HABIB UNIVERSITY

Data Structures & Algorithms

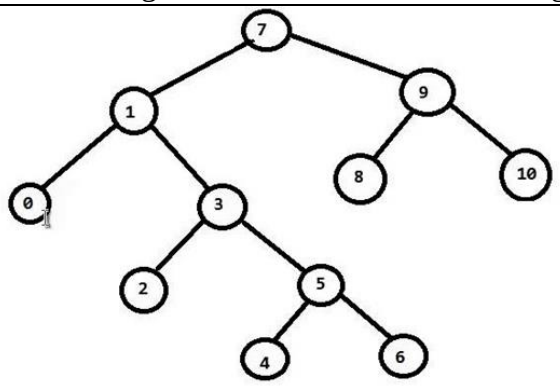
CS/CE 102/171 Spring 2023

Instructor: Maria Samad

Searching Binary Search Trees

Student Name: _____

For the given trees, answer the following:



Search Node 5:
Root: 7 → Left: 1 → Right: 3 → Right: 5
Successfully Found!

Search Node 11:
Root: 7 → Right: 9 → Right: 10 → Right: Null
Not found

Minimum:
Root: 7 → Left: 1 → Left: 0 → Left: Null
Therefore, minimum = Node 0

Maximum:
Root: 7 → Right: 9 → Right: 10 → Right: Null
Therefore, maximum = Node 10

Successor of Node 3:

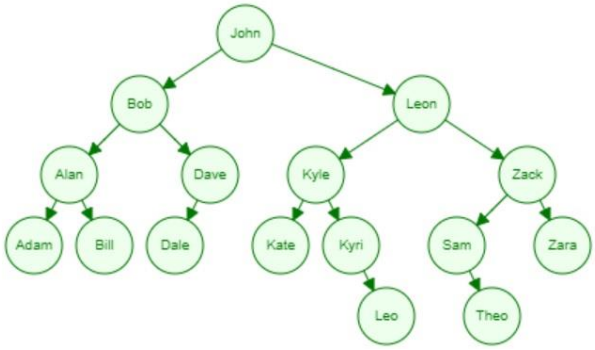
- **Node 3 exists, and has a right subtree**
- **Successor of Node 3 will be the left most child of its right subtree**
- **Start: 3 → Right: 5 → Left: 4 → Left: Null**

Therefore, Successor of Node 3 = Node 4

Predecessor of Node 7:

- **Node 7 exists, and has a left subtree**
- **Predecessor of Node 7 will be the right most child of its left subtree**
- **Start: 7 → Left: 1 → Right: 3 → Right: 5 → Right: 6 → Right: Null**

Therefore, Predecessor of Node 7 = Node 6



Search Kyri:
Root: John → Right: Leon → Left: Kyle → Right: Kyri
Successfully Found!

Search James:
Root: John → Left: Bob → Right: Dave → Right: Null
Not found

Minimum:
Root: John → Left: Bob → Left: Alan → Left: Adam → Left: Null
Therefore, minimum = Adam

Maximum:
Root: John → Right: Leon → Right: Zack → Right: Zara → Right: Null
Therefore, maximum = Zara

Successor of Sam:

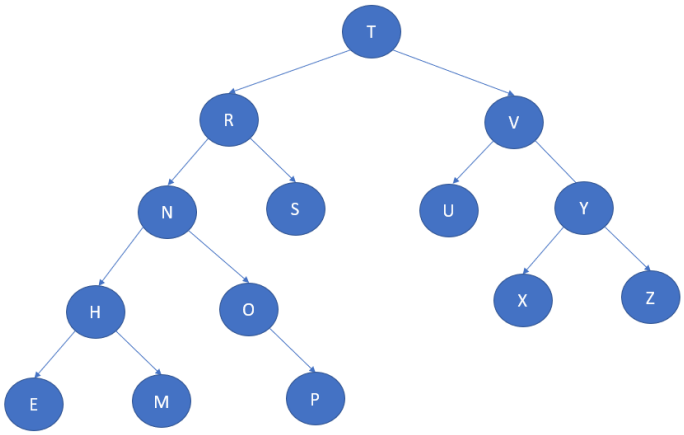
- **Sam exists, and has a right subtree**
- **As there is only one right child, it will become its successor**
- **Start: Sam → Right: Theo → Left: Null**

