CS & IT ENGINEERING

Data Structure

Trees

DPP

Discussion Notes

By- Pankaj Sharma sir

01 Question

02 Discussion

Q.1 A binary tree has 1024 leaves. The number of nodes in the tree having two children is __1023__. **[NAT]**

Ex1

leaf nodes = 1

no. of nodes with 2 child = 0

Ex2.

leaf node = 2

nodes with 2-childs = 1

Ex2.

leaf node = 1

nodes with 2-child = 0

nodes with 2-childs = # leaf nodes − 1

1024 − 1 = 1023

Q.2

The height of a tree is the length of the longest root-to-leaf path in it. The maximum and minimum number of nodes in a binary tree of height 9 are-

[MCQ]

A. 1024, 9

B. 1023, 10

C. 511, 9

D. 512, 10

$h=9$

$n_{min} = h+1$

$n_{max} = 2^{h+1}-1$

$n_{min} = 9+1 = 10$

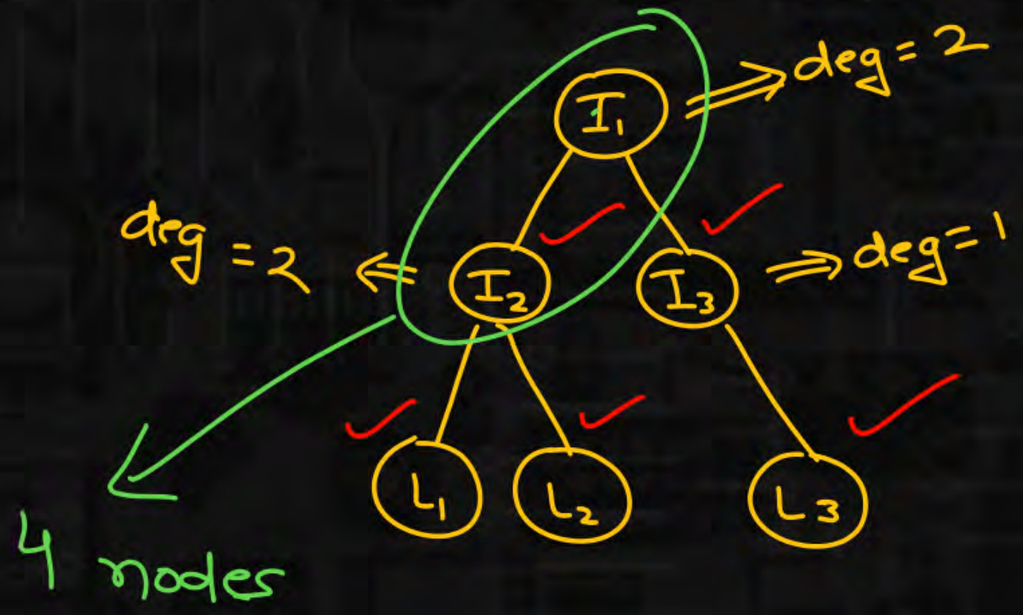
$n_{max} = 2^{9+1}-1 = 2^{10}-1 = 1024-1 = 1023$

1023, 10

Q.3

In a binary tree, the number of internal nodes of degree 1 is 6, and the number of internal nodes of degree 2 is 12. The number of leaf nodes in the binary tree is _____.

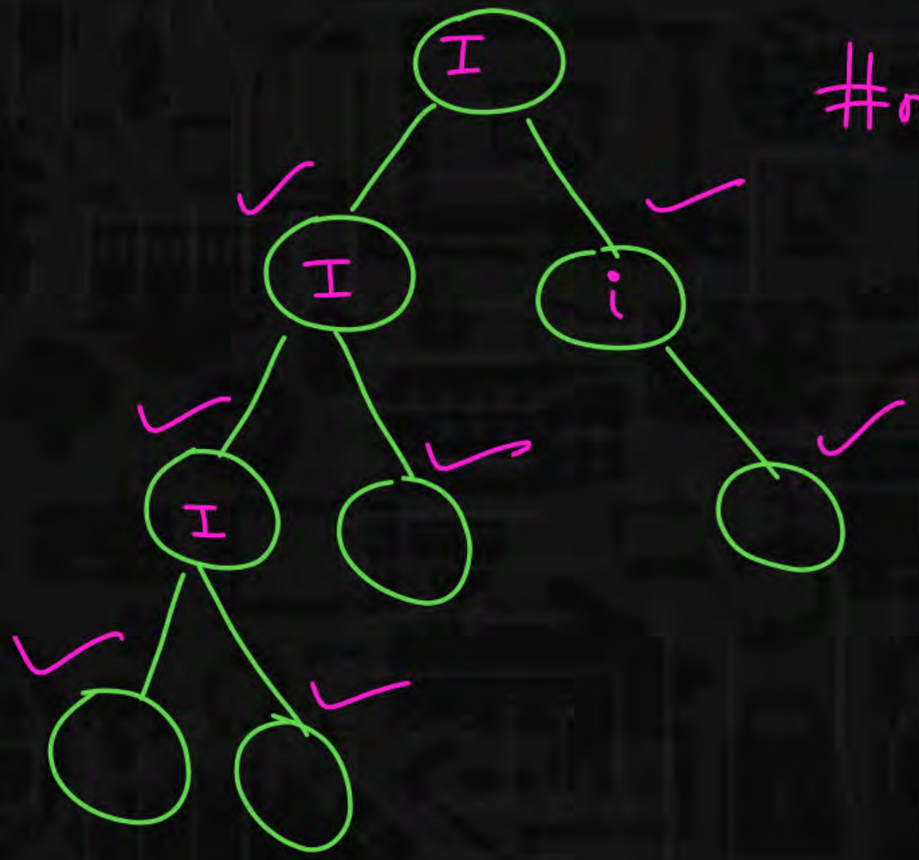
[NAT]



$2 + 2 + 1$

$\#\text{nodes} = 2 + 2 + 1 + 1$ (Root)

Q.3 In a binary tree, the number of internal nodes of degree 1 is 6, and the number of internal nodes of degree 2 is 12. The number of leaf nodes in the binary tree is _____. **[NAT]**



$\#nodes = 3 \times 2 + 1 \times 1 + 1$

(3 × 2: no. of nodes of deg 2; 1 × 1: no. of nodes of deg 1)

$3 + 1 + 4$

Q.3

In a binary tree, the number of internal nodes of degree 1 is 6, and the number of internal nodes of degree 2 is 12. The number of leaf nodes in the binary tree is 13.

[NAT]

Total nodes = $12 \times 2 + 6 \times 1 + 1$

$= 24 + 6 + 1$

Total nodes = 31

$12 + 6 +$ Leaf nodes $= 31$

leaf nodes $= 31 - 18$

$= 13$

Q.4

A strict k-ary tree T is a tree that contains exactly 0 or k children. The number of leaf nodes in tree T if there are exactly 'p' internal nodes is- [MCQ]

Strict 3-ary tree : 0 child (leaf), 3 child (Internal)

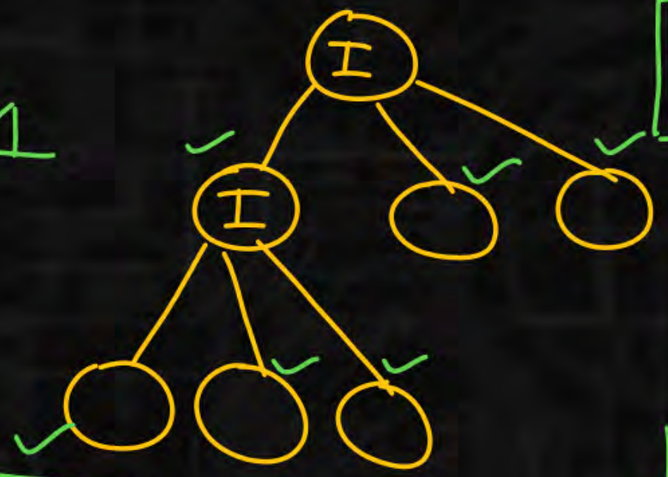
A. $(k-1)p+1$

B. $pk+1$

C. $pk+1+p$

D. None

#nodes = 2 × 3 + 1



#nodes = P × K + 1

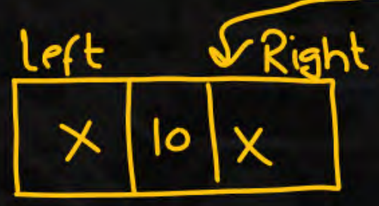
#nodes = P × 3 + 1

$L + P = PK + 1$

$L = PK + 1 - P$

$L = P(K-1) + 1$

Q.5 A linked list is used to store a binary tree with 1024 nodes. The number of null pointers present is 1025. **[NAT]**



nodes = 1

Null pointer = 2

Null pointers = $n+1$

$= 1024+1$

$= 1025$

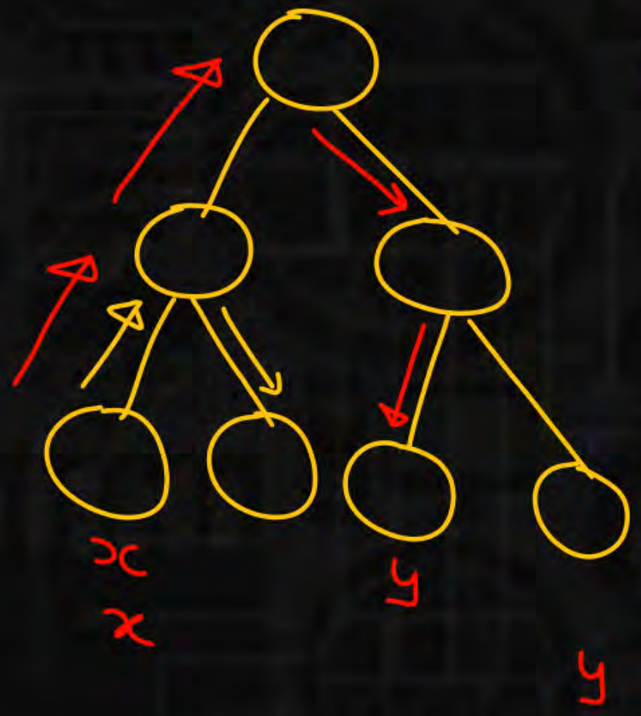
Q.6

Let T be a full binary tree with 4 leaves. (A full binary tree has every level full). Suppose two leaves x and y of T are chosen uniformly and independently at random. The expected value of the distance between x and y in T (i.e., the number of edges in the unique path between x and y) is (rounded off to 2 decimal places) ____.

[NAT]

4X4 =16

Path length	ways	P(i)
0	4	4/16
2	4	4/16
4	8	8/16



Q.6

Let T be a full binary tree with 4 leaves. (A full binary tree has every level full). Suppose two leaves x and y of T are chosen uniformly and independently at random. The expected value of the distance between x and y in T (i.e., the number of edges in the unique path between x and y) is (rounded off to 2 decimal places)

2.50.

[NAT]

	Path length	ways	P(i)
4×4 =16	0	4	4/16
	2	4	4/16
	4	8	8/16

$$E(i) = \sum i \times P(i)$$

$$= 0 \times \frac{4}{16} + 2 \times \frac{4}{16} + 4 \times \frac{8}{16}$$

$$= \frac{8}{16} + \frac{32}{16} = 2.5$$

∴

Q.7 The number of leaf nodes in a rooted tree of n nodes, with each node having 0 or 2 children is- [MCQ]

A. $\frac{n+1}{2}$

B. $\frac{n-1}{2}$

C. $\frac{n}{2}$

D. $n-1$

Every internal node = 2 childs

I : # of internal nodes
L : # of leaf nodes

Total no of nodes = $I \times 2 + 1$

$$L + I = 2I + 1$$

$$n = L + I$$

$$\Downarrow$$

$$I = n - L$$

$$L = I + 1$$
$$= n - L + 1$$
$$L = n - L + 1$$
$$2L = n + 1 \Rightarrow L = \frac{(n+1)}{2}$$

Q.1 Consider the following nested representation of binary trees: (X Y Z) indicates Y and Z are the left and right sub stress, respectively, of node X. Note that Y and Z may be NULL, or further nested. Which of the following represents a valid binary tree?

