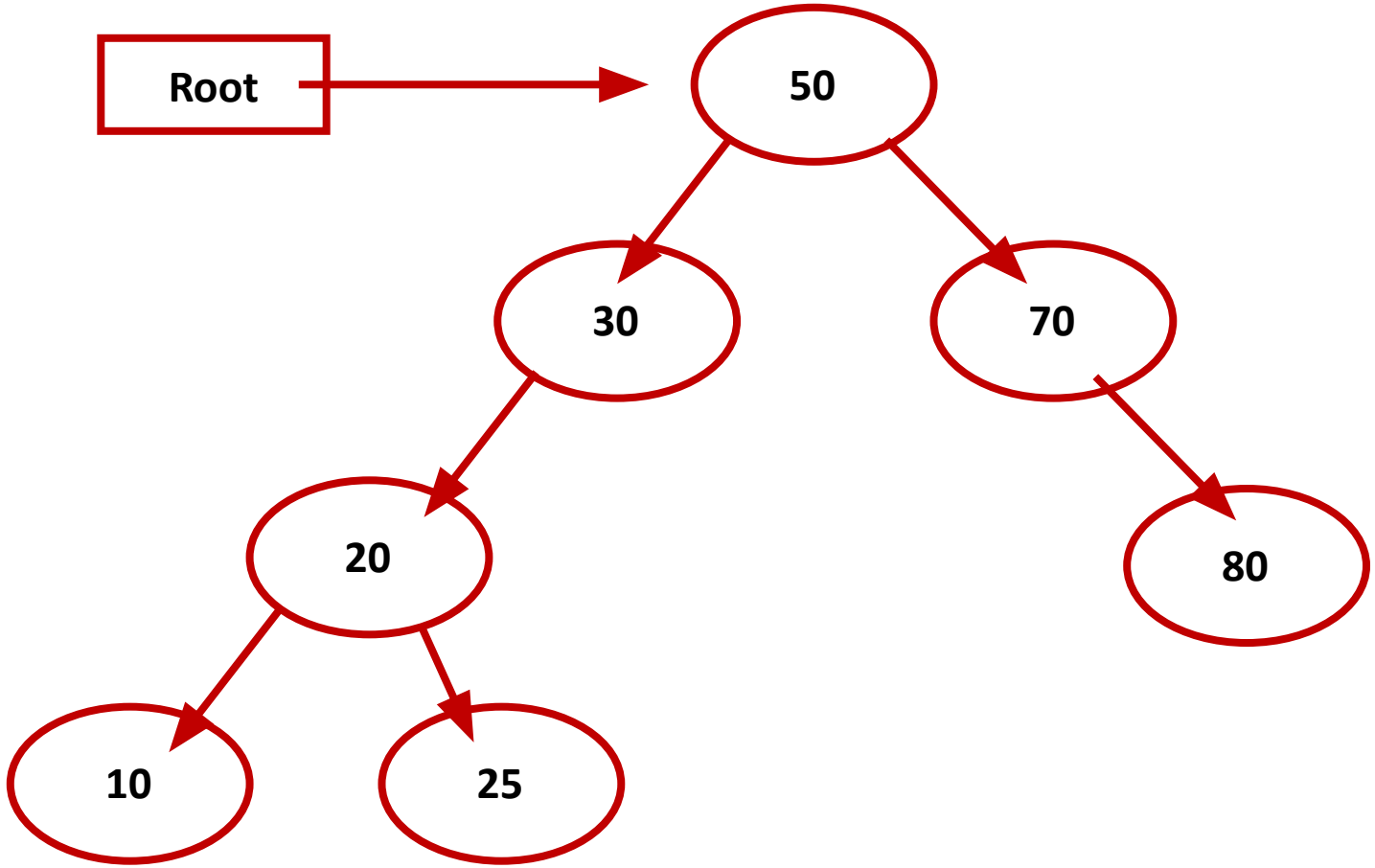
# Insert 80 in BST



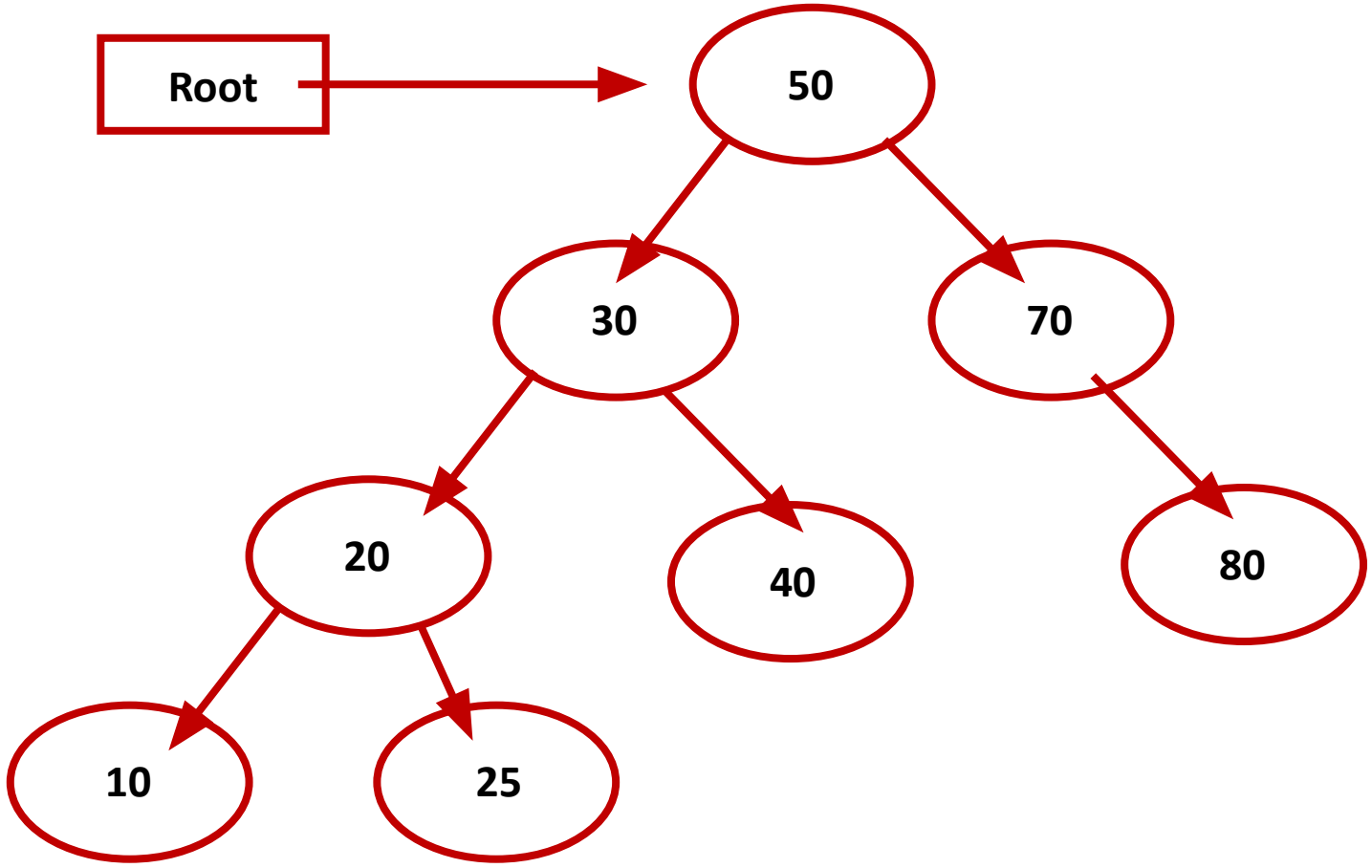
# Insert 10 in BST



