Mohamed AbdelHalem_1180182

Assignment_5

## Problems:

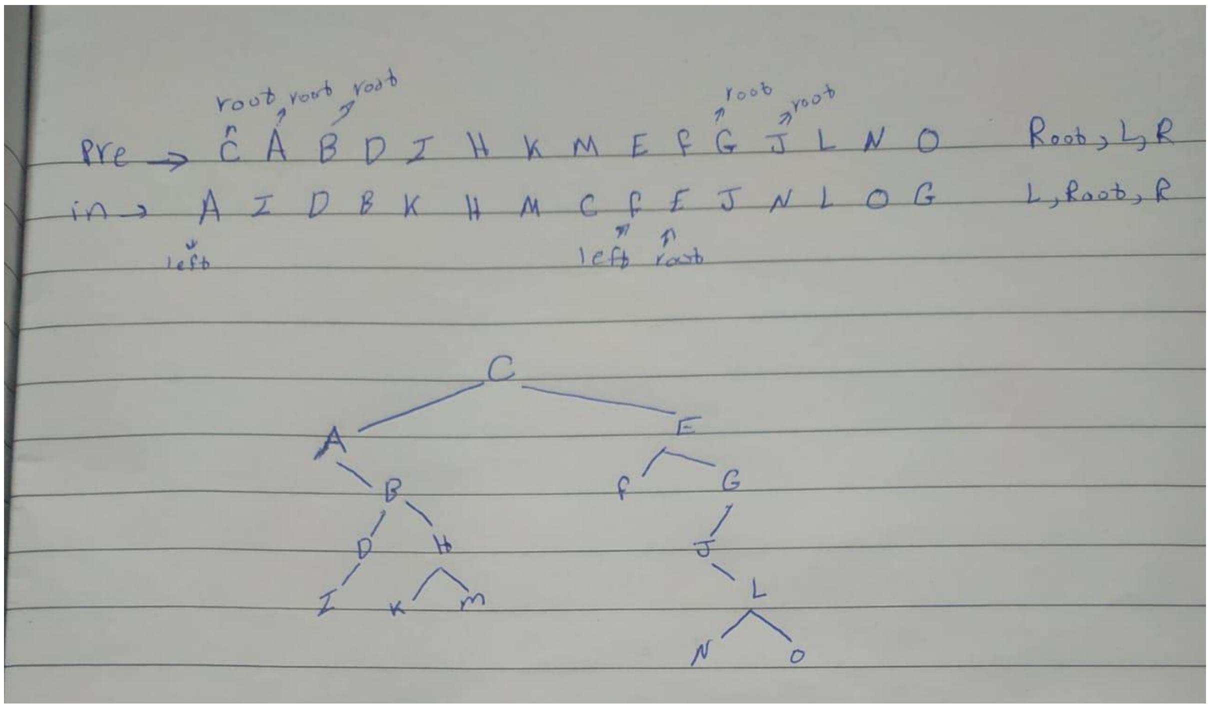
1)

*In order traversal: 4,2,5,1,3

*Pre order traversal: 1,2,4,5,3

*Post order traversal: 4,5,2,3,1

2)



post order traversal: I D K M H B A F N O L J GEC

3,4) I first created class Node and initialized constructor with key and left and right children as it's a BST then I created insertion function to test the min function with it and I created it with first checking if there is nothing we return a new node which is root else put the smaller values on left and larger on right finally return the node pointer then

For 3 : I created a function that keeps checking left until it’s left is null and returns it which is the min value as left most contains smallest value

For 4: I created a function that checks left of current and if it’s none return current else enter the function again giving it currnet’s left

Complexity: O(N)

Pseudo code:

1)loop

```
Minloop(node)
Current=node
While(current.left is not None):
Current=current.left
Finally return current.data which is now left most
```

2)Recursive

```
MinRecursion(node)
Current=node
If current.left is None:
Return current.data
Else:
Return MinRecursion(current.left)
```

5)

| Operation | Balanced BST | Unbalanced BST |
|---|---|---|
| Search | O(log(N)) | O(N) |
| insertion | O(log(N)) | O (N) |
| Deletion | O(log(N)) | O (N) |

For balanced we simply go in one direction left or right depending on the value we compare that’s why it’s O(log(N))

But for unbalanced we need to go through the whole tree in the worst case to insert an element , delete or search.

Sources used in this problem: https://www.geeksforgeeks.org/find-the-minimum-element-in-a-binary-search-tree/