[MCQ]

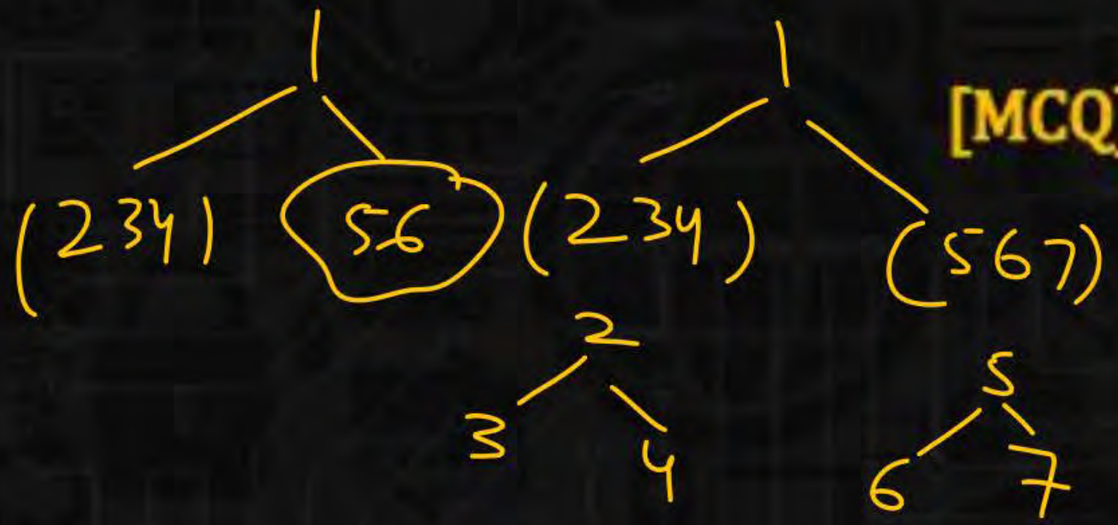
D

A. (1 2 (4 5 6 7))

B. (1 (2 3 4) 5 6) 7)

D. (1 (2 3 4) (5 6 7))

C. (1 (2 3 NULL) (4 5))



Q.2

Consider the following two statements:

S1: It is possible to construct a binary tree uniquely whose post-order and pre-order traversals are given. No

S2: It is possible to construct a binary tree uniquely whose in-order and pre-order traversals are given. ✓

S3: It is possible to construct a binary tree uniquely whose post-order and level-order traversals are given. ✗

Which of the following statement(s) IS/ARE INCORRECT?

[MCQ]

A S1 only

B S2 only

D S1 and S3

C S3 only

Q.3

Let LASTPOST, LASTIN and LASTPRE denote the last vertex visited in a postorder, inorder and preorder traversal respectively, of a complete binary tree. Which of the following is always true?

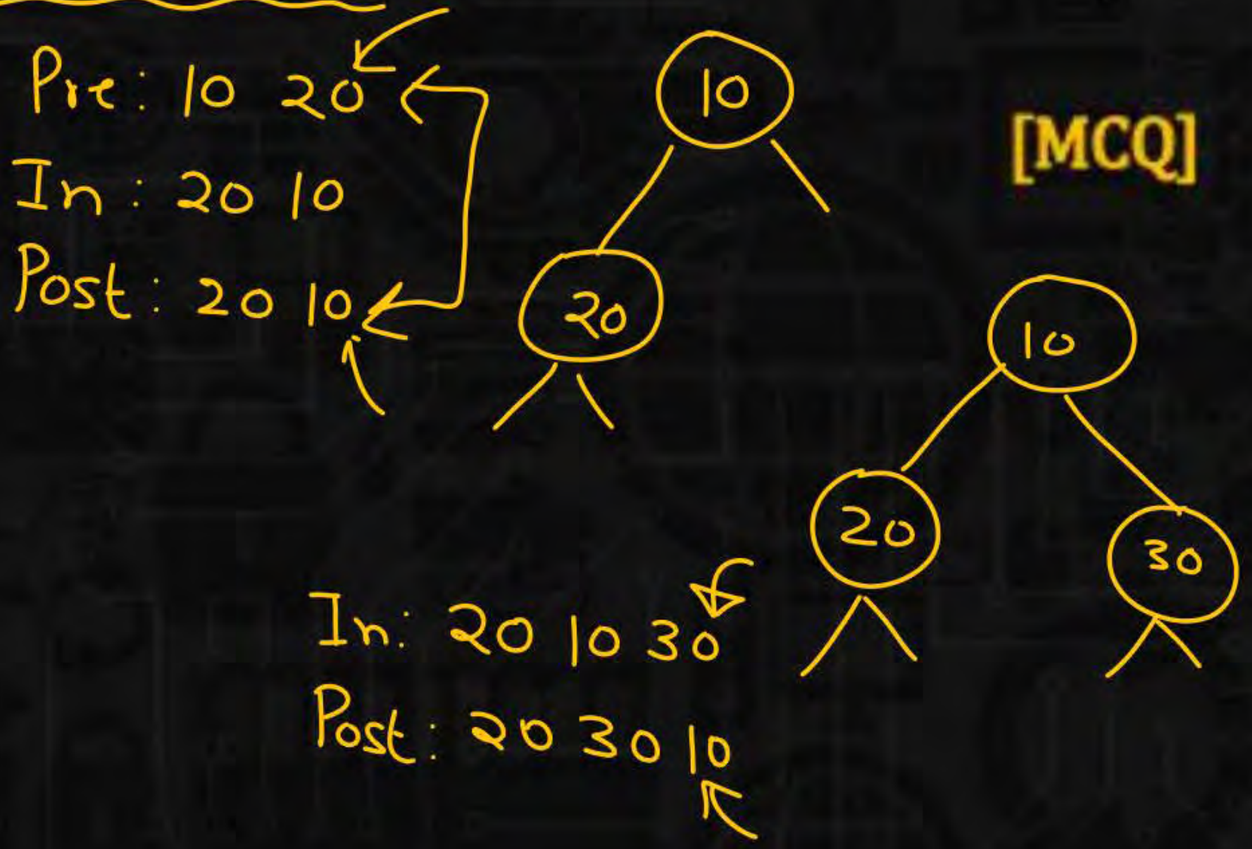
[MCQ]

A. LASTIN = LASTPOST

B. LASTIN = LASTPRE

D. LASTPRE = LASTPOST

C. None of the above



Q.4

Consider the following binary tree T-

41673258

The in-order traversal of T is-

[MCQ]

A 7 1 3 4 6 5 2 8

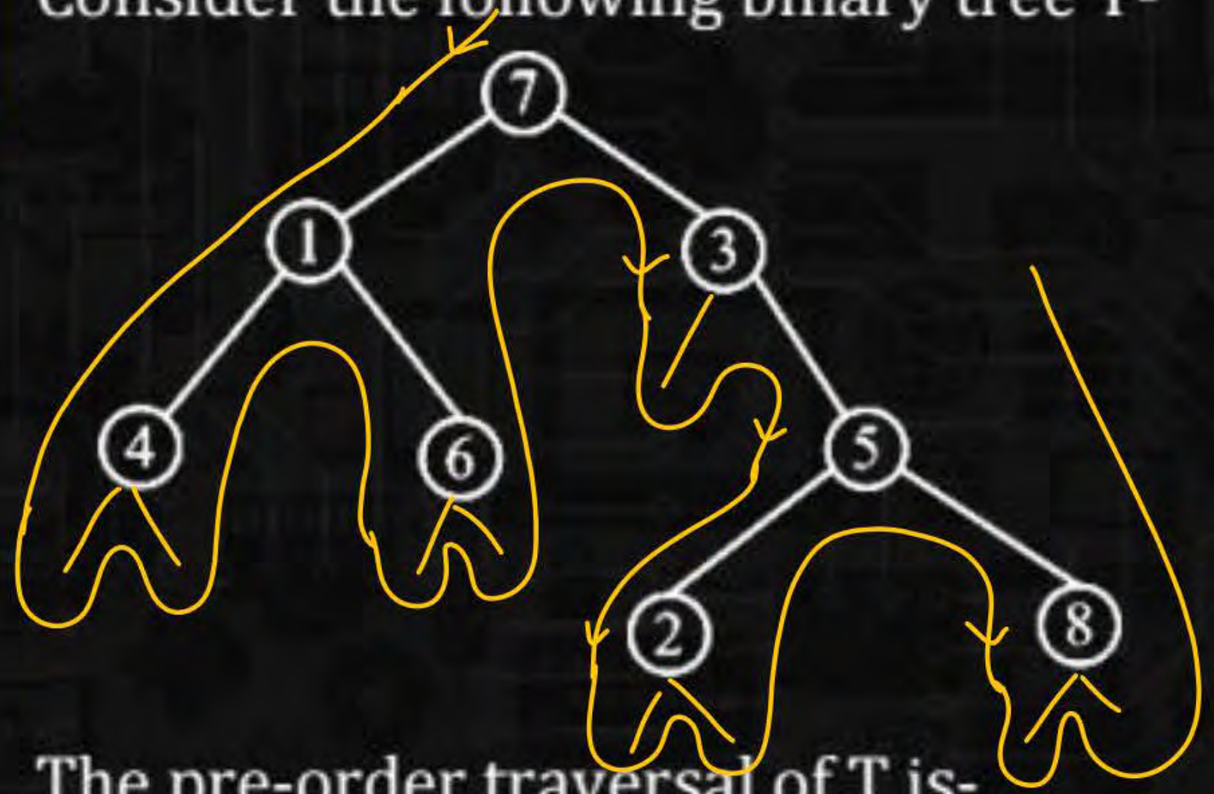
B 4 1 6 7 3 2 5 8

D 4 6 1 2 8 5 3 7

C 7 1 4 6 3 5 2 8

Q.5

Consider the following binary tree T-



The pre-order traversal of T is-

[MCQ]

7 1 4 6 3 5 2 8

A 7 1 3 4 6 5 2 8

B 4 1 6 7 3 2 5 8

D 4 6 1 2 8 5 3 7

C 7 1 4 6 3 5 2 8

Q.6

Consider the following binary tree T-

The post-order traversal of T is-

4 6 1 2 8 5 3 7

[MCQ]

A 7 1 3 4 6 5 2 8

B 4 1 6 7 3 2 5 8

D 4 6 1 2 8 5 3 7

C 7 1 4 6 3 5 2 8

Q.7

The pre-order traversal of a binary tree is 1, 2, 4, 7, 8, 3, 5, 6, 9. The in-order traversal of the same tree is 7 4 8 2 1 5 3 6 9. The height of a tree is the length of the longest path from the root to any leaf. The height of the binary tree above is 3.

[NAT]

Pre : 1 2 4 7 8 3 5 6 9

In : 7 4 8 2 1 5 3 6 9

Q.8

The post-order traversal of a binary tree is 9, 7, 4, 8, 2, 5, 1, 3, 6.
The in-order traversal of the same tree is 9, 7, 8, 4, 5, 2, 6, 3, 1.
The pre-order traversal of the above binary tree is-

[MCQ]

Post : 9 7 4 8 2 5 1 3 6

In : 9 7 8 4 5 2 6 3 1

6 5 8 7 9 4 2 3 1

(A) 1, 2, 4, 7, 9, 8, 5, 3, 6

(B) 1, 2, 4, 7, 8, 9, 5, 3, 6

(D) 1, 2, 3, 4, 5, 6, 7, 8, 9

(C) None of the above.

Q.1 The number of unlabelled binary trees possible with four nodes is 14.

[NAT]

unlabelled binary trees with n nodes $= \frac{{}^{2n}C_n}{n+1}$

$n = 4 \Rightarrow \frac{{}^{8}C_4}{5} = \frac{8!}{5 \times 4! 4!} = \frac{8 \times 7 \times 6 \times 5 \times 4!}{5 \times 4! \times 4!}$

$= \frac{2}{\frac{24}{3}} = 14$

Q.2 The number of labelled binary trees possible with the nodes-10, 30, 25, 40 is 336.

[NAT]

labelled binary trees with n nodes $= \frac{{}^{2n}C_n}{n+1} \times n!$

$n = 4 \Rightarrow \left(\frac{{}^{8}C_4}{4+1}\right) \times 4!$

$= 14 \times 4! = 14 \times 24$

$= 336$

$$\begin{array}{r} 1 \\ 24 \\ \hline 14 \\ \hline 96 \\ 24\times \\ \hline 336 \end{array}$$

Q.3 The number of binary search trees possible with the nodes-10, 30, 25, 40 is 14.

