6 CHAPTER

Trees

Q.1 T is a search tree of order K, its size is N and its height is H. The computation time needed to insert/delete a data item on T is

(a) $O(\log H)$ (b) $O(N)$
(c) $O(K)$ (d) $O(H)$

Q.2 Which is efficient tree structure, considering space and time complexities?

(a) AVL Tree
(b) Full Binary Tree
(c) Complete Binary Tree
(d) Binary search tree

Q.3 The inorder traversal of some binary tree produced the sequence MFIEJGZ, and post order sequence is HIFJZGE the number of nodes in left subtree of a given tree _____?

Q.4 The degree of a node in a tree is the number of children the node has if a tree has n_1 nodes of degree 1, n_2 nodes of degree 2, . . . n_m nodes of degree m, then number of leaves in the tree in terms of $n_1, n_2 \ldots n_m$

(a) $1 * n_1 + 2 * n_2 \ldots (m-1)n_m$
(b) $n_1 + n_2 + \ldots + n_m$
(c) $(n_m + n_1) * m / 2$
(d) $1 + [1 * n_2 (2-1) + 2 * n_3 \ldots + (m-1) * n_m]$

Q.5 A 2-3 tree is a tree such that (a) all internal nodes have either 2 or 3 children (b) all paths from root to the leaves have same length.

The number of internal nodes of a 2-3 tree having a leaves could be

(*i*) 4 (*ii*) 5
(*iii*) 6 (*iv*) 7

(a) (*i*) and (*iii*) (b) (*i*) and (*iv*)
(c) (*ii*) and (*iii*) (d) (*ii*) and (*iv*)

Q.6 A binary search tree is constructed by inserting the key values 1, 2, 3, 4, 5, 6, 7 in some order specified by a permutation of 1,..., 7 into an initially empty tree. Which of these permutation will lead to a complete binary search tree?

(a) 1, 2, 3, 4, 5, 6, 7 (b) 4, 6, 5, 4, 1, 2, 3
(c) 4, 2, 5, 1, 3, 5, 7 (d) 4, 1, 5, 3, 6, 2, 7

Q.7 Suppose that we have numbers between 1 and 1000 in a binary search tree and want to search for the number 363. Which of the following sequence could not be the sequence of nodes examined?

(a) 2, 252, 401, 398, 330, 344, 397, 368
(b) 924, 220, 911, 244, 898, 258, 362, 363
(c) 925, 202, 911, 240, 912, 245, 363
(d) 2, 399, 387, 219, 266, 382, 381, 278, 278, 363

Q.8 Which of the following statements is/are true.

(*i*) Suppose the search for key 'K' in a binary search tree ends up in a leaf. Consider three sets A, the keys to the left of the search path; B, the keys on the search path; and C, the keys to the right of the search path then any must satisfy $a \le b \le c$.

(*ii*) Operation of deletion is cumulative in the sense that deleting x and they y from a binary search tree leaves the same tree as deleting y and then x.

(a) only (*i*) (b) only (*ii*)
(c) Both true (d) Both false

Q.9 Consider three keys, k_1, k_2, k_3 such that $k_1 < k_2 < k_3$. A binary search tree is constructed with these three keys. Depending on the order in which the keys are instered, number of binary search trees possible are

(a) 3 (b) 5
(c) 6 (d) 4

Q.10 An AVL tree is constructed by inserting the key values 1, 2, 3, 4, 5 in some order specified by a permutation of 1, 2, 3, 4, 5 into an initially empty tree. For which of the following permutation there is no need to do any rotation at any stage during the insertion.

(a) 1, 3, 2, 5, 4 (b) 4, 2, 5, 1, 3
(c) 5, 3, 2, 1, 4 (d) 2, 3, 4, 5, 1

Q.11 Which of the following traversal is sufficient to construct Binary search tree from given traversal?

I. Preorder
I. Inorder
III. Postorder

(a) Any of the given traversal is sufficient
(b) Either I or III is sufficient
(c) I and III
(d) II and III

Q.12 If AVL tree has 15 nodes, what is the minimum and maximum possible height? Assume root is present at height 1?