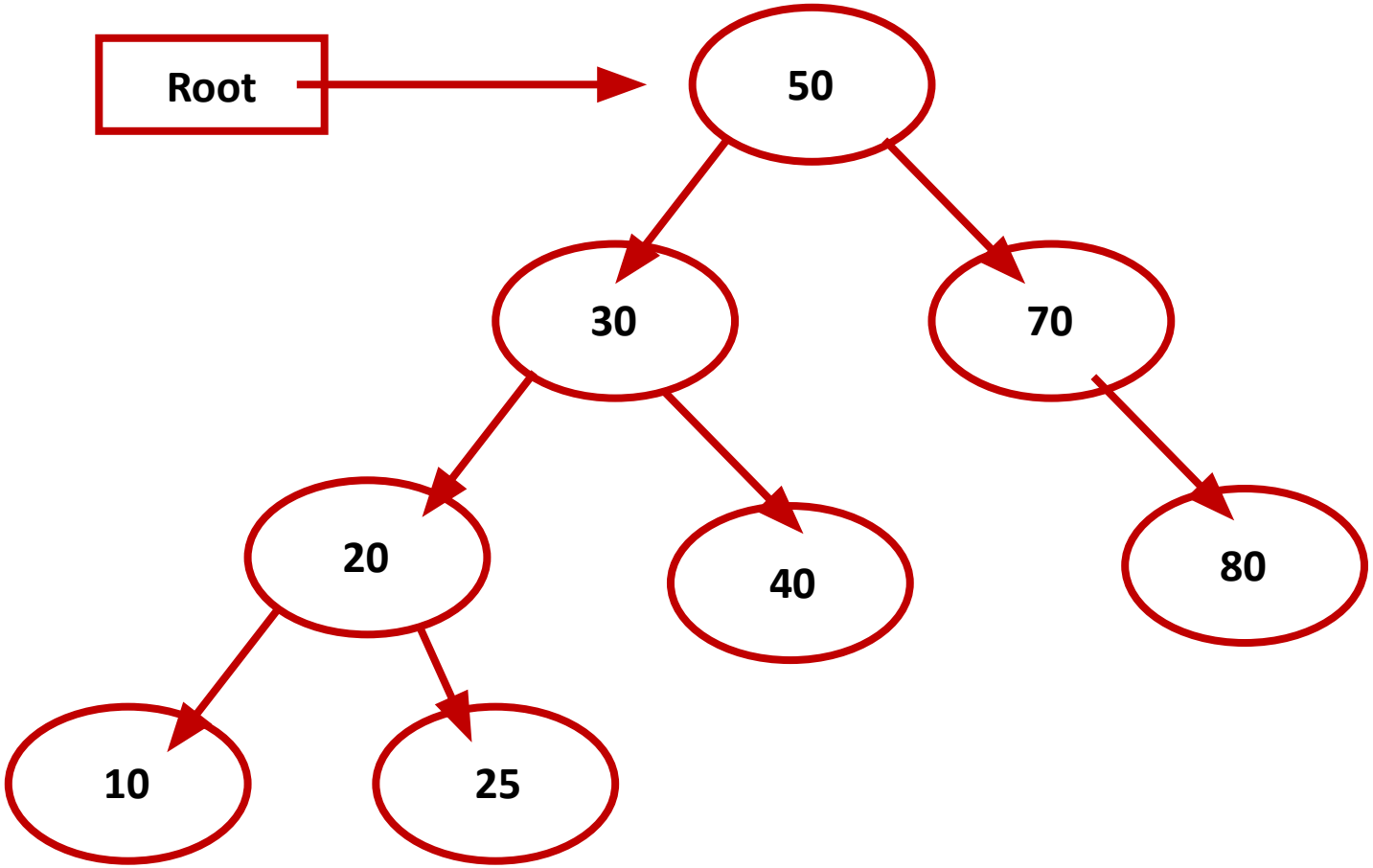
# Insert 25 in BST



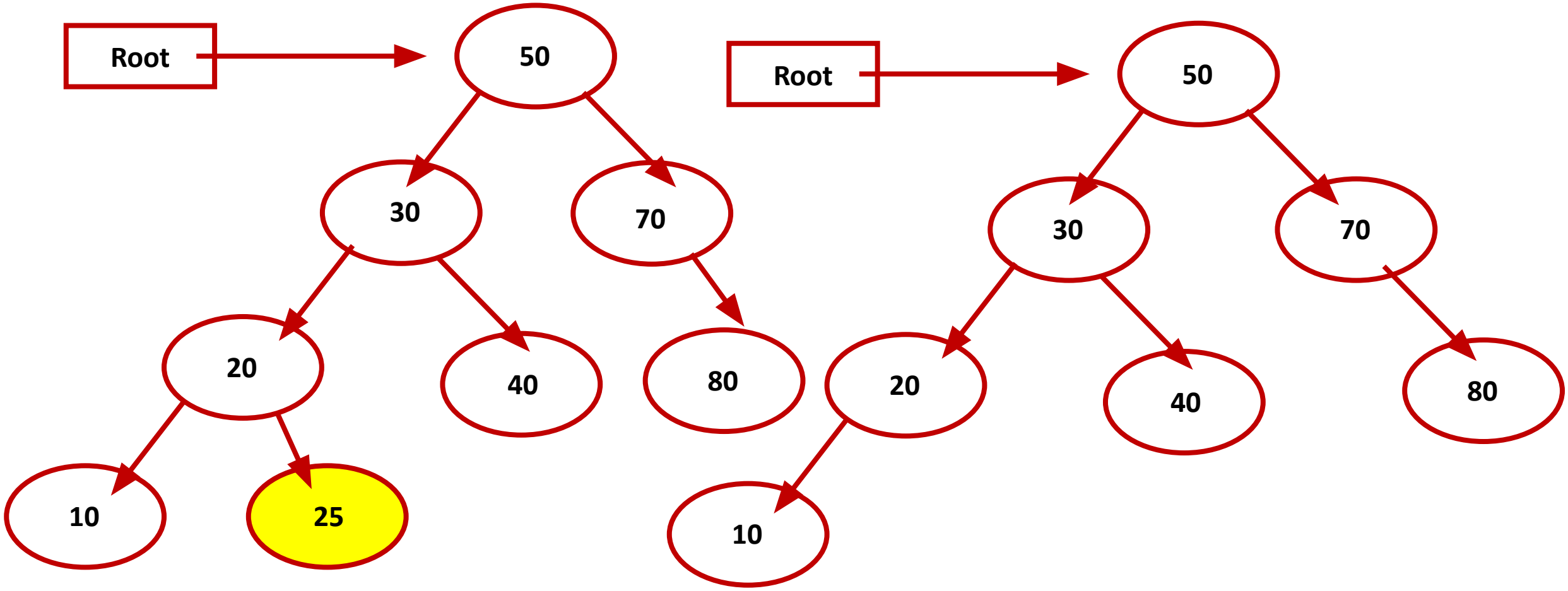
# Insert 40 in BST



# Deletion from