[NAT]

$$\#\text{labelled BST with } n\text{-keys} = \left(\frac{{}^{2n}C_n}{n+1}\right)$$

$$\#\text{ labelled BST with } 10,30,25,40 = \frac{{}^{8}C_4}{4+1} = 14$$

Q.4 [MCQ]

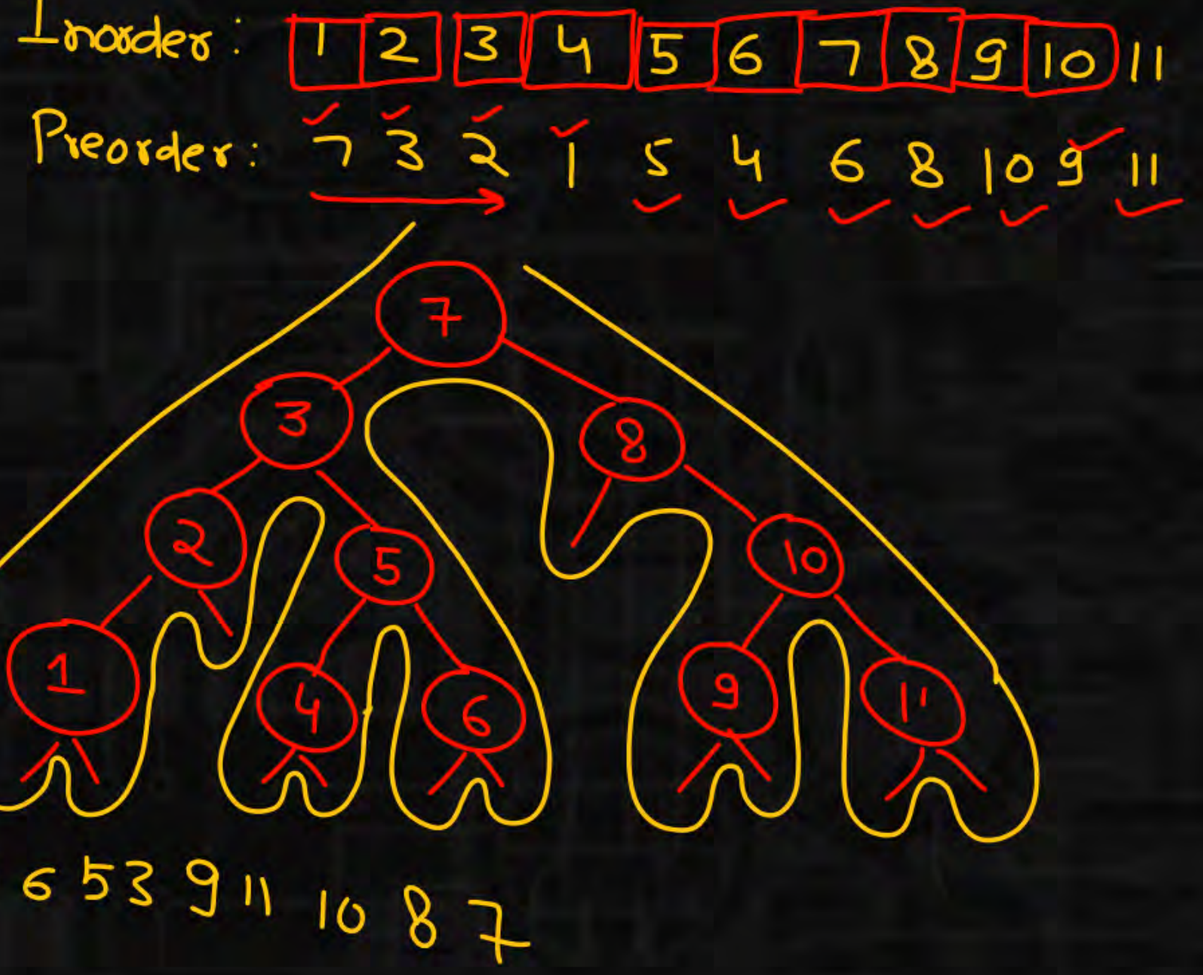
The pre-order traversal of a binary search tree is given as-
7, 3, 2, 1, 5, 4, 6, 8, 10, 9, 11
The post-order traversal of the above binary tree is-

A. 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11

B. 1, 2, 4, 6, 5, 3, 9, 11, 10, 8, 7

C. 1, 2, 4, 5, 6, 3, 9, 10, 11, 8, 7

D. 11, 9, 10, 8, 6, 4, 5, 1, 2, 3, 7



Q.5

[MCQ]

Consider the following two statements:

Statement P: The last elements in the pre-order and in-order traversal of a binary search tree are always same.

Incorrect

Statement Q: The last elements in the pre-order and in-order traversal of a binary tree are always same.

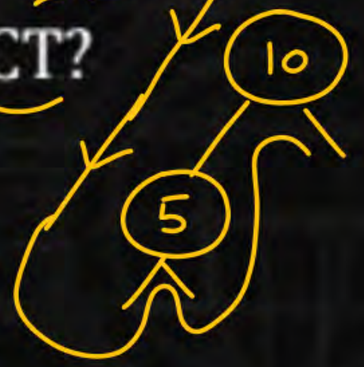
Incorrect

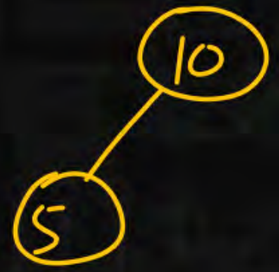
Which of the following tree is/are CORRECT?

Statements

Pre: 10 5

In: 5 10





A. Both P and Q only

B. Neither P nor Q

C. Q only

D. P only

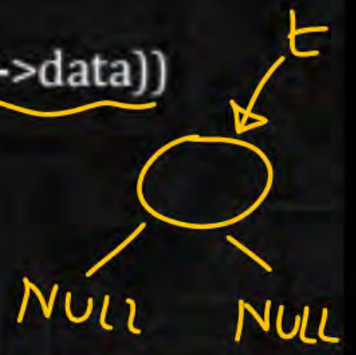
Q.6

[MCQ]

Consider the following function:

```
struct treenode{
struct treenode *left;
int data;
struct treenode *right;
};
```

```
int func (struct treenode *t){
if(t==NULL) return 1;
else if(t->left==NULL && t->right==NULL)
return 1;
else if
((t→left→data < t->data) && (t→right→data > t->data))
return func(t->left) && func(t->right);
else
return 0;
}
```



Assume t contains the address of the root node of a tree. The function-

A. Returns 1 if the given tree is a Binary Search Tree.

B. Returns 0 if the given tree is a complete binary tree.

C. Returns 0 if the given tree is a Binary Search Tree.

D. Returns 1 if the given tree is a complete binary tree.

Q.7

[MCQ]

Consider the following function:

```
struct treenode{
struct treenode *left;
int data;
struct treenode *right;
};
struct treenode * f(struct treenode *t,
int x){
if(t==NULL) return NULL;
elseif(x==t->data) return _____a_____;
else if (x<t->data) return _____b_____;
else return _____c_____;
}
```

Assume t contains the address of the root node of a binary search tree. The function finds an element x in the BST and returns the address of the node if found.

Which of the following is/are CORRECT?

x = 2

x = 12

f(t->left, x)

f(t->Right, x)

A. a: NULL ; b: f(t->left, x) ; c: f(t->right, x)

B. a: t ; b: f(t->right, x) ; c: f(t->left, x)

C. a: NULL ; b: f(t->right, x) ; c: f(t->left, x)

D. a: t ; b: f(t->left, x) ; c: f(t->right, x)

Q.1

Consider the following function:

```
struct treenode{
        struct treenode *left;
        int data;
        struct treenode *right;
  };
  int func(struct treenode *p, struct treenode *q){
  if(p==NULL && q==NULL) return 1;
  if((!p && q) || (!q && p)) return 0;
  return (p->data==q->data) && func(p->left, q->right) && func(p->right, q-
  >left);
  }
```

Initially the addresses of root node of two trees are passed into p and q respectively, the function-

A. Returns 1 iff the two trees are identical.

B. Returns 1 iff the two trees are mirror images of each other.

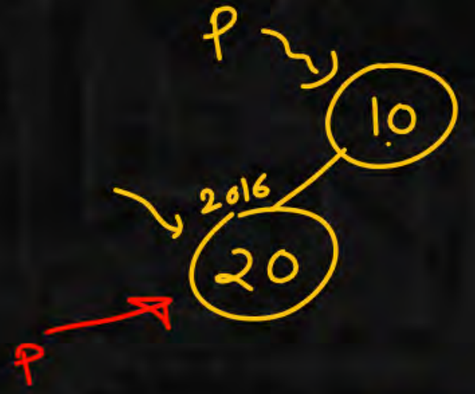
C. Returns 1 iff the two trees emerge from the same root node.

D. None of the above.

Q.1

Consider the following function:

```
struct treenode{
        struct treenode *left;
        int data;
        struct treenode *right;
 };
int func(struct treenode *p, struct treenode *q){
if(p==NULL && q==NULL) return 1;
if((!p && q) || (!q && p)) return 0;
return (p->data==q->data) && func(p->left, q->right) && func(p->right, q-
>left);
}
```



1 && func(2016, 1012)

&& func(NULL, NULL)

func(NULL, NULL) && func(NULL, NULL)

Initially the addresses of root node of two trees are passed into p and q respectively, the function-

A. Returns 1 iff the two trees are identical.

B. Returns 1 iff the two trees are mirror images of each other.

C. Returns 1 iff the two trees emerge from the same root node.

D. None of the above.

Q.1

Consider the following function:

```
struct treenode{
        struct treenode *left;
        int data;
        struct treenode *right;
  };
  int func(struct treenode *p, struct treenode *q){
  if(p==NULL && q==NULL) return 1;
  if((!p && q) || (!q && p)) return 0;
  return (p->data==q->data) && func(p->left, q->right) && func(p->right, q-
  >left);
  }
```

B

Initially the addresses of root node of two trees are passed into p and q respectively, the function-

A. Returns 1 iff the two trees are identical.

B. Returns 1 iff the two trees are mirror images of each other.

C. Returns 1 iff the two trees emerge from the same root node.

D. None of the above.

Q.2

Consider the following function:

```
struct treenode{
        struct treenode *left;
        int data;
        struct treenode *right;
    };
    int func(struct treenode *p, struct treenode *q){
    if(p==NULL && q==NULL) return 1;
    if((!p && q) || (!q && p)) return 0;
    return (p->data==q->data) && func(p->left, q->left) && func( p->right, q >right);
    }
```

Initially the addresses of root node of two trees are passed into p and q respectively, the function-

A. Returns 1 iff the two trees are identical.

B. Returns 1 iff the two trees are mirror images of each other.

C. Returns 1 iff the two trees emerge from the same root node.

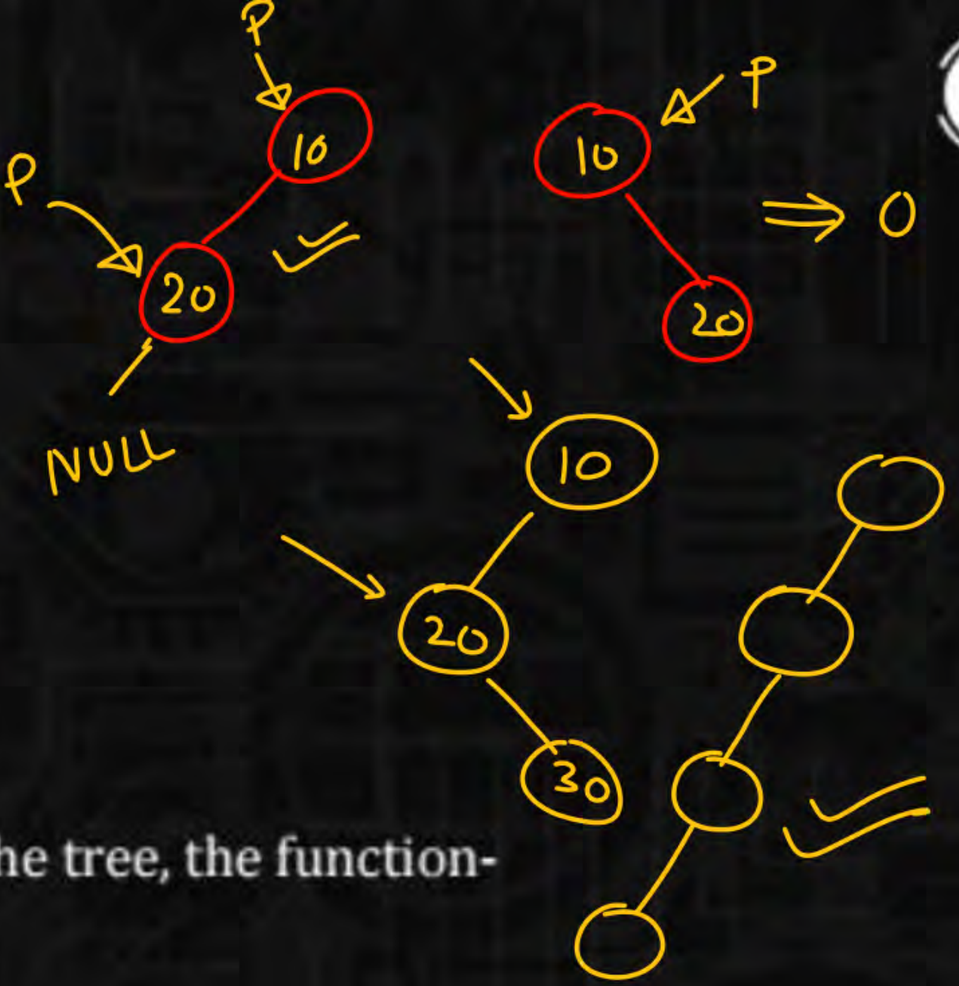
D. None of the above

Q.3

Consider the following function:

```
struct treenode{
        struct treenode *left;
        int data;
        struct treenode *right;
    };
  int func(struct treenode *p){
      if(p==NULL) return 1;
      else if(p->right!=NULL) return 0;
      return func(p->left);
      }
```

