

# Data Structure Algorithms & Applications

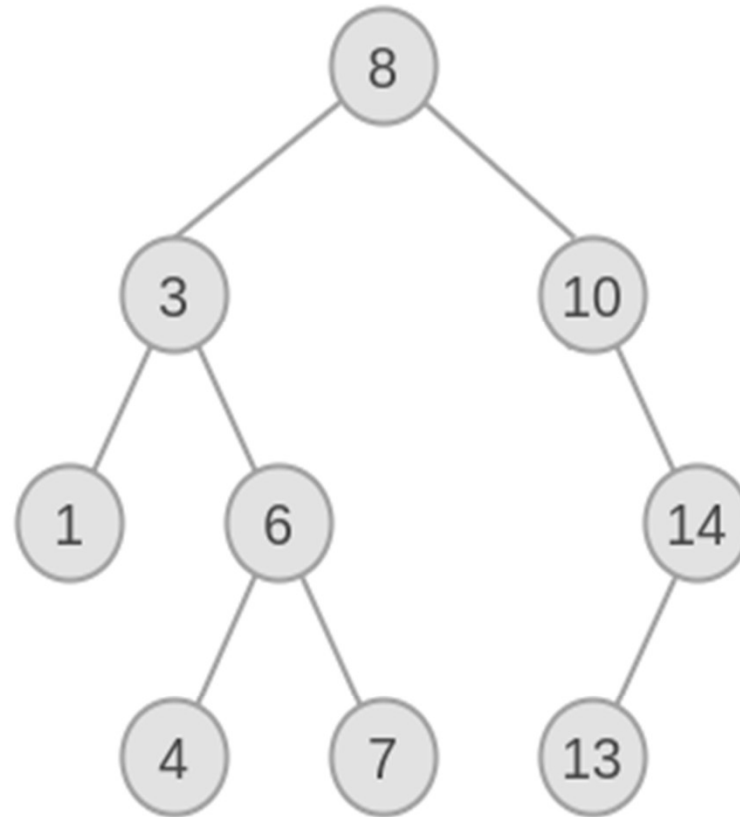
**CT-159**

Prepared by  
Muhammad Kamran

# Binary Search Tree

- A **Binary Search Tree** is a data structure used in computer science for organizing and storing data in a sorted manner.
- Each node in a **Binary Search Tree** has at most two children, a **left** child and a **right** child, with the **left** child containing values less than the parent node and the **right** child containing values greater than the parent node.
- This hierarchical structure allows for efficient **searching**, **insertion**, and **deletion** operations on the data stored in the tree.