BST – Case 1: Delete Leaf Node (25)



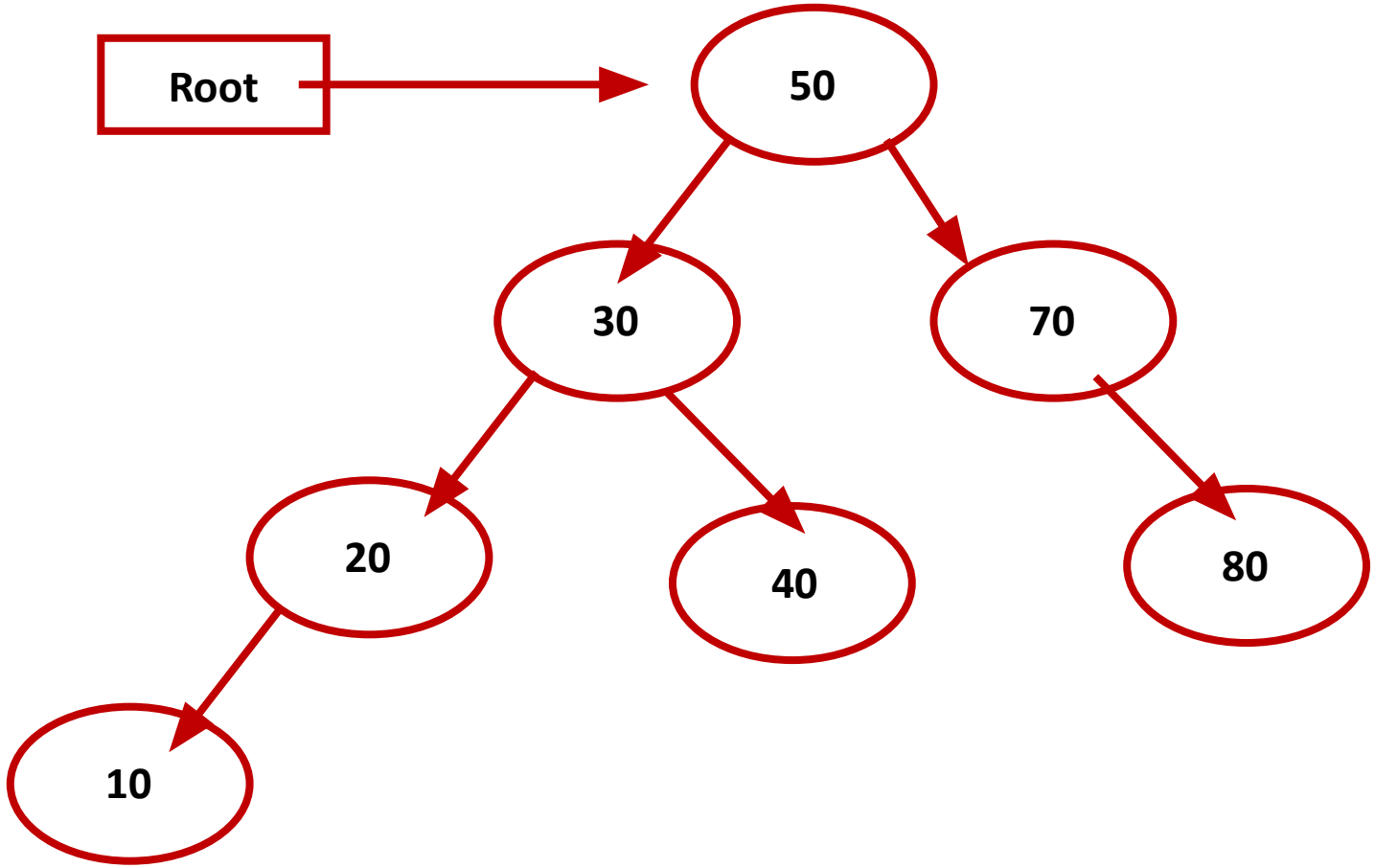
# Deletion from BST – Case 1: Delete Leaf Node (25)