Initially p contains the root node address of the tree, the function-



A. Returns 1 if a binary tree is left-skewed.

B. Returns 1 if a binary tree is right-skewed.

C. Returns 1 if a binary tree is not right-skewed.

D. None of the above.

Q.4

Consider the following functions:

```
struct treenode{
        struct treenode *left;
        int data;
        struct treenode *right;
 };
int f1(struct treenode *t){
     if(t==NULL) return 1;
     else if(t->left!=NULL) return 0;
     return func(t->right);
}
int * f2 (struct treenode *t){
if(t==NULL) return 1;
else if(t->left==NULL && t->right==NULL)
return 1;
else if
((t → left → data < t->data) && (t → right → data > t->data))
return func(t->left) && func(t->right);
else
return 0;
}
 int f3(){return f2(t) && f1(t);}
```

Right Skewed

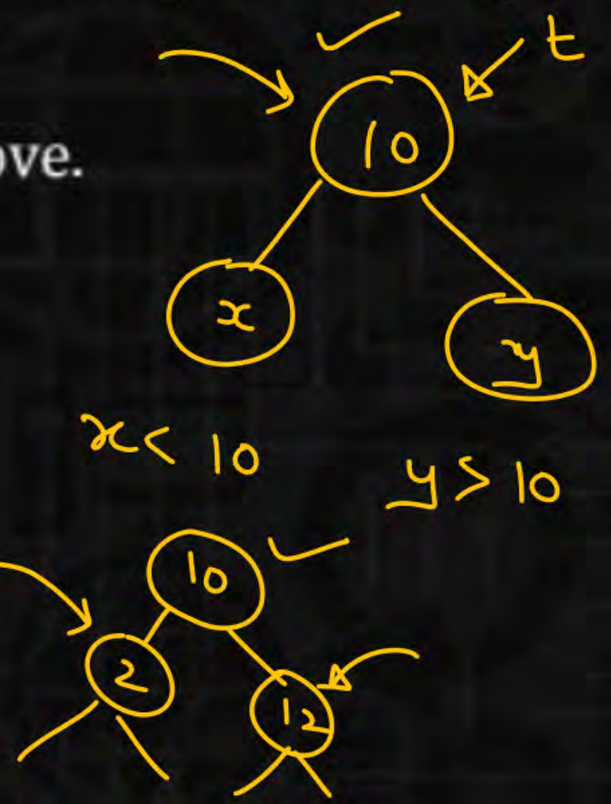
Assume, t is a pointer to the root node of a binary tree, the function f(3):

A. Returns 1 if the binary tree is a left skewed BST

B. Returns 1 if the binary tree is not a left skewed BST

C. Returns 1 if the binary tree is a right skewed BST

D. None of the above.



Q.4

Consider the following functions:

```
struct treenode{
        struct treenode *left;
        int data;
        struct treenode *right;
 };
int f1(struct treenode *t){
     if(t==NULL) return 1;
     else if(t->left!=NULL) return 0;
     return func(t->right);
}
int * f2 (struct treenode *t){
if(t==NULL) return 1;
else if(t->left==NULL && t->right==NULL)
return 1;
else if
((t → left → data < t->data) && (t → right → data > t->data))
return func(t->left) && func(t->right);
else
return 0;
}
 int f3(){return f2(t) && f1(t);}
```

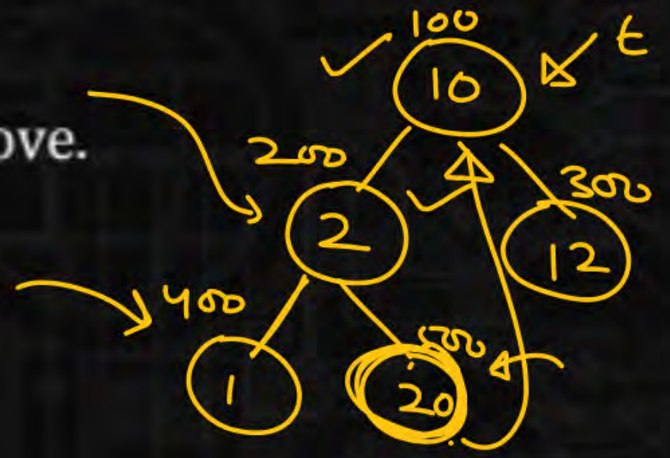
Assume, t is a pointer to the root node of a binary tree, the function f(3):

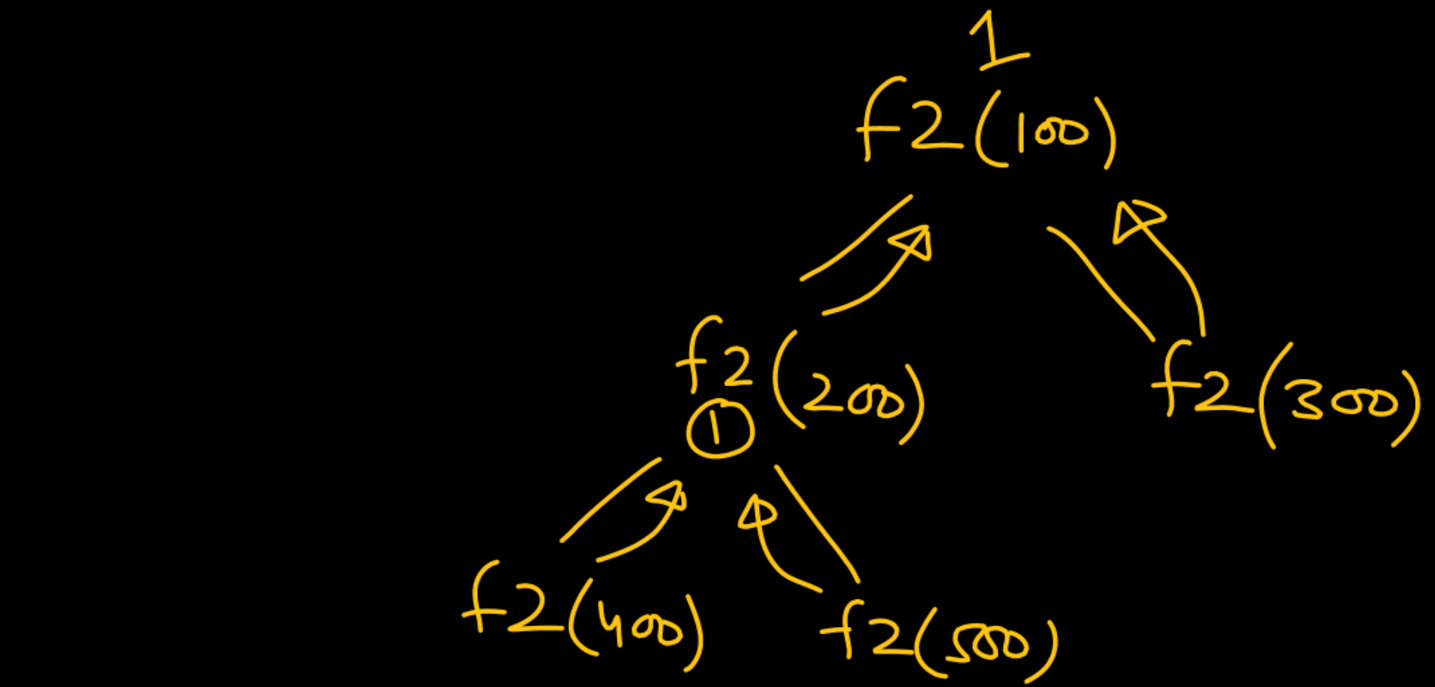
A. Returns 1 if the binary tree is a left skewed BST

B. Returns 1 if the binary tree is not a left skewed BST

C. Returns 1 if the binary tree is a right skewed BST

D. None of the above.

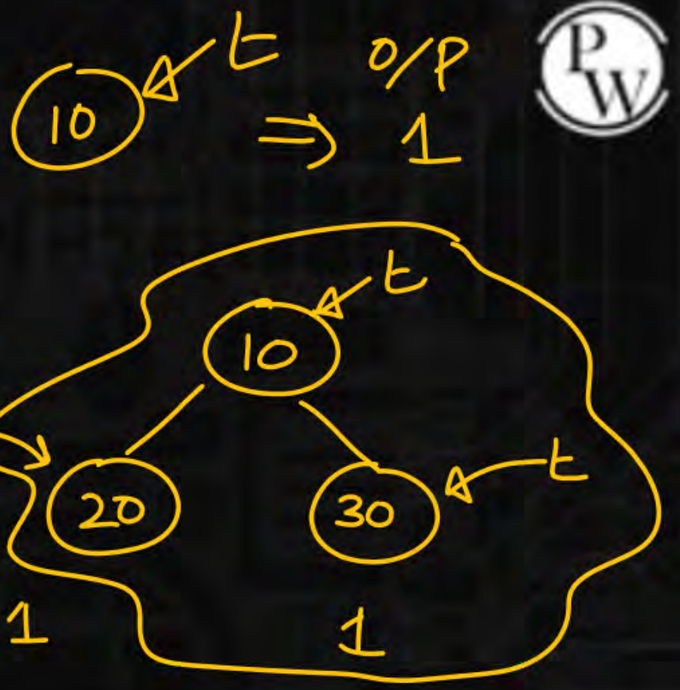



1
f2(100)
f2(200)
1
f2(300)
f2(400)
f2(500)

Q.5

Consider the following function:

```
struct treenode{
        struct treenode *left;
        int data;
        struct treenode *right;
 };
int func(struct treenode *t){
        if(t==NULL) return 0;
elseif(t->left==NULL && t->right==NULL) return 1;
        else
        return 1+func(t->left)+func(t->right);
}
```



Assume, t is a pointer to the root node of a binary tree, the function computes-

A. Number of leaf nodes in the binary tree

B. Number of internal nodes in the binary tree

C. Total number of nodes in the binary tree

D. None of the above

Q.6

The given tree is passed to the following function:

```
void func(struct treenode *t)
{
  if(t)
    {
1   printf("%d", t->data);
2   func(t->right);
3   printf("%d", t->data);
4   func(t->left);
    }
}
```