# Binary Search Tree



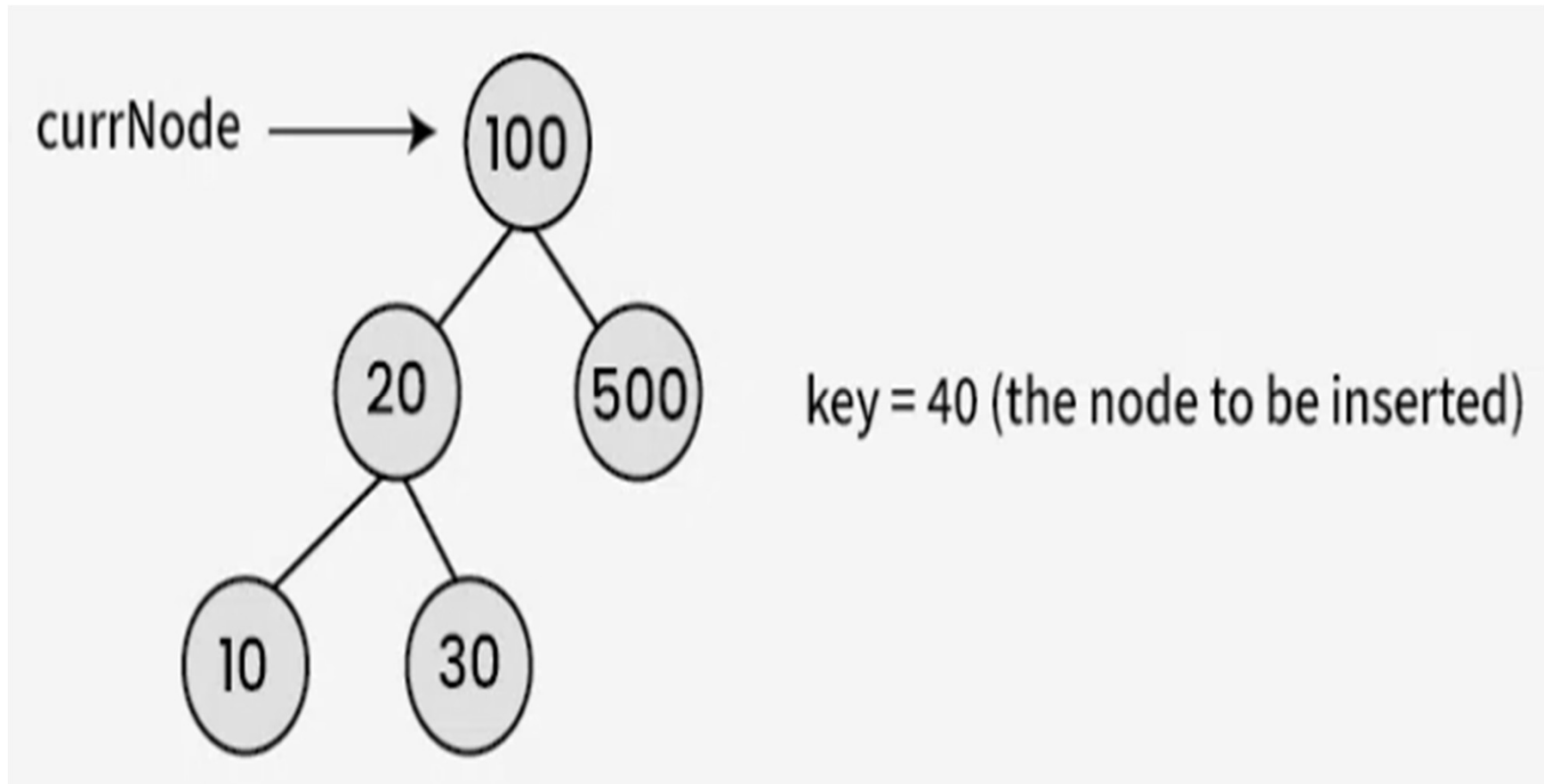
Binary Search Tree

# Insertion Operation

The below steps are followed while we try to insert a node into a binary search tree:

- Initialize the current node (say, **currNode** or **node**) with root node
- Compare the **key** with the current node.
- **Move left** if the **key** is less than or equal to the current node value.
- **Move right** if the **key** is greater than current node value.
- Repeat steps 2 and 3 until you reach a leaf node.
- Attach the **new key** as a left or right child based on the comparison with the leaf node's value.

# Insertion Operation

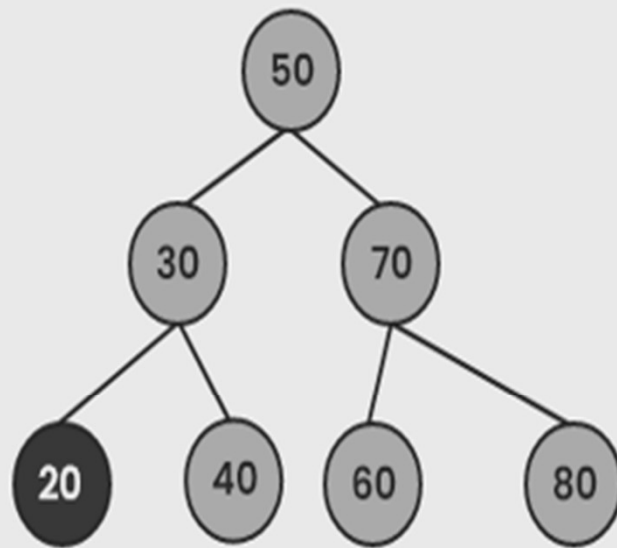


# Deletion of a Node in BST

Given a **BST**, the task is to delete a node in this **BST**, which can be broken down into 3 scenarios:

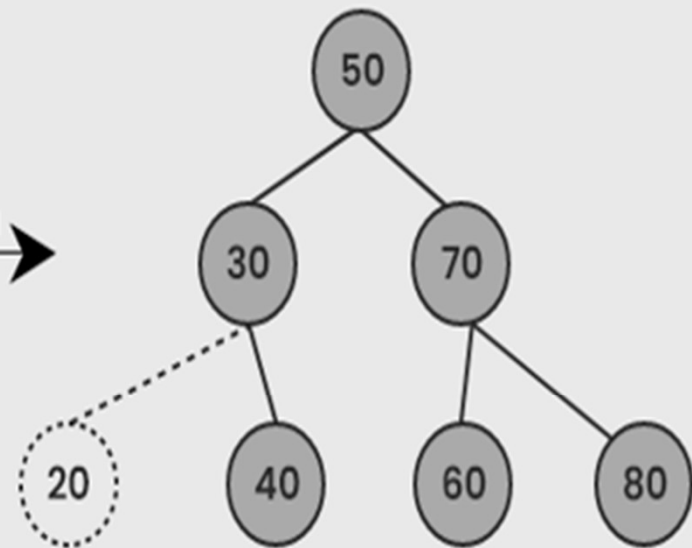
# Case 1. Delete a Leaf Node in BST

## Case 1 : Delete A Leaf Node In BST



Delete Node 20

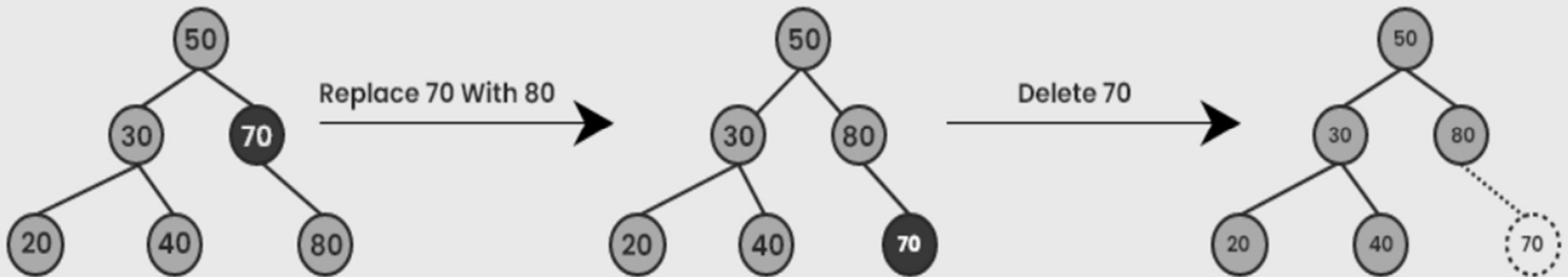
Assign Node To Null



Deleted Node 20

## Case 2. Delete a Node with Single Child in BST