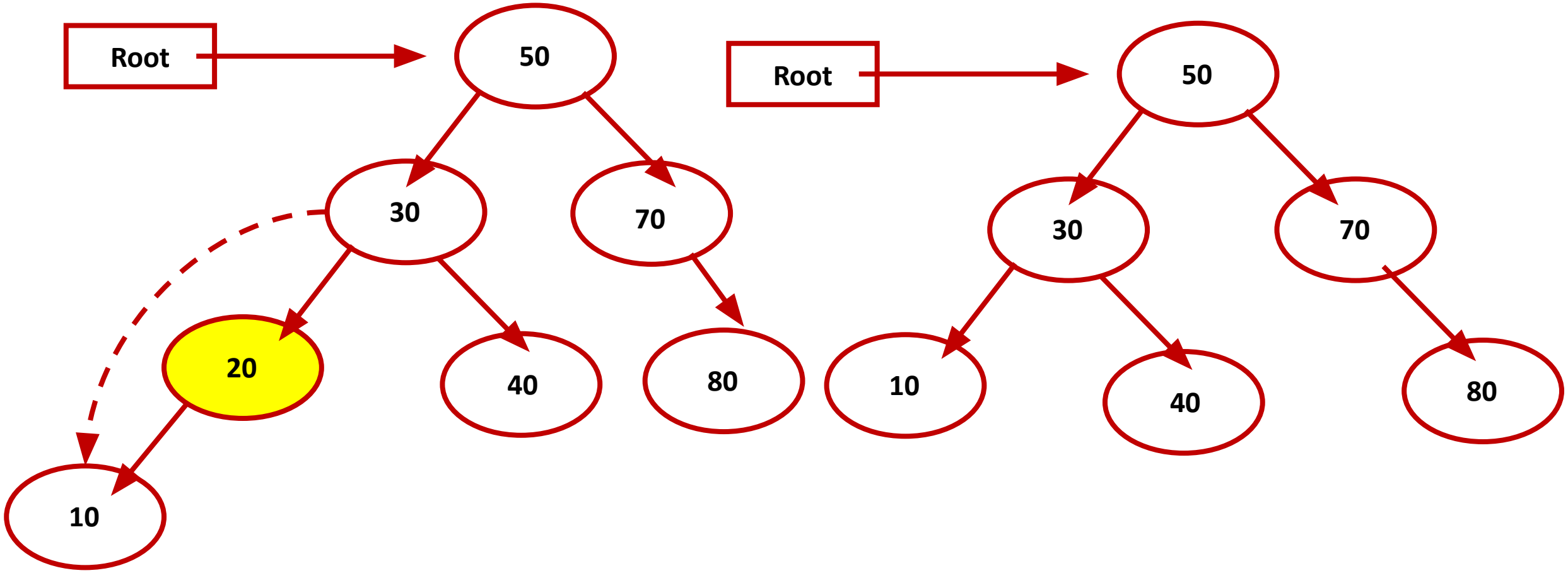

Before Deletion

After Deletion

# Case 2: Delete Node having Right or Left Child Only (20)



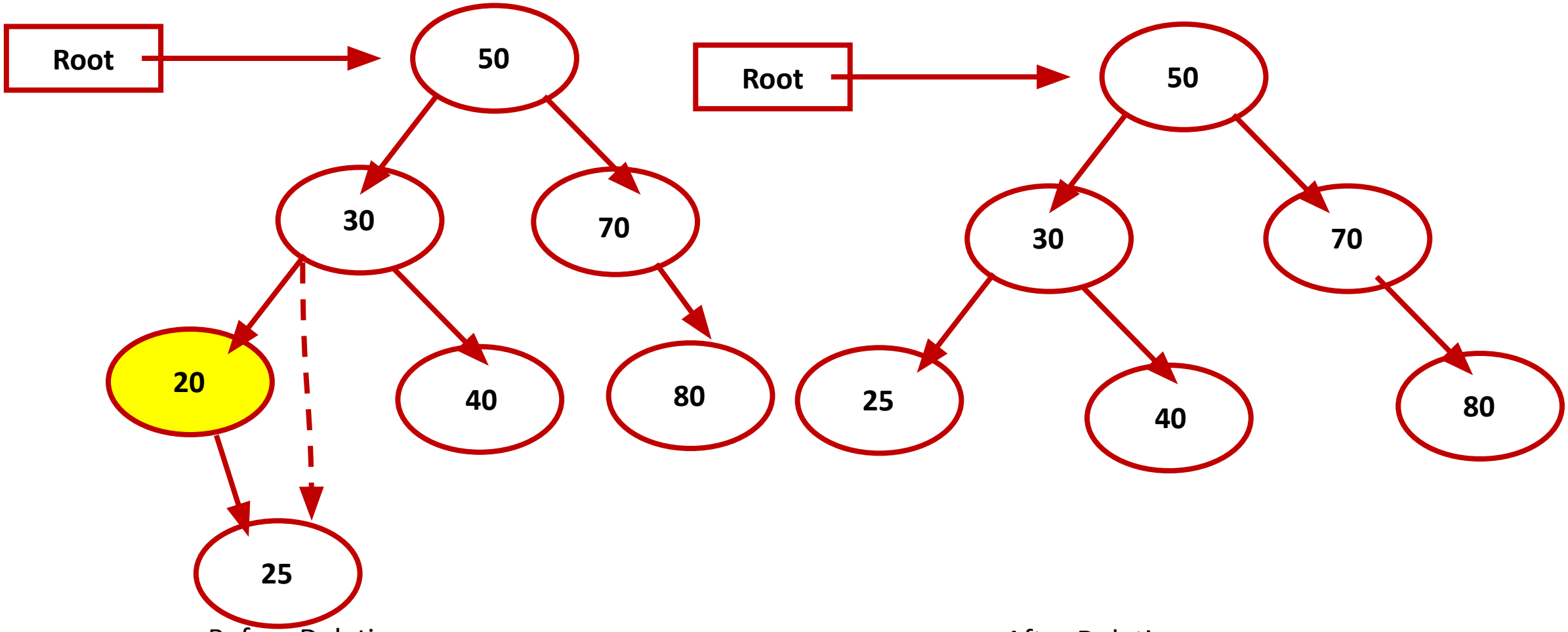
# Case 2: Delete Node having Right or Left Child Only (20)




Before Deletion

After Deletion

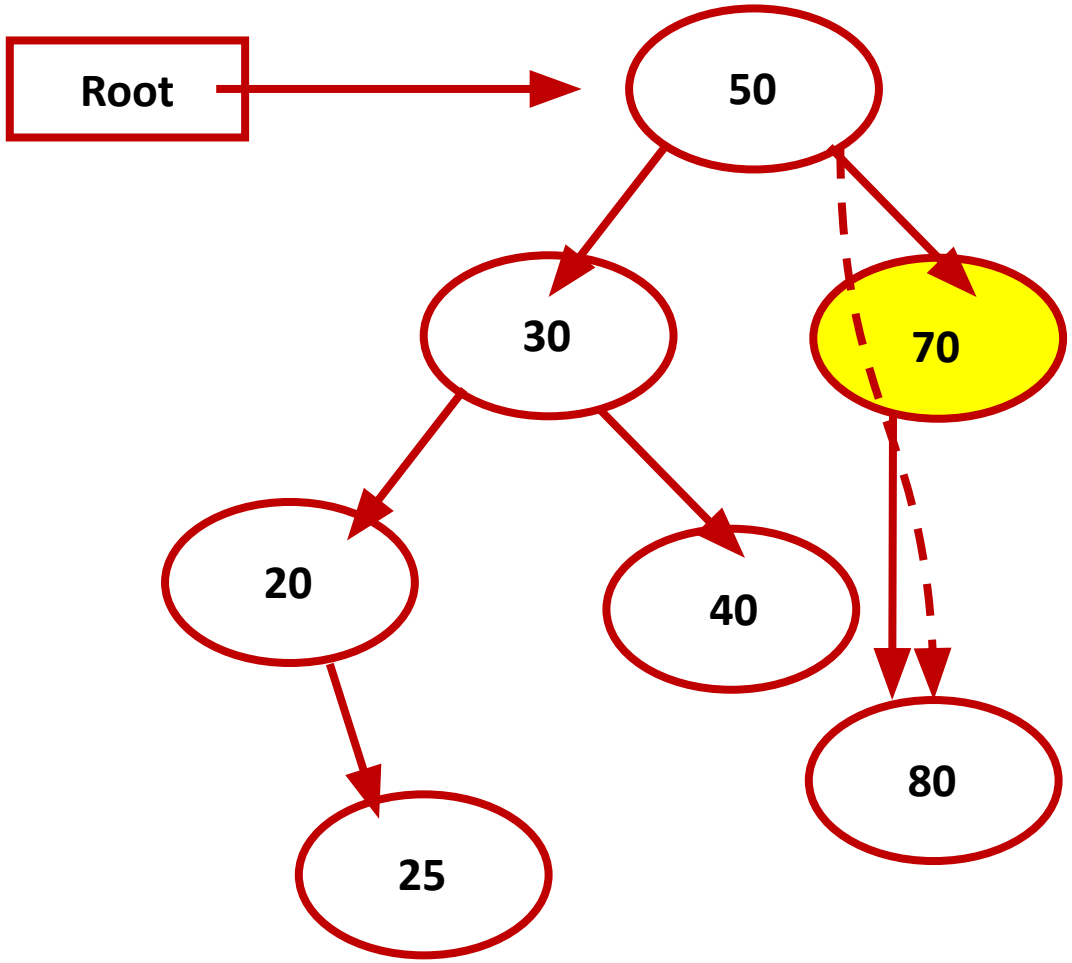
# Case 2: Delete Node having Right or Left Child Only (20)