(a) 4, 8 (b) 4, 5
(c) 3, 7 (d) 3, 8

Q.13 Consider the following code segment

```
struct node
{
    struct node *left;
    int data;
    struct node *right;
};
struct node *fun (struct node *P)
{
    If (P → right → right == NULL)
        P →  right = P → right → left;
    else
        P = fun (P → right);
    return (P);
}
```

What does the function fun do? Assume left subtree and right subtree of Binary search tree is not NULL.

(a) Finds the largest node in the binary search tree
(b) Finds the largest node in the binary search tree and deletes it
(c) Finds the smallest node in Binary search tree
(d) Finds the smallest node in binary search tree and deletes it

Q.14 A binary search tree was constructed by inserting following elements into an initially empty binary tree: 50, 27, 16, 88, 34, 65, 52, 77, 93, 4, 12, 29, 44, 92.

Preorder and postorder traversals of the resultant binary search tree were stored in arrays A and B respectively. How many elements have same index location in both the arrays? [Assume arrays A and B start from the same index]

Q.16 Let T be a binary search tree with 120 elements. What is the smallest possible height of T? Consider root is at height 0.

Q.17 Find the height of a tree for the following given traversals of the tree.

Inorder : *h, d, i, b, e, j, a, f, c, g, k*
Pre-order : *a, b, d, h, i, e, j, c, f, g, k*

Q.18 Let T be a K-ary tree (each internal node of T has at most K children). Suppose that the maximum depth of any node of T is d. What is the maximum number of leaves that T can have? [Assume root is at depth 0]

(a) K (b) Kd
(c) K^d (d) d^K

Q.19 If a tree has n_1 nodes of degree 1, n_2 nodes of degree 2 and n_3 nodes of degree 3, then which of the following holds?

(a) $n_1 = 2$ (b) $n_1 = n_3 + 2$
(c) $n_1 = n_3 - 1$ (d) $n_1 = n_2 + n_3$

Q.20 Consider the following recursive function.

```
bool f(Struct node *P)
{
    if(P = NULL) return TRUE;
    if(P → Left != NULL && Max(P → Left) > P →
    data)
        return FALSE;
    if(P → right != NULL && Min(P → right) <=
    P → data)
```

```
        return FALSE;
    if(! f(P → Left) || ! f(P → right))
        return FALSE;
    return TRUE;
}
```

Assume Max(*q*) function returns the maximum value from *q* and subtrees of *q*, Min(*q*) function returns the minimum value from *q* and subtrees of *q*. If root of the binary tree is passed to the function *f*( ), then what is the functionality of the above code?

(a) It checks if a given tree is a binary search tree or not.
(b) It checks if a given tree is a heap tree or not.
(c) Both (a) and (b)
(d) Neither (a) nor (b)

Tree

1) Tree → T, order → k, size → N, height → H

1) Time to insert/delete – ?

→ O(H) Ans

2) Efficient tree structure, considering space & time complexity

→ Complete binary tree Ans

3) Inorder → MFIEJGZ, postorder → HIFJZGE

→ HFIEJGZ

3) No. of node in left-subtree = 3 Ans

4) Tree(n_0) → degree(1), Tree(n_2) → degree(2)