The output string is-

AEFFEABDDBCC

A. AEFFEBDDCCBA

B. AEFFEABDDBCC

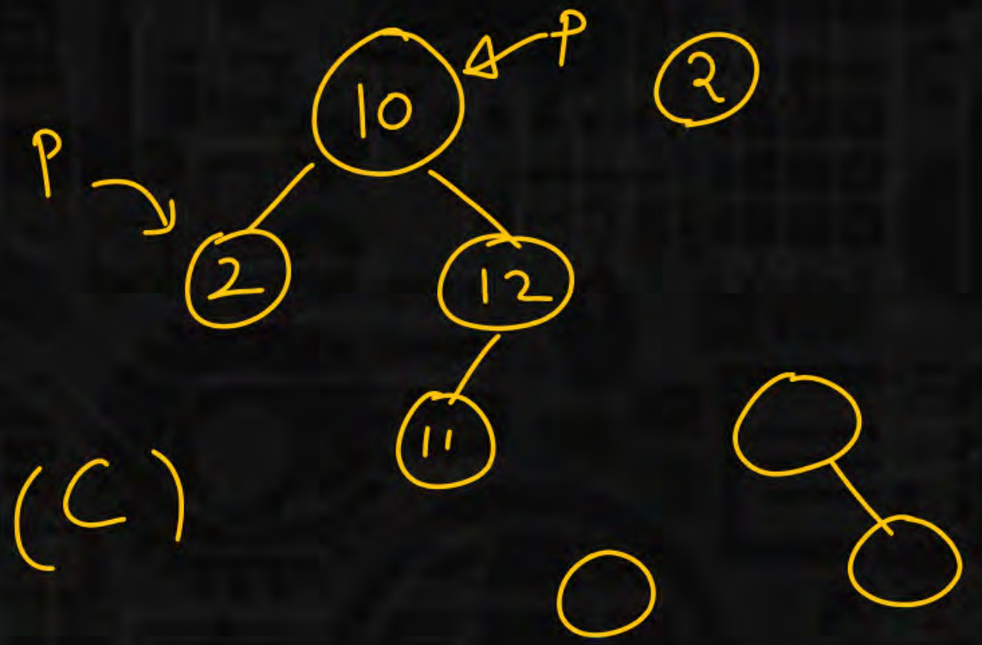
C. AEFFEBDDCCBA

D. None of the above

Q.7

Consider the following function:

```
struct treenode{
        struct treenode *left;
        int data;
        struct treenode *right;
  };
  void func(struct treenode *p){
  while(p->left!=NULL) p=p->left;
  printf("%d", p->data);
}
```



If the address of the root node of the BST is passed to p, the above function prints-
(Assume, the tree contains at least one node)

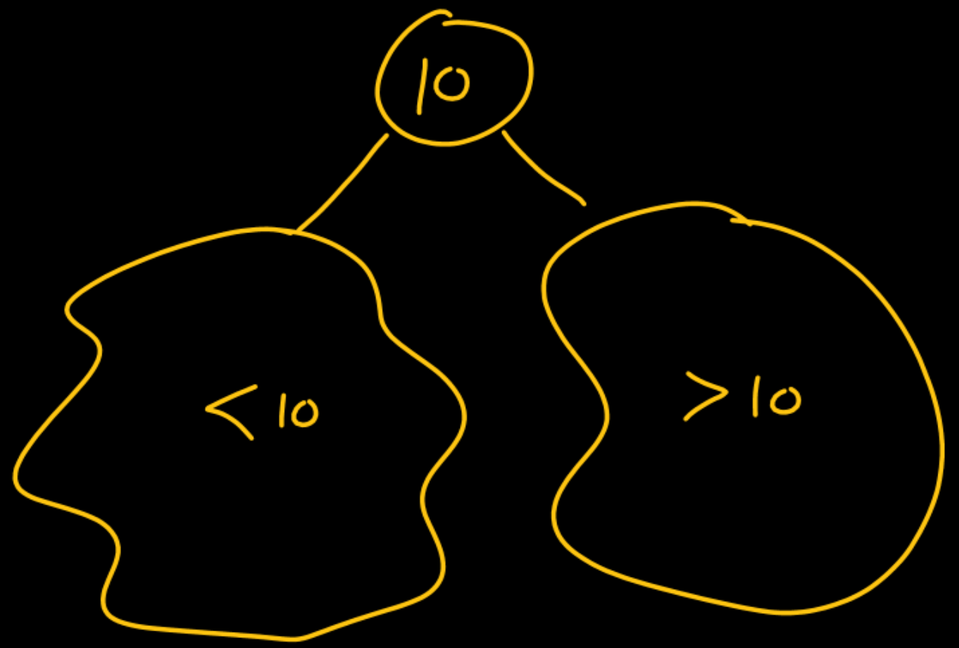
A. The maximum element in the BST

B. The ancestor of two leftmost leaf nodes

C. The minimum element in BST

D. None of the above


10
5
greater than 5


10
< 10
> 10

Q.8

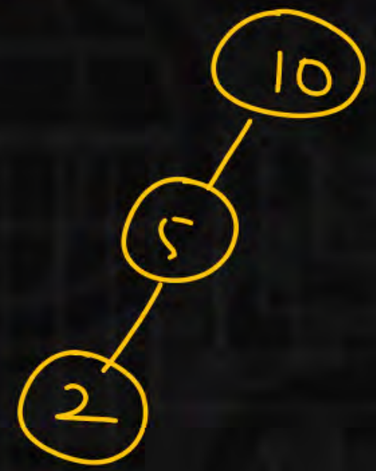
Consider the following two statements:

P: The minimum number of nodes in a complete binary tree of height h is 2^{h+1}.

$2^h \leq n \leq 2^{h+1}$

Q: A binary search tree is always a complete binary tree.

Which of the statement(s) is/are CORRECT?



(D)

A. P only

B. Q only

C. Both P and Q

D. Neither P nor Q

Q.1 The minimum number of nodes in AVL tree of height 6 is 20.
(Assume that the height of the root node is 1)

[NAT]

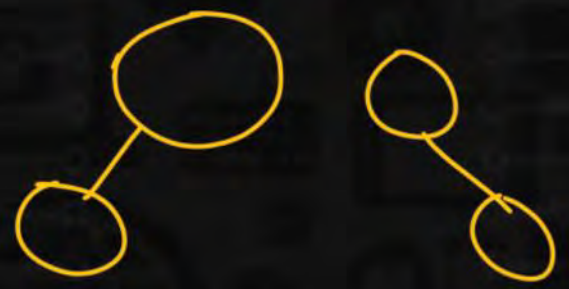
$n_{max} = 2^{h+1} - 1$

$n(h) = 1 + n(h-1) + n(h-2)$

$n(1) = 1$

$n(2) = 2$

$n(h)$: min. no. of nodes in an AVL tree of height h.



$n(1) = 1$
$n(2) = 2$
$n(3) = 1+2+1 = 4$
$n(4) = 1+4+2 = 7$
$n(5) = 1+7+4 = 12$
$n(6) = 1+12+7 = 20$

Q.2

Consider the following statements:

P: An AVL tree is a height-balanced complete binary tree.

Q: A heap is necessarily a complete binary tree.

Which of the following statement(s) is/are CORRECT?

[MCQ]

A. P only

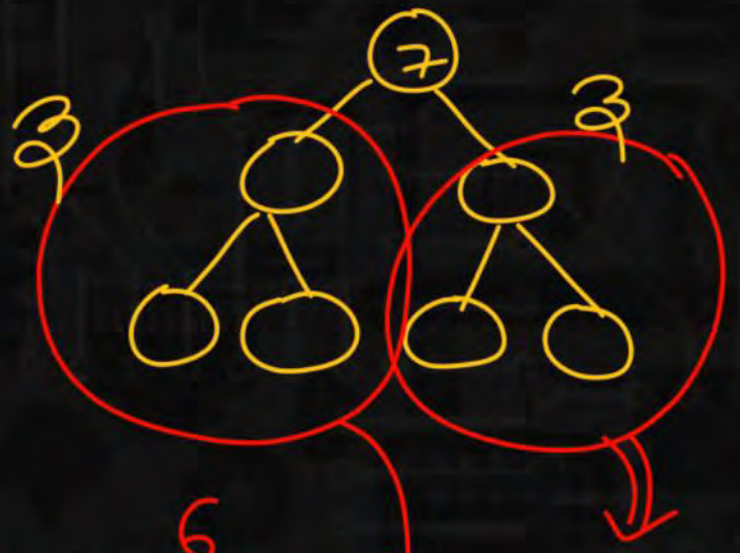
B. Q only

C. Both P and Q

D. Neither P nor Q

Q.3 The total number of ways in which a max-heap can be constructed with the keys-7, 6, 1, 4, 5, 2, 3 is 80.

[NAT]



$$F(7) = 1 \times {}^{6}C_3 \times F(3) \times F(3)$$

$$= \frac{6!}{3!3!} \quad 2 \times 2$$

$$= \frac{6 \times 5 \times 4 \times 3!}{3! \times 3!} \times 4 = 80$$

No. of max. heap with 7 keys $= {}^{6}C_3 \times x \times y$

x (No. of max. heap with selected 3 keys)

y (No. of max heap possible with rem. 3 keys)

$$F(n) = 1 \times {}^{n-1}C_k \times F(k) \times F(n-1-k)$$

10, 20, 30

$F(1) = 1$

$F(2) = 1$

$F(3) = 2$

$1 \times {}^{2}C_1 \times F(1) \times F(1)$

$= 2 \times 1 \times 1$

$= 2$

Q.4

Consider the following statements:

P: If the root node of a BST is deleted, it can be replaced by inorder predecessor. Correct

Q: If the root node of a BST is deleted, it can be replaced by preorder successor. ✗

Which of the following is/are CORRECT

Root → leaf

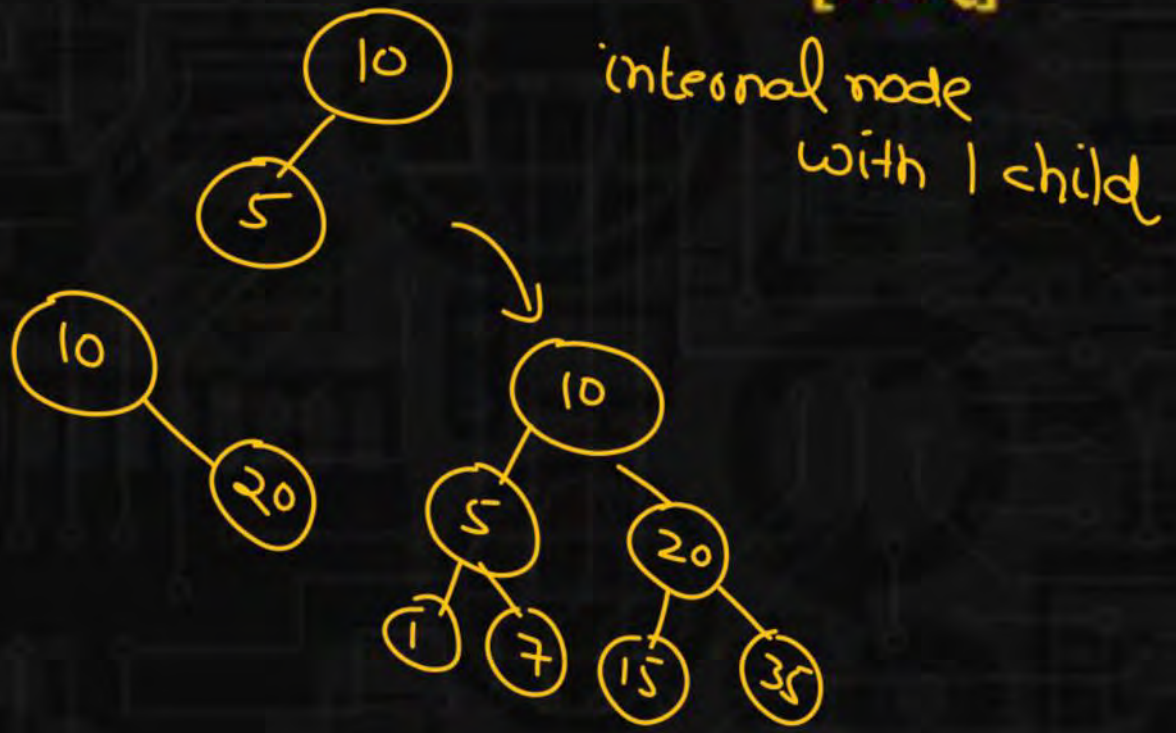
[MCQ]

A. P only

B. Q only

C. Both P and Q

D. Neither P nor Q



internal node with 1 child

Q.5

Consider the following operations in a BST-
INSERT(23), INSERT(17), INSERT(25), INSERT(4), INSERT(21), INSERT(1), INSERT(7), DELETE(17), DELETE(23).
The post-order traversal of the resultant BST is-

[MSQ]