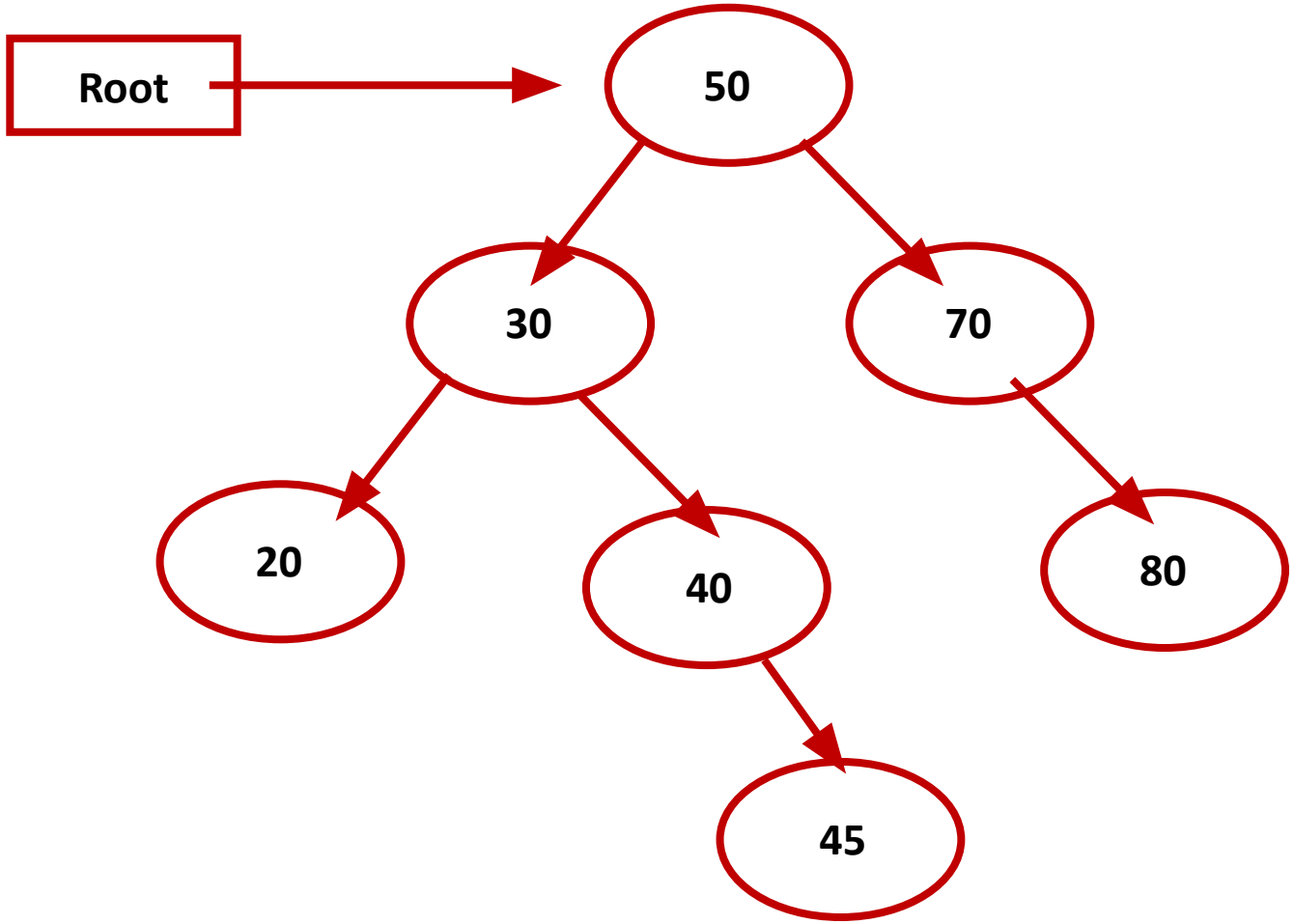

Before Deletion

After Deletion

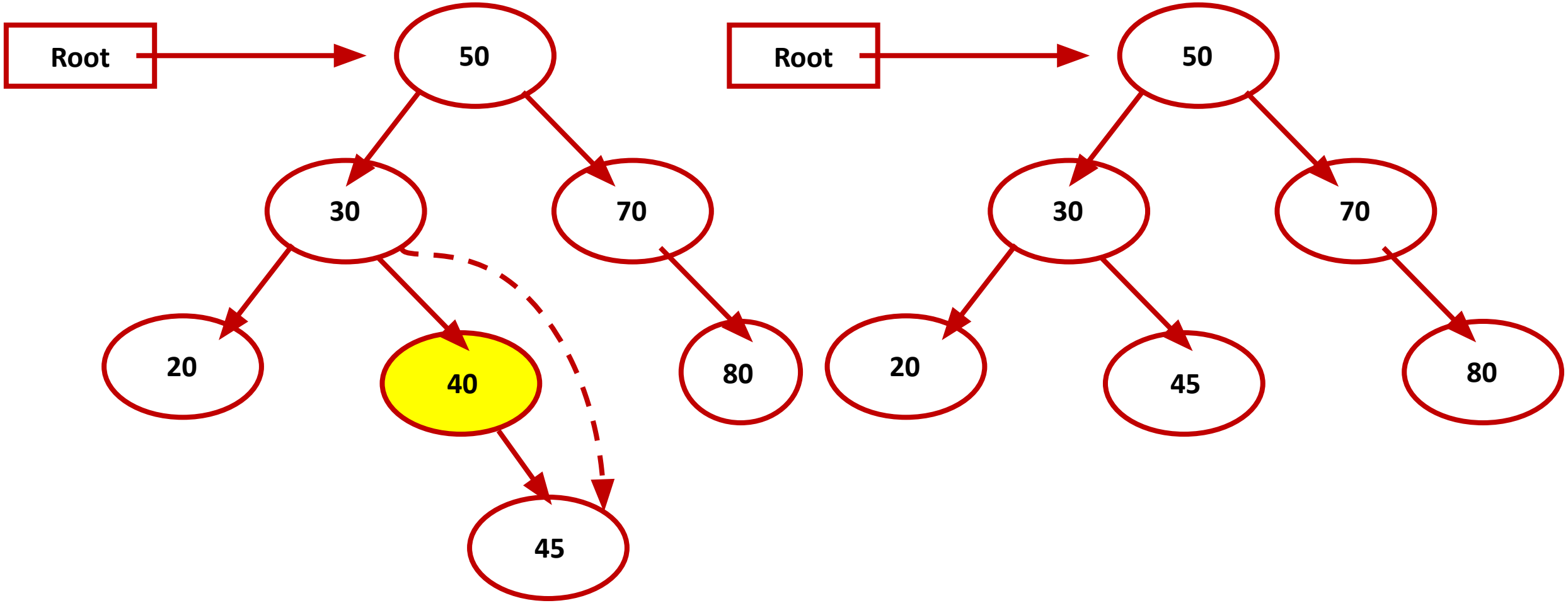
# Case 2: Delete Node having Right or Left Child Only (70)



