BST – 2

1. **Find path in BST**

**Send Feedback**

Given a BST and an integer k. Find and return the path from the node with data k and root (if a node with data k is present in given BST). Return null otherwise.

Assume that BST contains all unique elements.

**Input Format :**

Line 1 : Elements in level order form (separated by space)

(If any node does not have left or right child, take -1 in its place)

Line 2 : Integer k

Output Format :

Path from node k to root

**Sample Input :**

8 5 10 2 6 -1 -1 -1 -1 -1 7 -1 -1

2

**Sample Output :**

2

5

8

1. **BST Class**

**Send Feedback**

Implement the BST class which includes following functions -

**1. search**

Given an element, find if that is present in BSt or not. Return true or false.

**2. insert -**

Given an element, insert that element in the BST at the correct position. Assume unique elements will be given.

**3. delete -**

Given an element, remove that element from the BST. If the element which is to be deleted has both children, replace that with the minimum element from right sub-tree.

**4. printTree (recursive) -**

Print the BST in ithe following format -

For printing a node with data N, you need to follow the exact format -

N:L:x,R:y

wherer, N is data of any node present in the binary tree. x and y are the values of left and right child of node N. Print the children only if it is not null.

There is no space in between.

You need to print all nodes in the recursive format in different lines.

**Note : main function is given for your reference which we are using internally to test the class.**

1. **Insertion In BST**

**Send Feedback**

Which one of the following is the tightest upper bound that represents the time complexity of inserting an object into a binary search tree of n nodes?

1. O(1)
2. O(logn)
3. O(n) answer
4. O(logn)
5. **Insertion/Deletion In BST**

**Send Feedback**

What are the worst-case complexities of insertion and deletion of a key in a binary search tree?

1. θ (log(n)) for both insertion and deletion
2. θ (n) for both insertion and deletion answer
3. θ (n) for insertion and θ (log ) n for deletion
4. θ (log (n)) for insertion and θ (n) for deletion