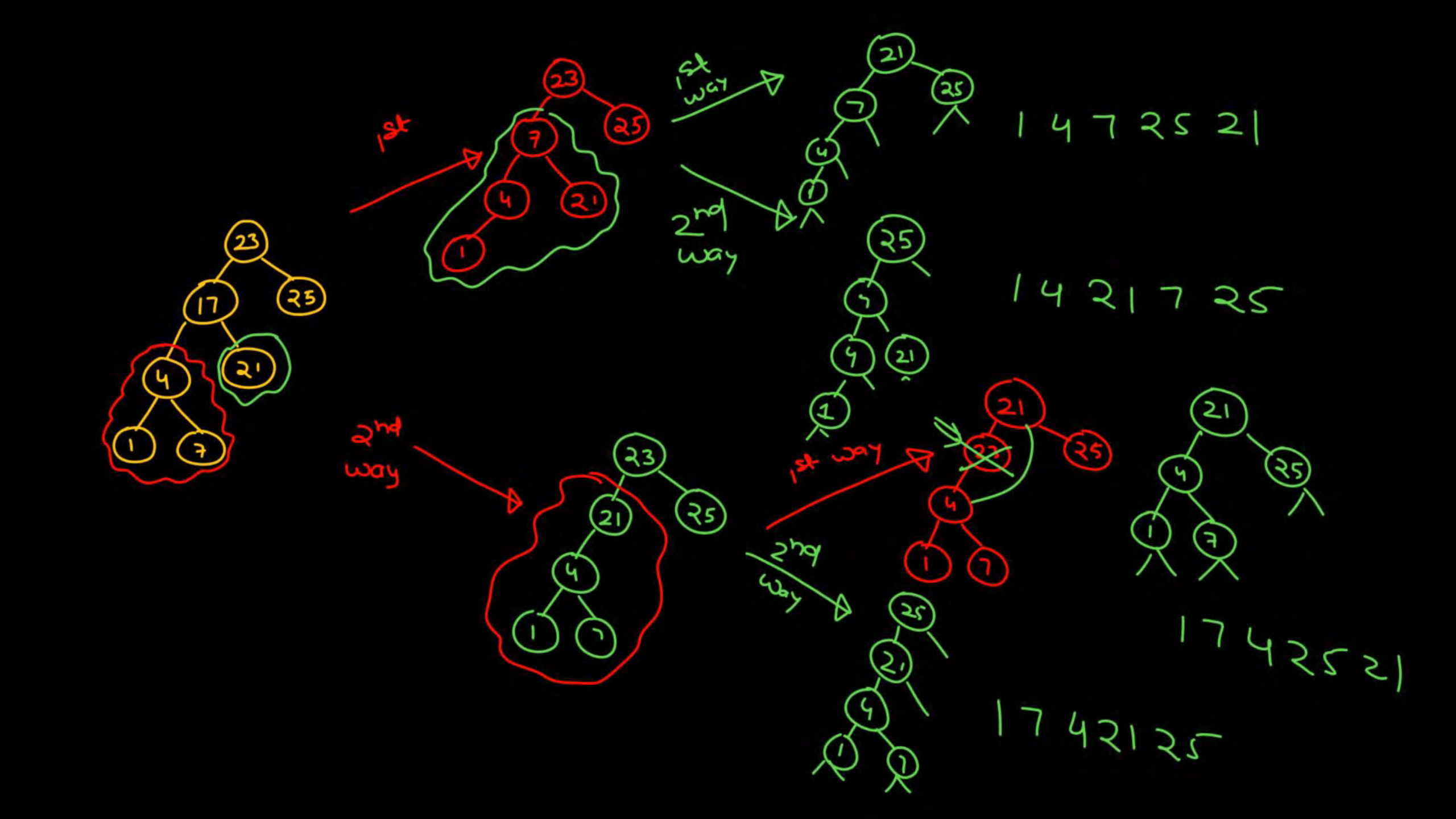
A, B, C

A. 1, 7, 4, 21, 25

B. 1, 4, 7, 25, 21

C. 1, 4, 21, 7, 25

D. None of the above


1st
2nd
1st way
2nd way
1 4 7 25 21
1 4 21 7 25
1 7 4 25 21
1 7 4 21 25

Q.6

Which of the following sequence(s) of array form a heap?

[MSQ]

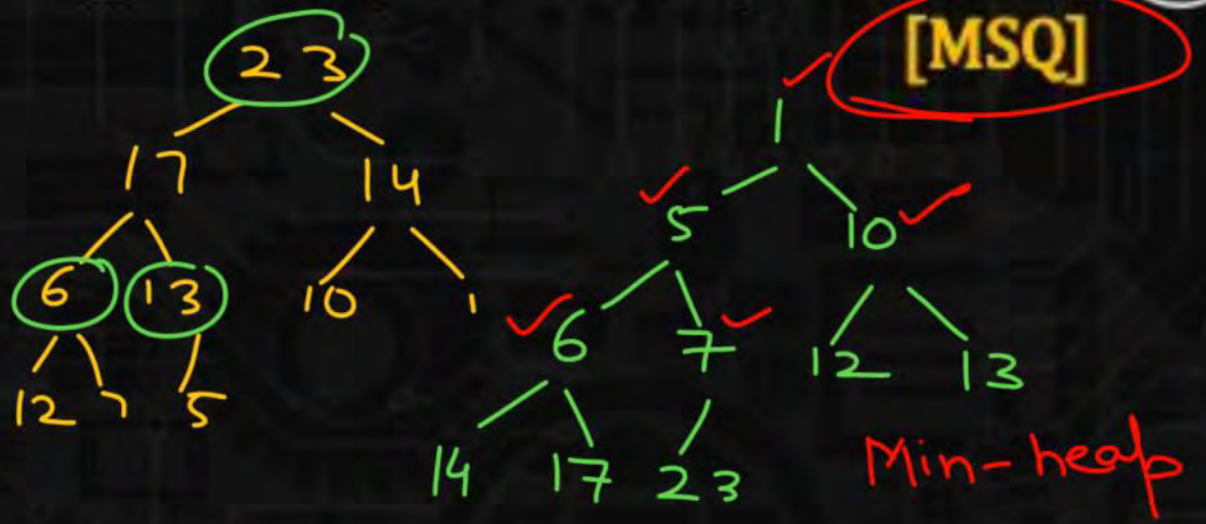
B, C

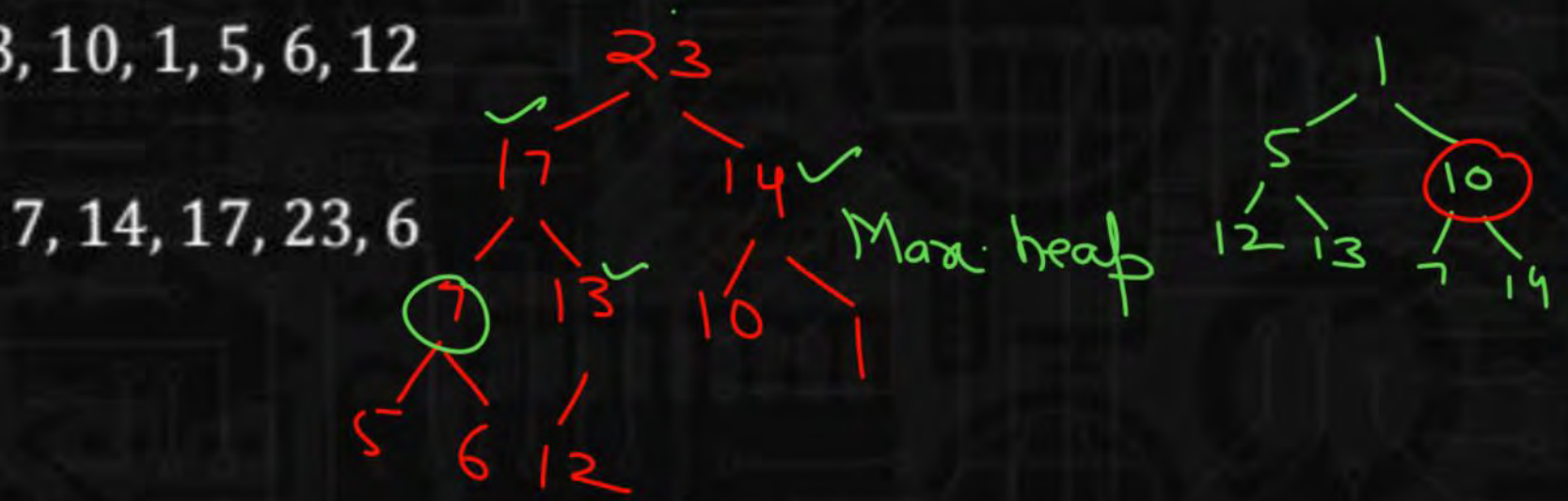
A. 23, 17, 14, 6, 13, 10, 1, 12, 7, 5

B. 1, 5, 10, 6, 7, 12, 13, 14, 17, 23

C. 23, 17, 14, 7, 13, 10, 1, 5, 6, 12

D. 1, 5, 10, 12, 13, 7, 14, 17, 23, 6





Q.7

Consider the following statements:

P: The accepted balanced factor in an AVL tree are –1, 0 and +1.

Q: The height of an AVL tree with n nodes is given as $\lceil \log_2 n \rceil$.

The number of INCORRECT statements is ____.

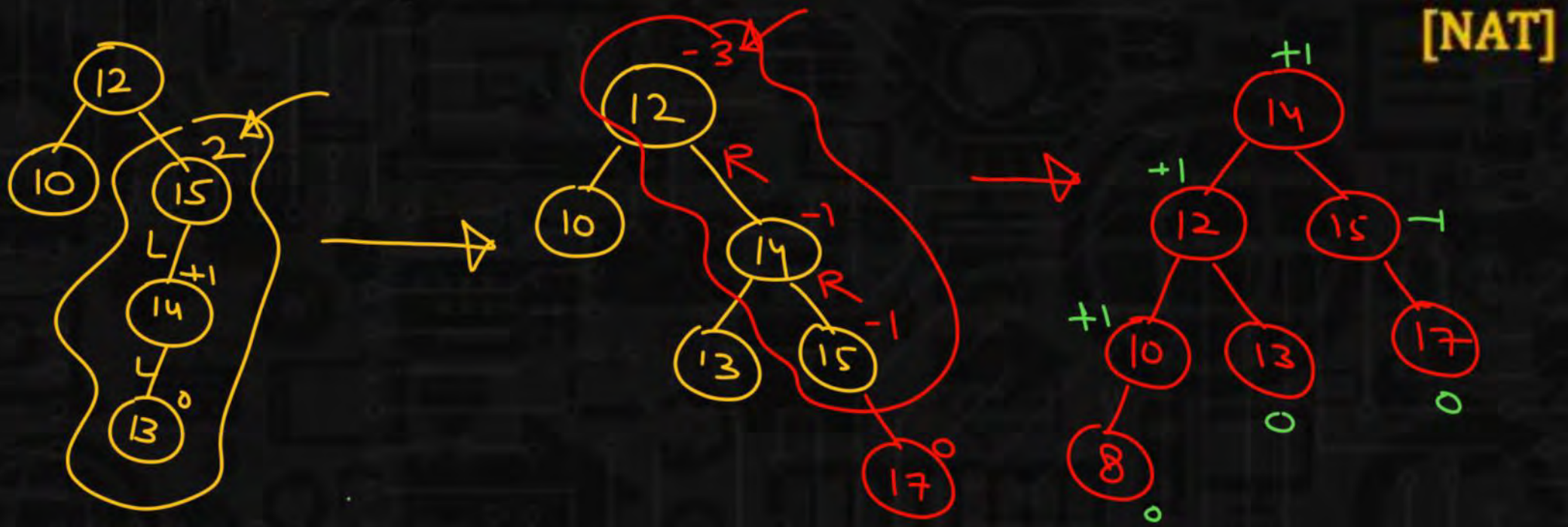
[NAT]

Q.8

Construct an AVL tree with the following keys:
12, 10, 15, 14, 13, 17, 8

The immediate left child key value of the root node of the AVL tree is 12.