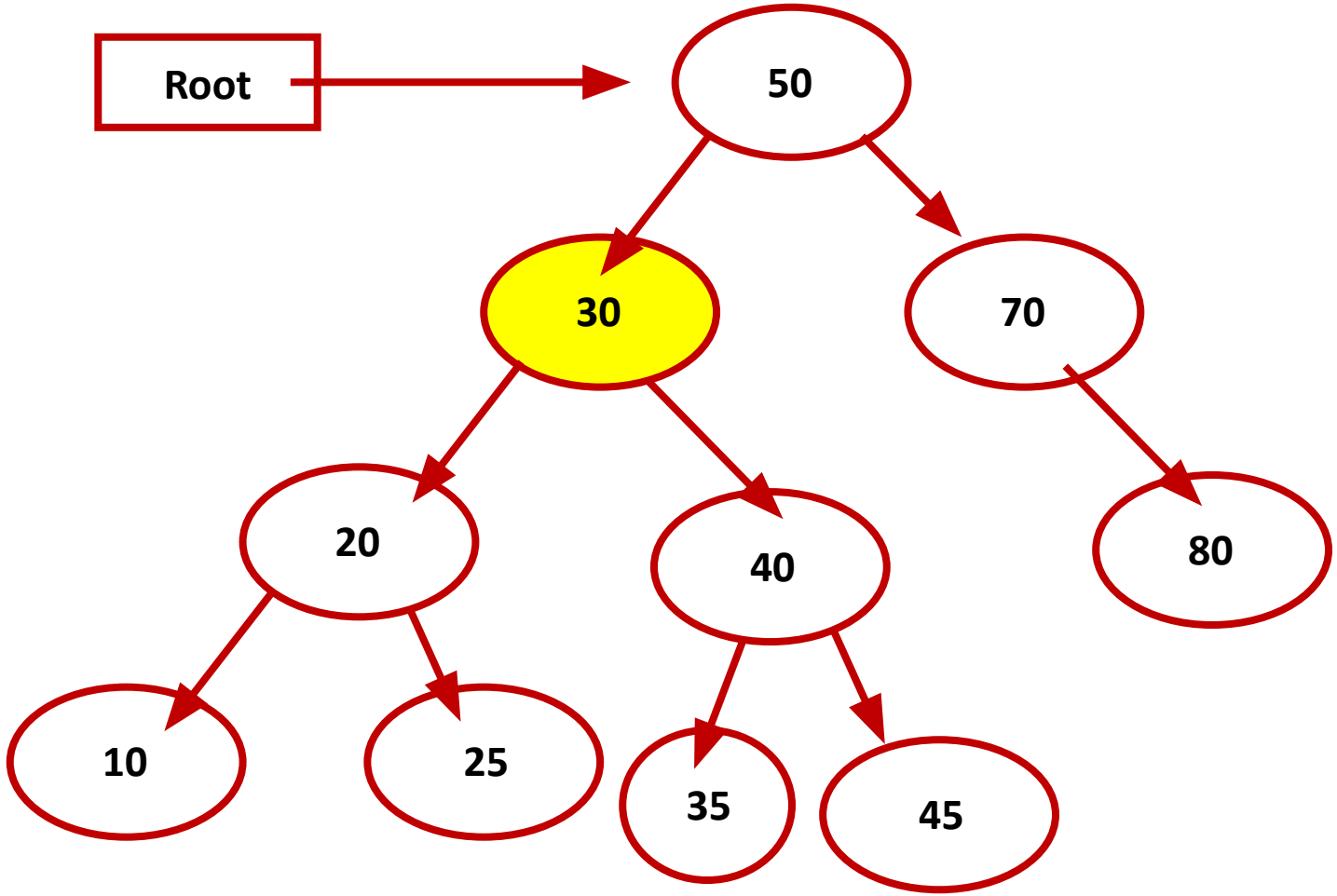
# Case 2: Delete Node having Right or Left Child Only (40)