Therefore, Successor of Sam = Theo

Predecessor of Dave:

- **Dave exists, and has a left subtree**
- **As there is only one left child, it will become its predecessor**
- **Start: Dave → Left: Dale → Right: Null**

Therefore, Predecessor of Dave = Dale



Search Node O:
Root: T → Left: R → Left: N → Right: O
Successfully found

Search Node U:
Root: T → Right: V → Left: U
Successfully found

Minimum:
Root: T → Left: R → Left: N → Left: H → Left: E → Left: Null
Therefore, minimum = Node E

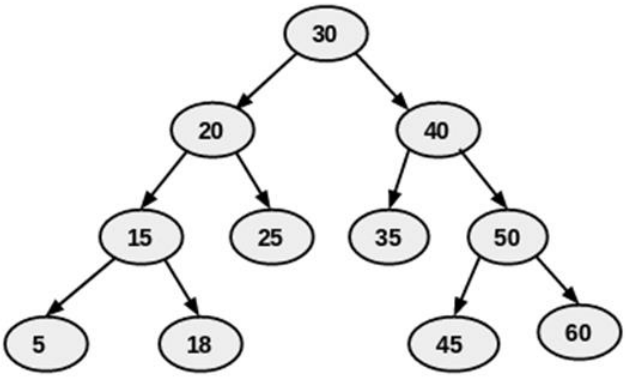
Maximum:
Root: T → Right: V → Right: Y → Right: Z → Right: Null
Therefore, maximum = Node Z

Successor of Node P:

- **Node P exists, and is a leaf node**
- **As it is a leaf node, its successor will be its ancestor that has immediately bigger value than P**
- **Start: P → Parent: O → Parent: N → Parent: R**
- **As R is bigger than P, then:**
  - **Successor of Node P = Node R**

Predecessor of Node X:

- **Node X exists, and is a leaf node**
- **As it is a leaf node, its predecessor will be one of its ancestors whose value is immediately smaller than X**
- **Start: X → Parent: Y → Parent: V**
- **As V is smaller than X, then:**
- **Predecessor of Node X = Node V**



Search Node 19:
Root: 30 → Left: 20 → Left: 15 → Right: 18 → Right: Null
Not found

Search Node 48:
Root: 30 → Right: 40 → Right: 50 → Left: 45 → Right: Null
Not found

Minimum:
Root: 30 → Left: 20 → Left: 15 → Left: 5 → Left: Null
Therefore, minimum = Node 5

Maximum:
Root: 30 → Right: 40 → Right: 50 → Right: 60 → Right: Null
Therefore, maximum = Node 60

Successor of Node 60:

- **Node 60 exists, and is a leaf node**
- **As it is a leaf node, its successor will be its ancestor that has immediately bigger value than 60**
- **Start: 60 → Parent: 50 → Parent: 40 → Parent: 30 → Parent: Null**
- **All ancestors have been checked, and there is no value bigger than 60 for any ancestor, therefore:**
  - **Successor of Node 60 = Does Not Exist**

Predecessor of Node 5:

- **Node 5 exists, and is a leaf node**
- **As it is a leaf node, its predecessor will be its ancestor that has immediately smaller value than 5**
- **Start: 5 → Parent: 15 → Parent: 20 → Parent: 30 → Parent: Null**
- **All ancestors have been checked, and there is no value smaller than 5 for any ancestor, therefore:**
  - **Predecessor of Node 5 = Does Not Exist**