### Case 2: Delete A Node With Single Child In BST



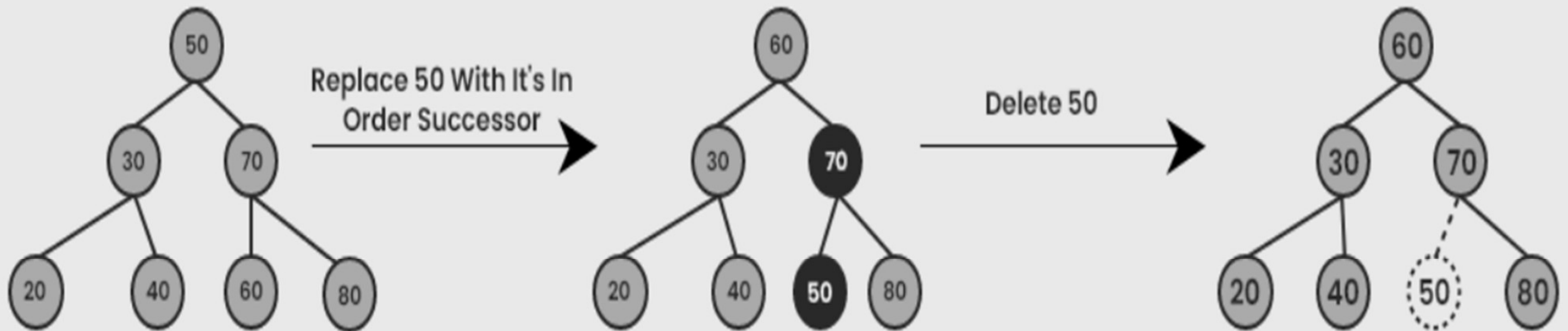
Delete Node 70

After Deletion



# Case 3. Delete a Node with Both Children in BST

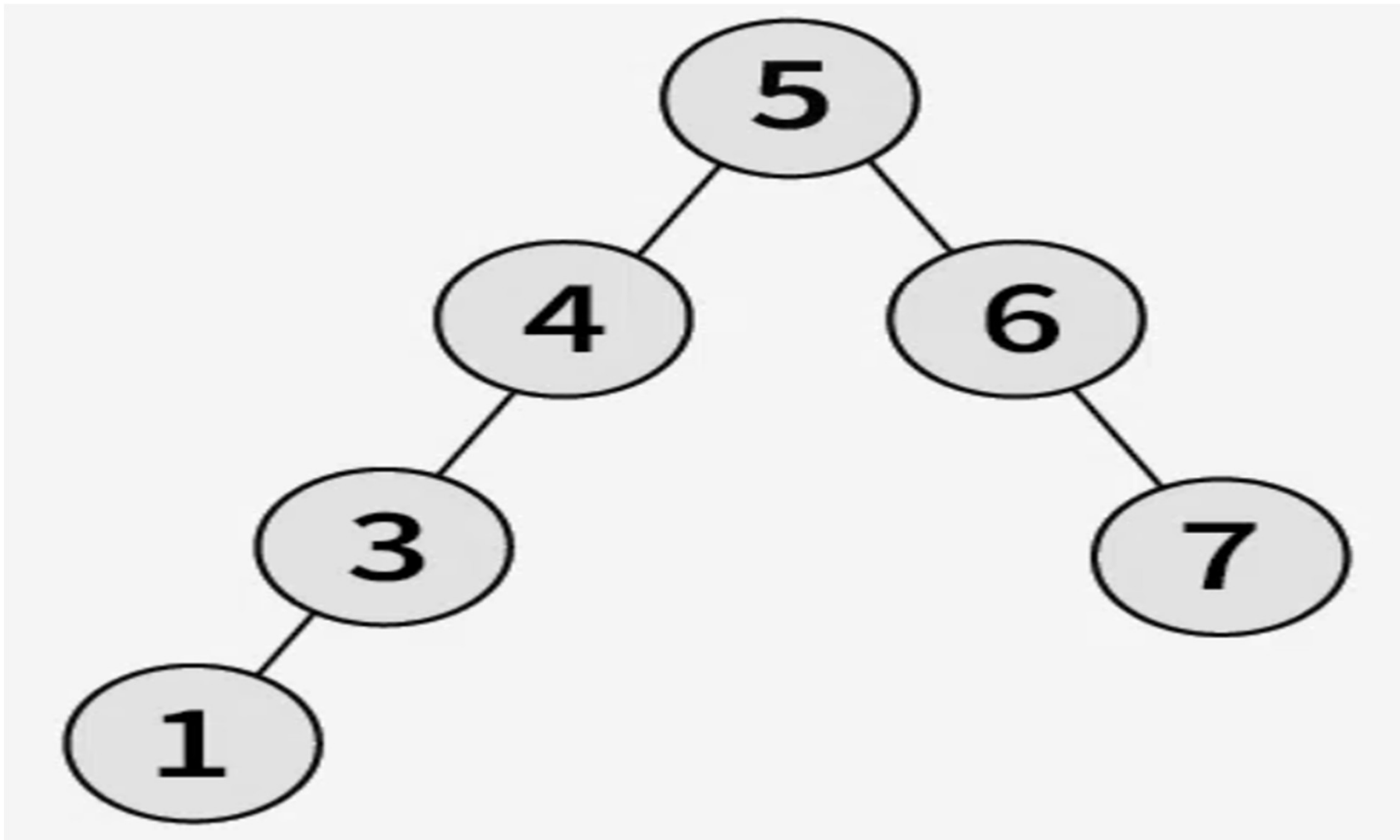
## Case 3 : Delete A Node With Both Children In BST



