Data Science & Artificial Intelligence

Data Structures Through Python

Trees

DPP- 02

Discussion Notes

By- Satya Sir

[NAT]

ANS: 509

#Q. Suppose a binary search tree with 1000 distinct elements is also a complete binary tree. The tree is stored using the array representation of binary heap trees. Assuming that the array indices start with 0, the 3rd largest element of the tree is stored at index_____.

BST: Left Subtree < Parent < Right Subtree

Big < Bigger < Biggest

3rd largest | 2nd largest | 1st largest

3rd largest Element's Parent index

1+2+4+8+16+32+64+128+256+512 = 1023

From Root it is at level 8

511 Elements [0 ---- 510]

489 Elements

2i+2 510 ⇒ Right child

2i+1 509 ⇒ left child

⇒ 2i+1 = 509

⇒ 2i = 508

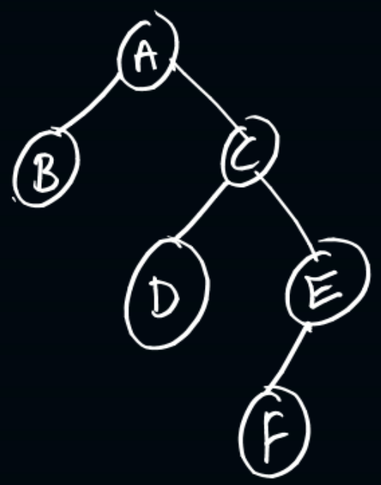
i = 254

[MCQ]

#Q. What will be post order traversal of a binary Tree T, if preorder and in order traversals of T are given by ABCDEF and BADCFE respectively?

Pre order

In order



Post order: B D F E C A

(LRP)

A BEFDCA

B BFDECA

C BCFDEA

D ✓ BDFECA

[MCQ]

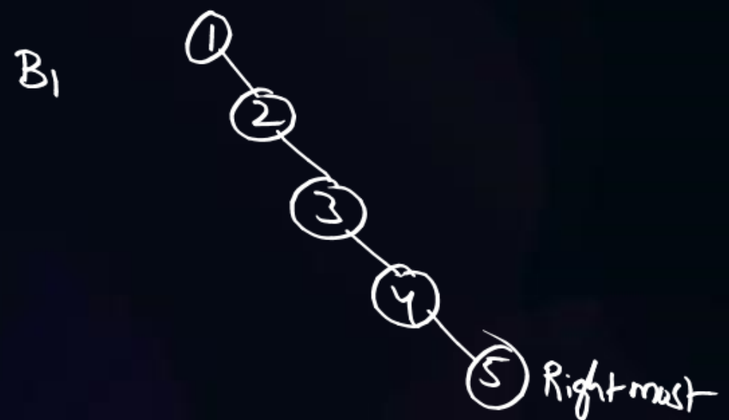
#Q. We create a binary search tree B_1 by inserting the numbers 1,2, 3, 4, 5 into an empty binary search tree. We create another binary search tree B_2 by inserting the numbers into an empty binary search tree in the reverse order. What is the difference between the right-most element of B_1 and the left-most element of B_2?

A 1

B 3

C ✓ 4

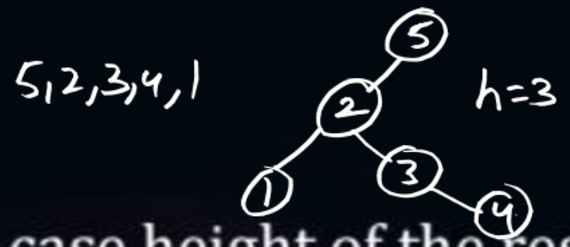
D 2



5 - 1 = 4

[MCQ]

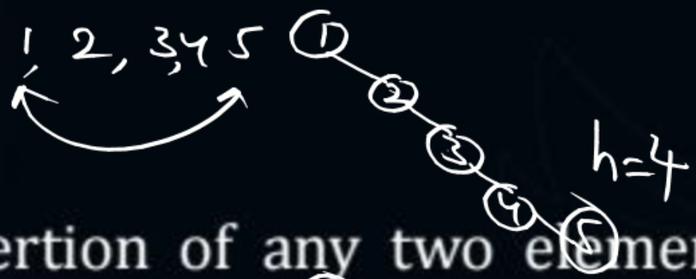
#Q. Consider that N distinct elements (N > 3) are inserted into an initially empty binary search tree (BST). Which of the following statements are true?