[NAT]



Q.1 The maximum number of comparisons to find the maximum element in a min heap of 1024 elements is 511 [NAT]

leaf nodes

n elements $\quad \left\lceil \frac{n}{2} \right\rceil$

$\left\lceil \frac{3}{2} \right\rceil = \lceil 1.5 \rceil = 2$

$\left\lceil \frac{4}{2} \right\rceil = 2$

$n=6 \quad \left\lceil \frac{6}{2} \right\rceil = 3$

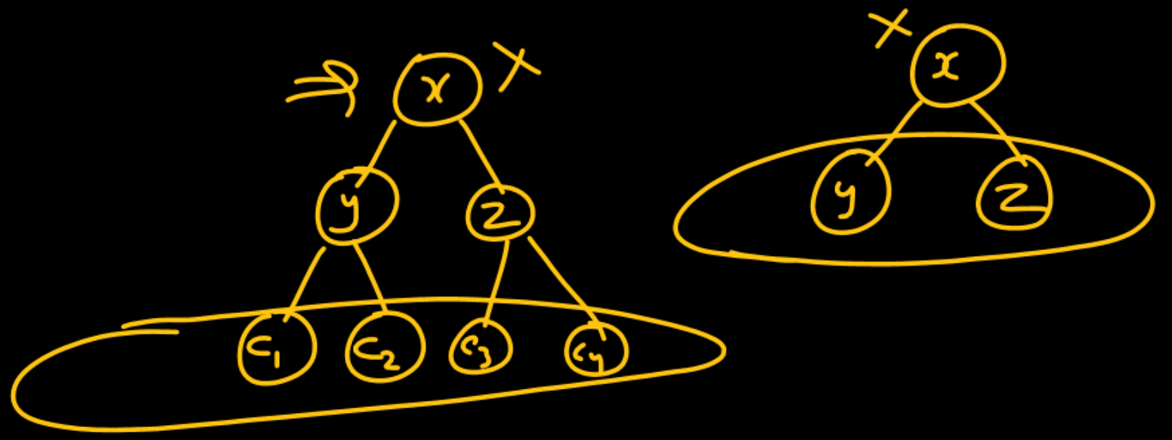
$\left\lceil \frac{7}{2} \right\rceil = \lceil 3.5 \rceil = 4$

1024

$$\#\text{leaf} = \left\lceil \frac{1024}{2} \right\rceil$$

$$= 512$$

element



$c_1, c_2 > y$

$c_3\ c_4 > z$

$y, z \Rightarrow$ can not be largest

9

512 − 1

= 511

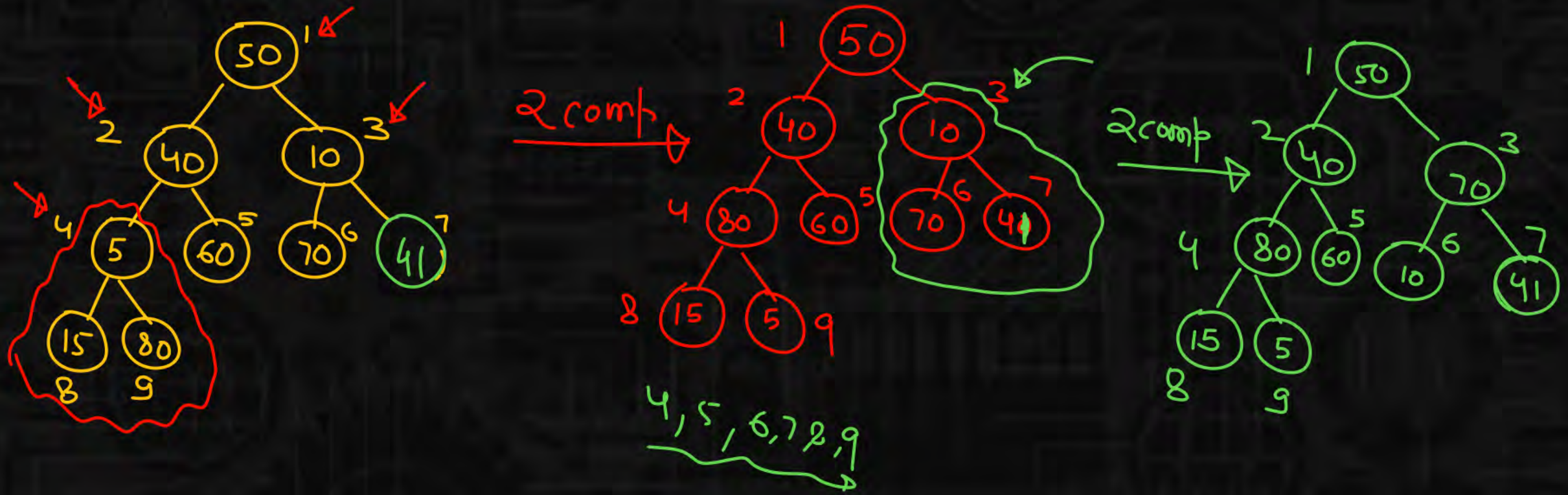
Q.2

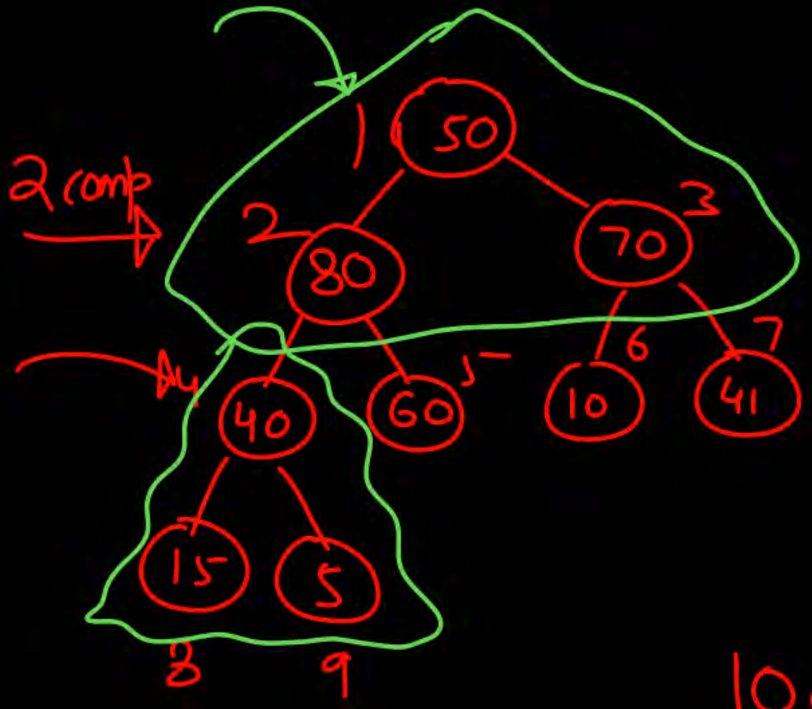
[NAT]

Consider the array given below:

50	40	10	5	60	70	40	15	80

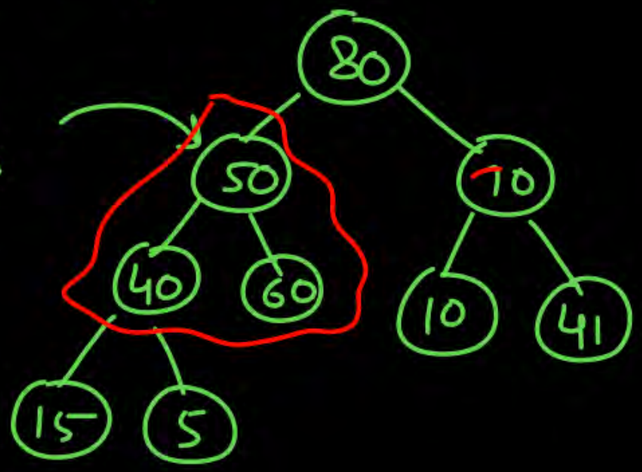
The minimum number of comparisons required to convert the above array into max heap is 10





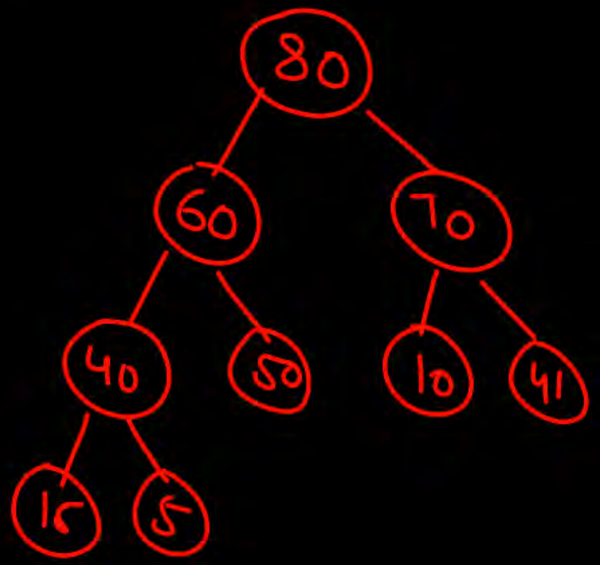
element comp.

2 comp



10 comp.

2 comp



Q.3 Consider the array given below: **[NAT]**

50	40	10	5	60	70	4	15		80

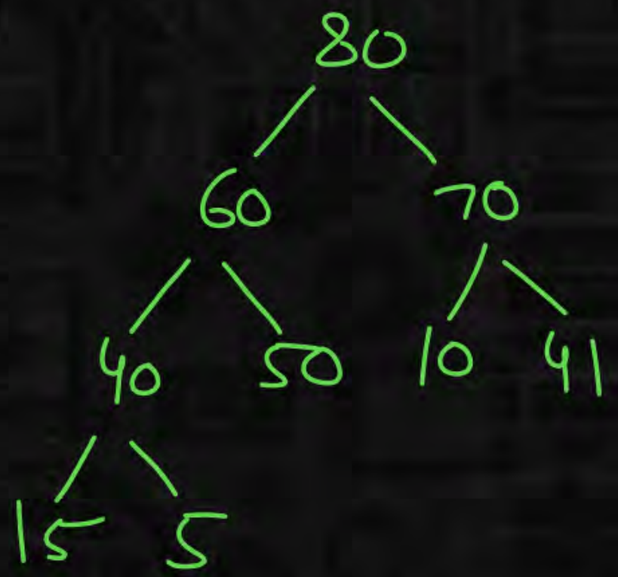
The minimum number of swap operations required to convert the above array into max-heap is _____.

Q.3 [NAT]

Consider the array given below:

50	40	10	5	60	70	41	15		80

The minimum number of swap operations required to convert the above array into max-heap is 5.



Q.4

[MCQ]

Consider the array given below:

50	40	10	5	60	70	4(	15	80

The resultant max-heap using bottom-up approach of build heap is-

A. 80, 60, 70, 40, 50, 10, 4(, 15, 5

B. 80, 70, 60, 50, 40, 10, 4(, 5, 15

C. 80, 70, 60, 50, 40, 4(, 15, 10, 5

D. None of the above

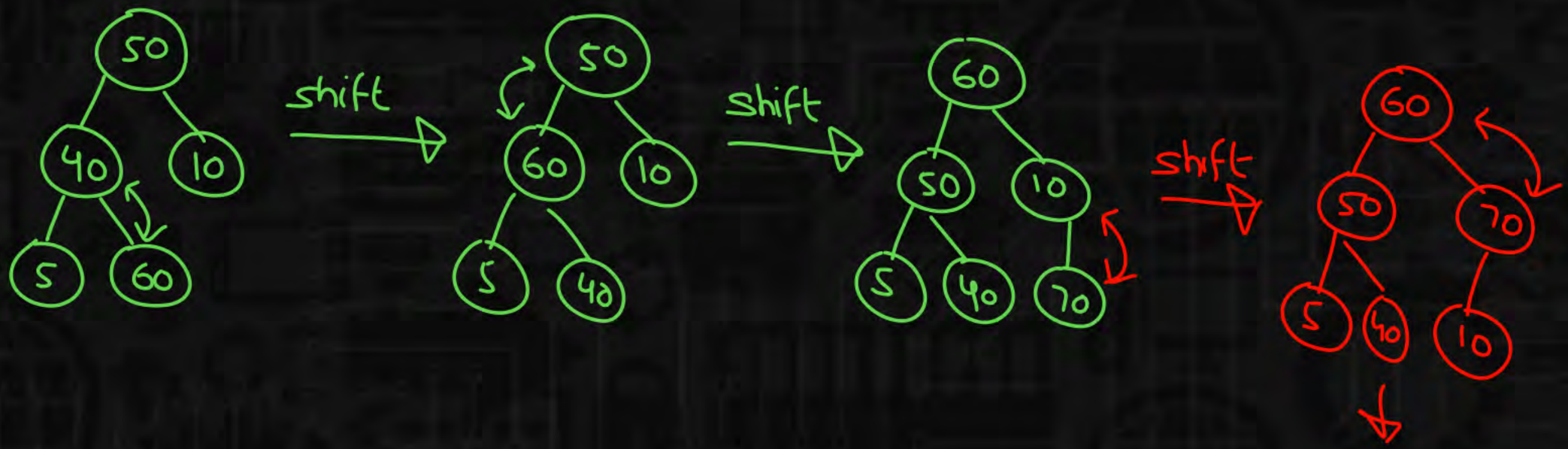
A

Q.5 [NAT]

Consider a sequence of elements are inserted into a max-heap one after another as-

50, 40, 10, 5, 60, 70, 4(), 15, 80

The number of shift operations required in building the heap one element at a time is _____.