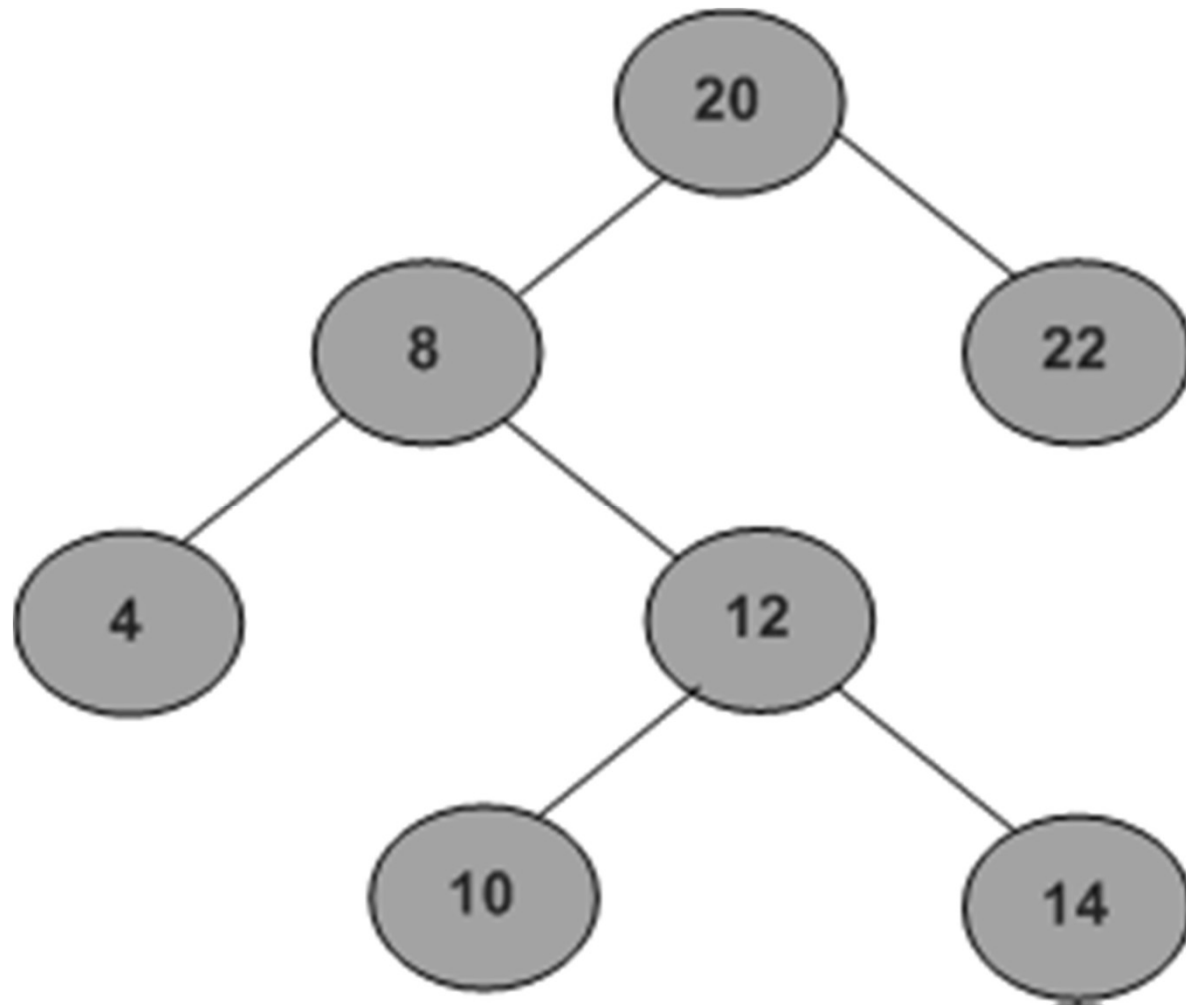
Delete Node 50

After Deletion

# Minimum Value in a BST



# Maximum Value in a BST



Thank You