⋮

D) Tree(n_m) → m, no. of leaves in tree

→ Sum of degree = $1 * n_1 + 2 * n_2 + \cdots m * n_m$

Total = $n_0 + n_1 + n_2 + \cdots n_m$

edge = node − 1

So, $(n_0 + n_1 + \cdots n_m) - 1 = (1 * n_1 + 2 * n_2 + \cdots m * n_m)$

$n_0 = 1 + [1 * n_2 (2-1) + 2 * n_3 - \cdots + (m-1) * n_m]$ Ans

5 → No. of internal node of 2-3 tree having 4 levels could be —

B

4, 5, 6, 7

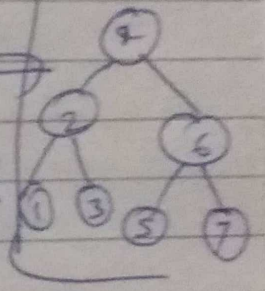
4, 7 Ans

6 → Key value — 1, 2, 3, 4, 5, 6, permutation — 1 to 7

D

complete binary search tree

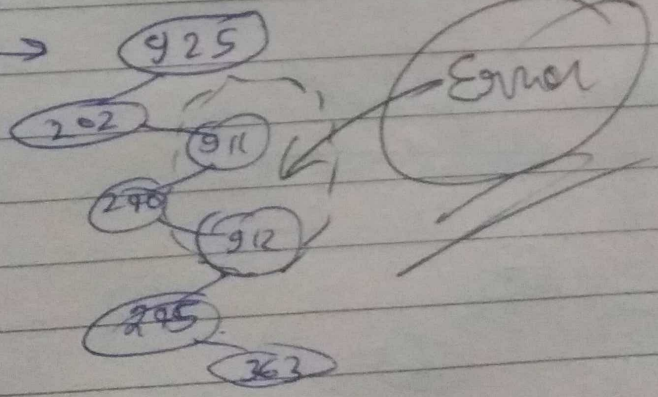
4, 2, 5, 1, 3, 5, 7 Ans



7 → number b/w 1 — 1000, search → 363

which one not nodes examined ?

C



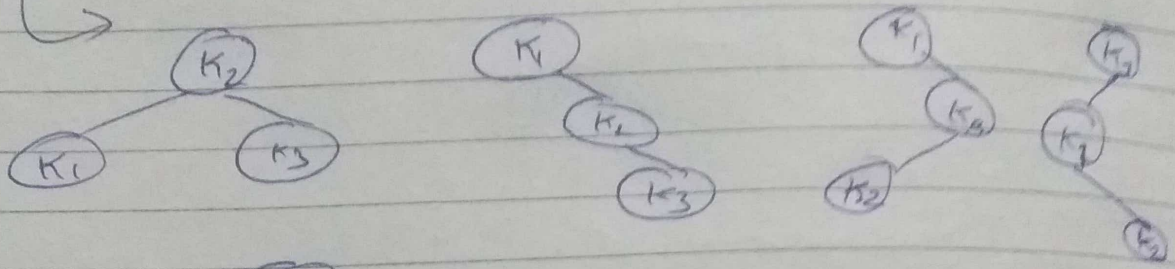
8 → Three statements

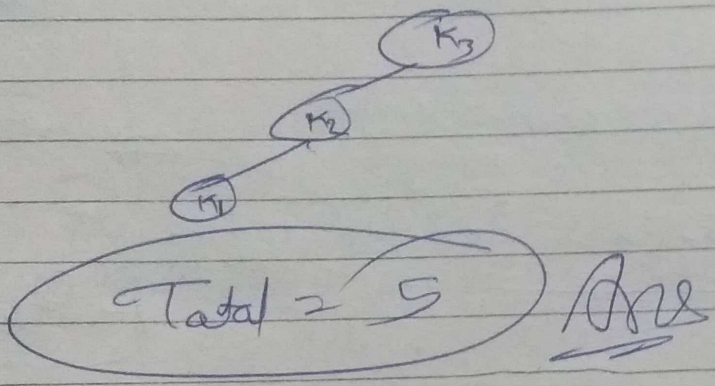
D

Both statement false

(9) Key → K_1, K_2, K_3, $K_1 < K_2 < K_3$

(B) No. of Binary search tree possible —



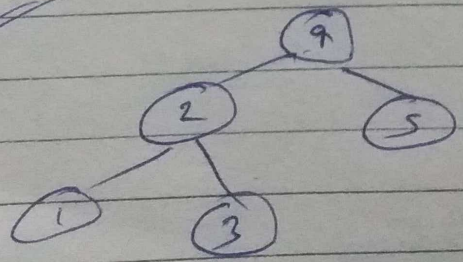


Ans

(10) AVL Tree — Key — 1, 2, 3, 8, 5, 6

No rotation required

(B)



Ans

11) construct BST

8

I Preorder
II Inorder → Always sorted list
III postorder

→ Either I or III Ans

12)

