# Name: Najim Khan
# Roll No: 2401420003
# Course: B.Tech CSE - Data Structures

## Binary Search Tree (BST) for Buildings — Complete Notes

### STEP 1 — BST Node Structure

Each node holds:

- BuildingID (used for ordering)
- BuildingName
- LocationDetails
- left child pointer
- right child pointer

### Node Structure (Language-Neutral)

```
class TreeNode:
    int BuildingID
    string BuildingName
    string LocationDetails
    TreeNode* left
    TreeNode* right
```

### STEP 2 — BST Insertion Logic

Insert LEFT if newID < node.BuildingID

Insert RIGHT if newID > node.BuildingID

**Pseudocode:**

```
function insert(node, data):
    if node is NULL:
        return new Node(data)
    if data.BuildingID < node.BuildingID:
        node.left = insert(node.left, data)
    else:
        node.right = insert(node.right, data)
    return node
```

**Example BST (40,20,60,10,30,50,70):**

```
        40
     /          \
    20         60
   /   \     /    \
 10   30  50   70
```

### STEP 3 — Searching

```
function search(node, keyID):
    if node is NULL: return "Not Found"
    if keyID == node.BuildingID: return node
    if keyID < node.BuildingID: return search(node.left, keyID)
    else: return search(node.right, keyID)
```

### STEP 4 — Traversals

**Inorder (sorted):**

10, 20, 30, 40, 50, 60, 70

**Preorder:**

40, 20, 10, 30, 60, 50, 70

**Postorder:**

10, 30, 20, 50, 70, 60, 40