# Case 2: Delete Node having Right or Left Child Only (40)

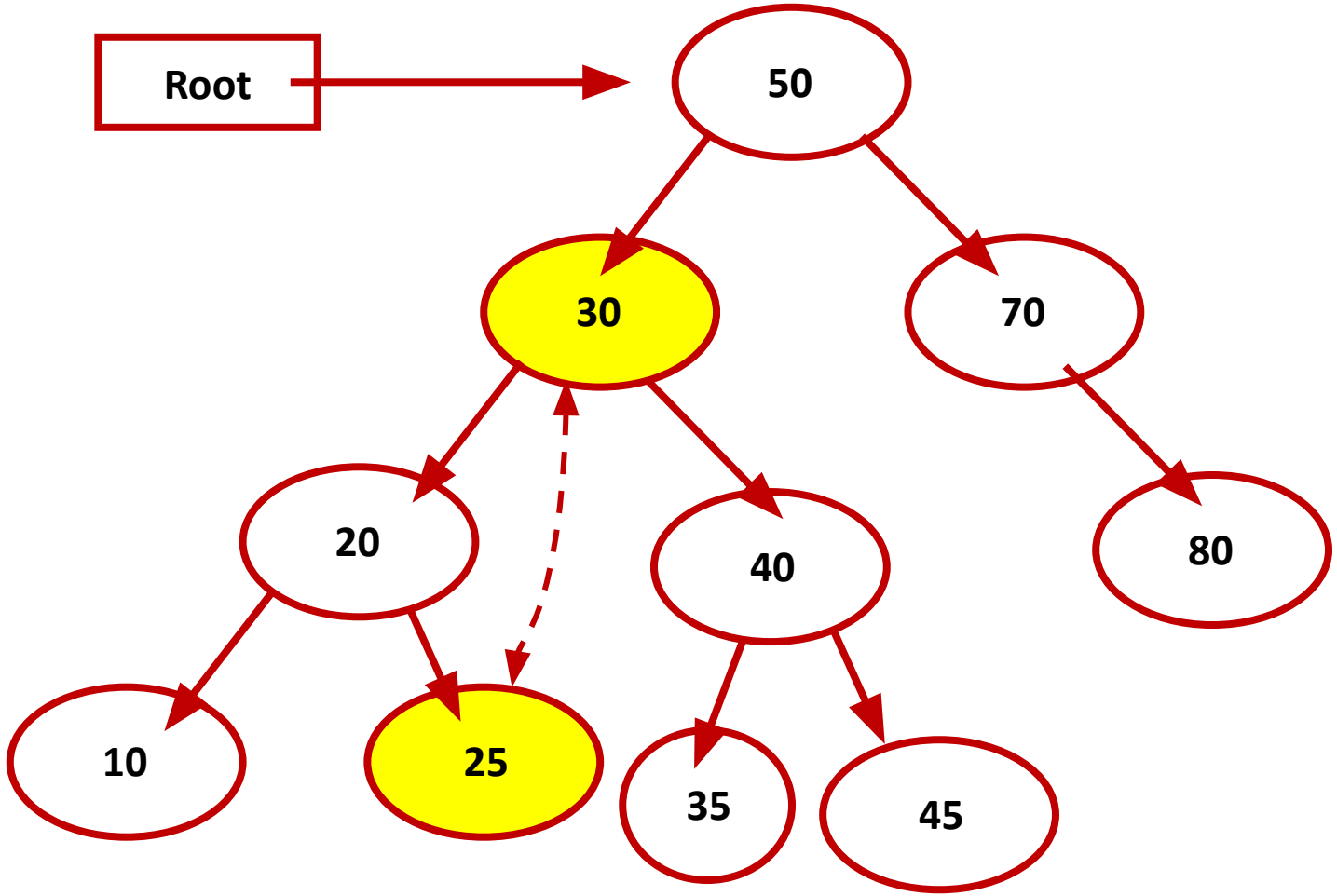


# Case 3: Delete Node having both Right and Left Subtrees (30)



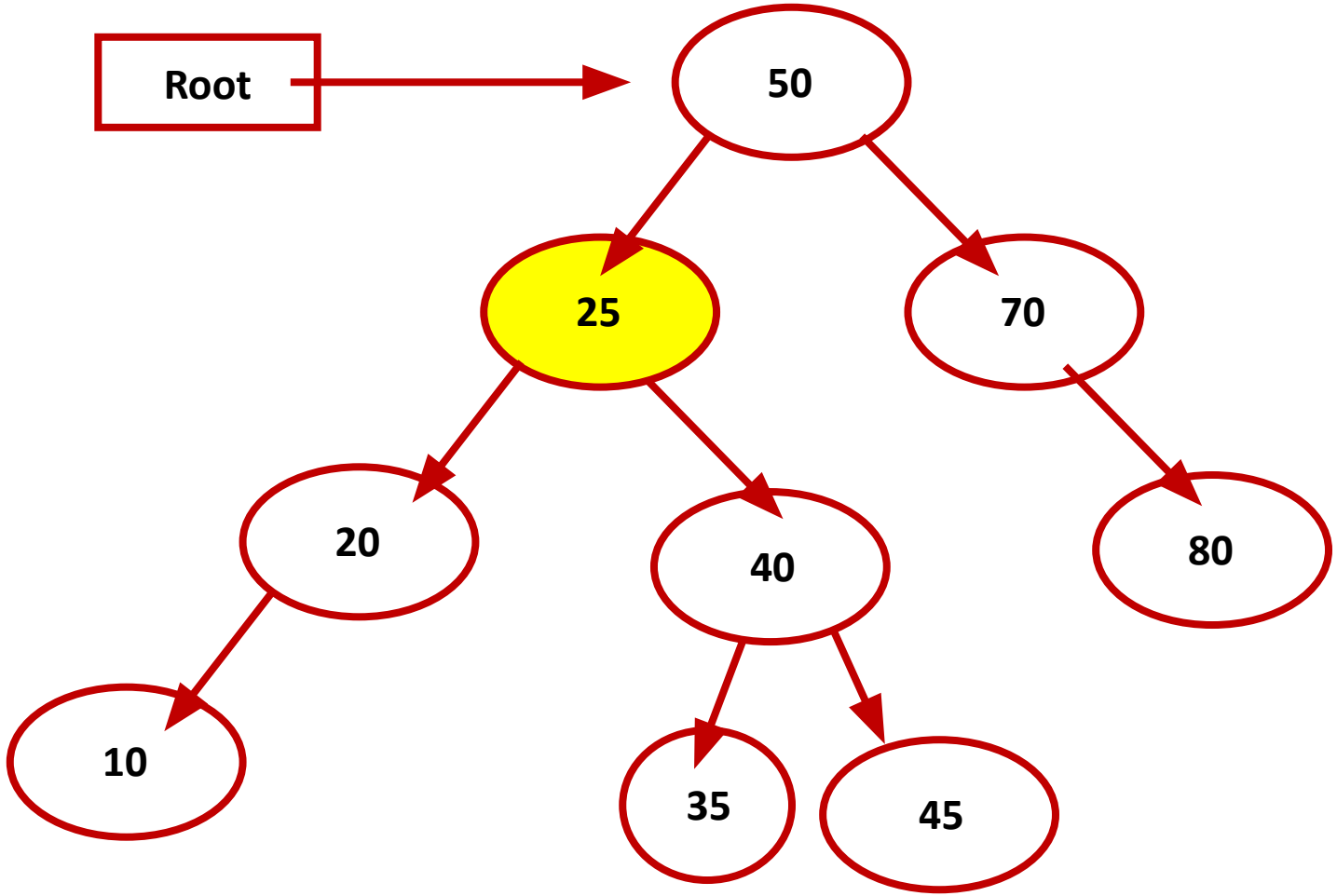
# Case 3: Delete Node having both Right and Left Subtrees (30)