8

AVL Tree → 15 nodes
min & max height

→ $\lfloor \log_2(15) \rfloor = 4$

$\lceil \log_2(15) \rceil = 5$

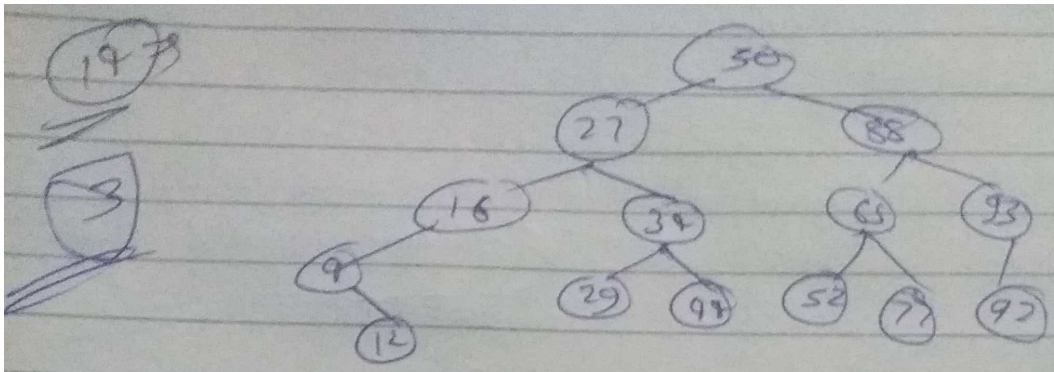
⇒ 4,5 Ans

13)

8

```
struct node *fun(struct node *p)
{ if (p->right->right == NULL)
      p->right = p->right->right;
  else
      p = fun(p->right);
  return(p);
}
```

→ Find largest node & delete Ans



preorder → 50, 27, 16, 9, 12, 34, 29, 97, 88, 65, 52, 77, 93, 92

postorder → 12, 9, 16, 29, 97, 34, 27, 52, 77, 65, 92, 93, 88, 50

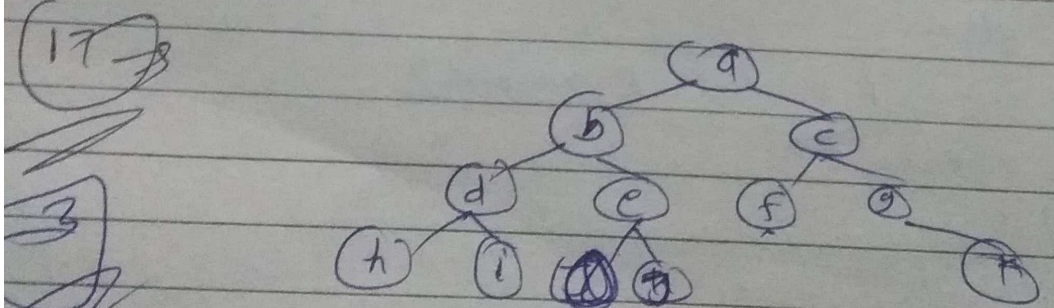
16, 34, 65 → Same location

15 → No Que

Tree → 120 elements

↳ Smallest height = $\lceil \log_2^{(120+1)} \rceil - 1$

$= 7 - 1 = 6$ Ans



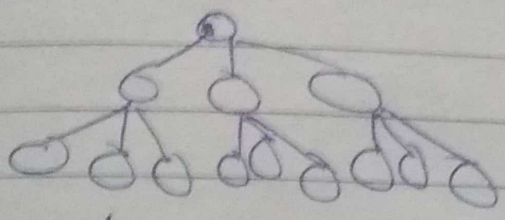
Height = 3 Ans

18) K-ary tree , depth → d

max no. of leaves ?

D

Let K = 3



d=0 , $3^0 = 1$

d=1 , $3^1 = 3$

d=3 , $3^2 = 9$ leaf node

→ $3^{depth} = K^d$ Ans

19)

B

Total node = $n_1 + n_2 + n_3$

Total edge = ~~$e_1 + e_2 + e_3$~~ Total node − 1

= $(n_1 + n_2 + n_3) - 1$

Sum of degree = 2e

$1 \times n_1 + 2 \times n_2 + 3 \times n_3 = 2e$

$n_1 + 2n_2 + 3n_3 = 2(n_1 + n_2 + n_3 - 1)$

$n_1 = n_3 + 2$ Ans

20)

A

gf root node pass in f^n of C.

→ It checks if a given tree is BST or not

Ans