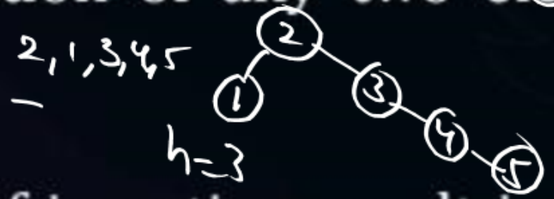
A ✗ The worst case height of the resulting BST is $\log_2 N$

Worst Case: Skewed/DBT: O(n)

B ✗ None of the above



C ✗ Swapping the order of insertion of any two elements can always half the height of the resulting BST.



D ✓ Consider that a given order of insertion result in a BST of height N. One can always find two elements in the original where swapping the order of insertion of the two elements can half the height of the resulting BST.

[MCQ]

#Q. In searching an element in a binary tree, number of comparisons is

A O (log n -1)

B O (log n -2)

C O (log n)

D O (n log n)

[MCQ]

#Q. What is a Binary Search Tree (BST)?

A. A tree in which each node has at most two children

B. A tree in which all the nodes are arranged in increasing order

C. A tree in which the left child is always smaller than the parent and the right child is always larger ✓

D. A tree in which each node has exactly two children

[MCQ]

#Q. What is the time complexity of searching for an element in a balanced BST?

O(log n)

A O(1)

B O(n)

C O (log n)

D O (n log n)

[MCQ]

#Q. Given the following Python function, what does it do?

```
def insert(root, key):
    if root is None:
        return Node(key)
    else:
        if root.val < key:
            root.right = insert(root.right, key)
        else:
            root.left = insert(root.left, key)
    return root
```

key = 10

70<10 Fals.

A Deletes a node from the BST

B Searches for a node in the BST

C ✓ Inserts a node into the BST

D Finds the minimum value in the BST

[MCQ]

#Q. Which of the following statements is true for a BST?

A. In order traversal of a BST yields the keys in increasing order

B. Preorder traversal of a BST yields the keys in decreasing order

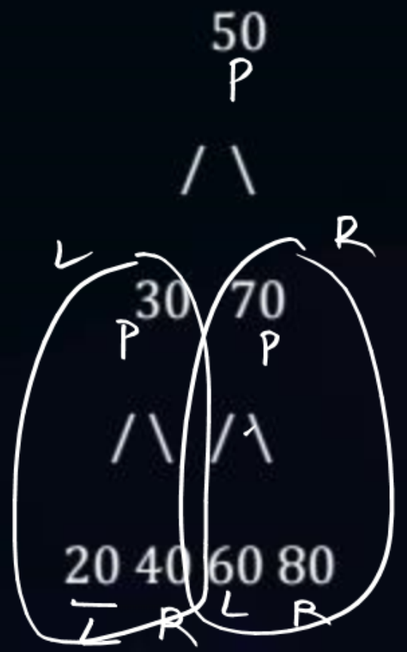
C. Post order traversal of a BST yields the keys in increasing order

D. Level order traversal of a BST yields the keys in random order

[MCQ]

(LPR)

#Q. What will be the in order traversal of the following BST?



20, 30, 40, 50, 60, 70, 80

A 50 30 20 40 70 60 80

B ✓ 20 30 40 50 60 70 80

C 20 40 30 50 60 80 70

D 20 30 50 40 60 70 80

THANK - YOU