Case (i) – Replace Node with Max
Of Left Subtree



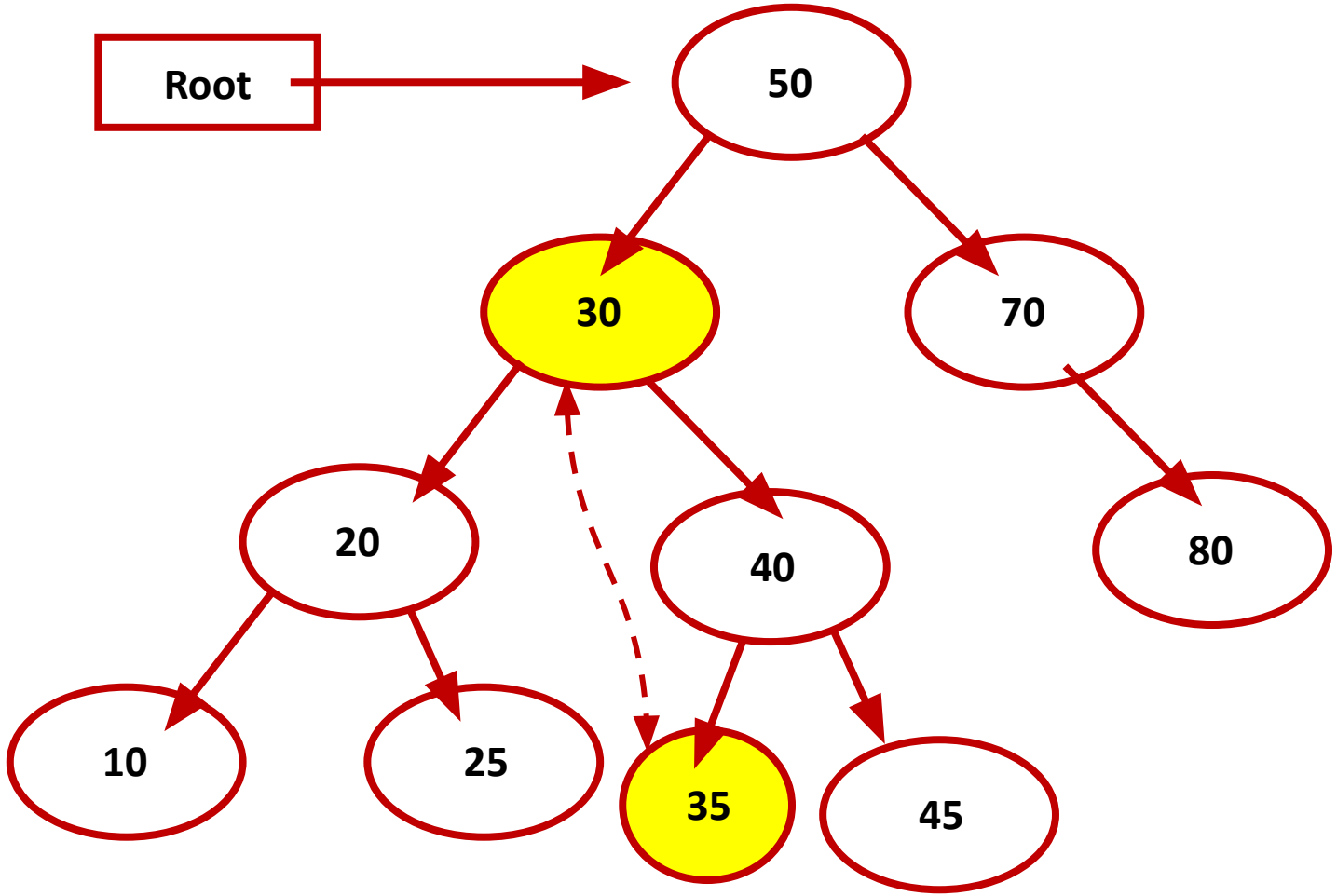
# Case 3: Delete Node having both Right and Left Subtrees (30)

Case (i) – Replace Node with Max
Of Left Subtree

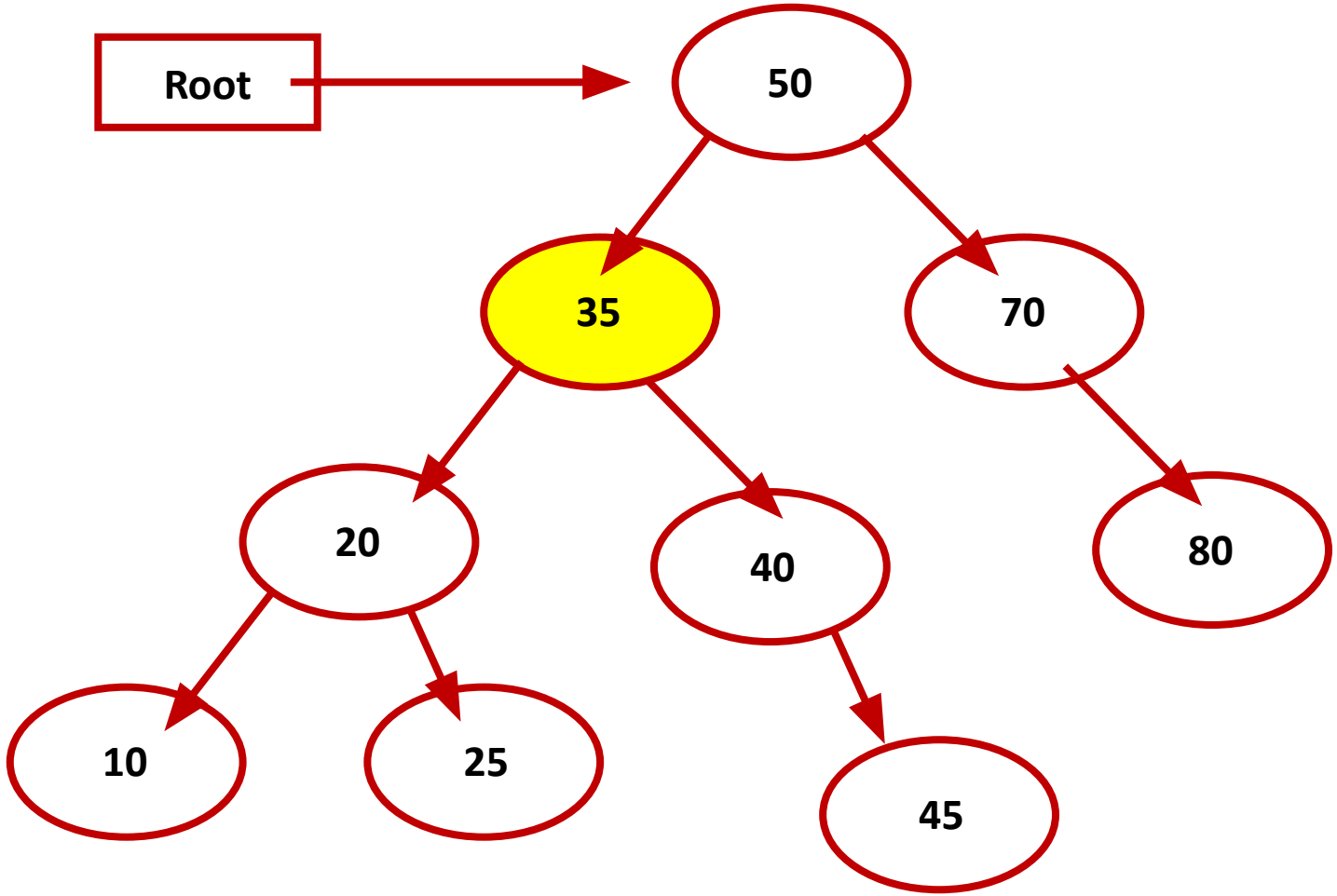


# Case 3: Delete Node having both Right and Left Subtrees (30)

Case (ii) – Replace Node with Min
Of Right Subtree



# Case 3: Delete Node having both Right and Left Subtrees (30)



# Homework

- Total number of Nodes in a Tree
- Degree of a node
- Total number of leaf nodes
- Total number of intermediate/non-leaf nodes
- Print sibling of a node
- Print Ancestors of a node
- Print Level of a node
- Total number of nodes <= level ‘l’
- Find Height of Tree

# Homework…

- Is the tree full?
- Find min level of leaf
- Recursive and Iterative Versions of
  - InOrder
  - PreOrder
  - Post Order
- Level Order Traversal
- Bool IsBST()
- Bool IsEqual(Node* tree1, Node* tree2)
- ~BST()