Q.5 [NAT]

Consider a sequence of elements are inserted into a max-heap one after another as-

50, 40, 10, 5, 60, 70, 41, 15, 80

The number of shift operations required in building the heap one element at a time is 8.

Q.6

[MCQ]

Consider a sequence of elements are inserted into a max-heap one after another as-

50, 40, 10, 5, 60, 70, 4(, 15, 80

The resultant max-heap using bottom-up approach of build heap is-

A. 80, 60, 70, 40, 50, 10, 4(, 15, 5

B. 80, 70, 60, 50, 40, 10, 4(, 5, 15

C. 80, 70, 60, 50, 40, 4(, 15, 10, 5

D. None of the above

(B)

Q.7 [MCQ]

Consider the following two statements:

P: The number of comparisons required to find the minimum element in a min heap of n elements is n – 1.

Incorrect

return A[1]

Q: Only one comparison is required to find the minimum element in a max heap of n elements.

Incorrect

$\lceil \frac{n}{2} \rceil - 1$

Which of the following is/are CORRECT?

A. P only

B. Q only

C. Both P and Q

D. Neither P nor Q

Q.1 Which of the following is/are correct inorder traversal sequence(s) of binary search tree(s)? [MCQ]

inc. order of keys

I. 3, 5, 7, 8, 15, 19, 25

II. 5, 8, 9, 12, 10, 15, 25

III. 2, 7, 10, 8, 14, 16, 20

IV. 4, 6, 7, 9 18, 20, 25

(A)

A. I and IV

B. II and III

C. II and IV

D. II only

Q.2 What is the worst-case time complexity of inserting n^2 elements into an AVL-tree with n elements initially? **[MCQ]**

A. $O(n^2)$

B. $O(n^2 \log n)$

C. $O(n^4)$

D. $O(n^3)$

n^2

1st elem $\longrightarrow$ $\log n$

2nd elem $\longrightarrow$ $\log(n+1)$

3rd elem $\longrightarrow$ $\log(n+2)$

$\vdots$

n^2 elem $\longrightarrow$ $\log(n+n^2-1)$

n elements

$(n+1)$

$(n+2)$

$$O(\log n + \log(n+1) + \log(n+2) + \cdots \log(n+n^2-1))$$

$$= O(\log[(n)(n+1)(n+2)(n+3)\cdots(n+n^2-1)])$$

$$= O(\log(n)^{n^2})$$

$$= O(n^2 \log n)$$

Q.3

Suppose the numbers 7, 5, 1, 8, 3, 6, 0, 9, 4, 2 are inserted in that order into an initially empty binary search tree. The binary search tree uses the usual ordering on natural numbers. What is the pre-order traversal sequence of the resultant tree?

[MCQ]

A. 7 5 1 0 3 2 4 6 8 9

B. 0 2 4 3 1 6 5 9 8 7

C. 0 1 2 3 4 5 6 7 8 9

D. 9 8 6 4 2 3 0 1 5 7

7 5 1 0 3 2 4 6 8 9

Q.4

[MCQ]

Consider the following statements.

S_1: The sequence of procedure calls corresponds to a preorder traversal of the activation tree.

True

S_2: The sequence of procedure returns corresponds to a postorder traversal of the activation tree.

True

Which one of the following options is correct?

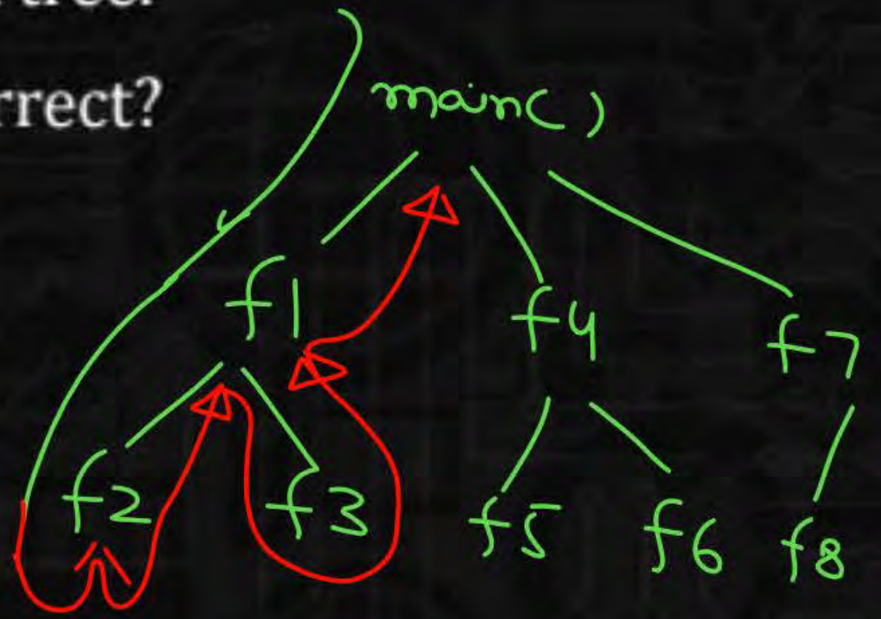
A. S_1 only

B. S_2 only

C. Both S_1 and S_2

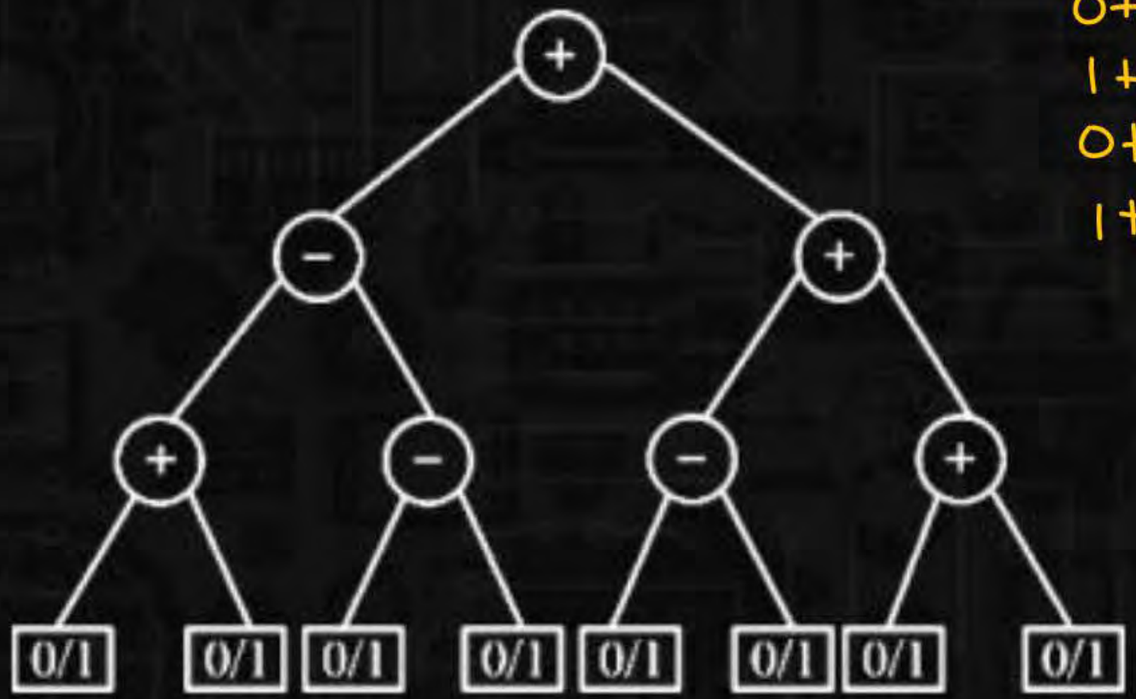
D. Neither S_1 nor S_2

```
Void f4()
{
f5();
f6();
}

main(){
f1();
f4();
f7();
}

Void f7(){
f8();
}

Void f1()
{
f2();
f3();
}
```



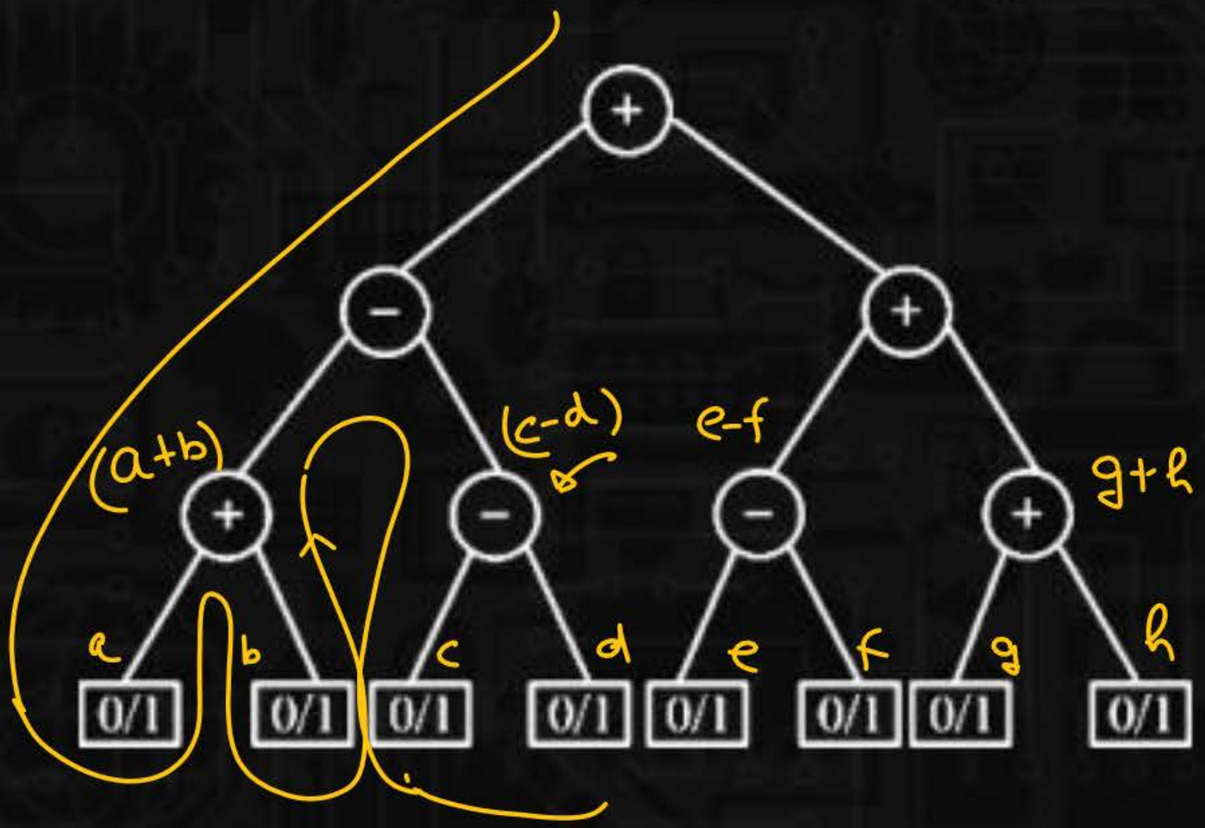
Q.5

Consider the expression tree shown. Each leaf represents a numerical value, which can either be 0 or 1. Over all possible choices of the values at the leaves, the maximum possible value of the expression represented by the tree is ___. **[NAT]**



	Subtraction		Max	Min
0+0=0	0−0=0	a+b	2	0
1+0=1	1−0=1	a−b	1	−1
0+1=1	0−1=−1			
1+1=2	1−1=0			

Q.5 Consider the expression tree shown. Each leaf represents a numerical value, which can either be 0 or 1. Over all possible choices of the values at the leaves, the maximum possible value of the expression represented by the tree is 6. **[NAT]**



	Max	Min
a+b	2	0
a-b	1	-1

Max

$$[(a+b)-(c-d)]+\overbrace{[(e-f)+(g+h)]}^{\text{Max}}$$

$$[2-(c-d)]+[1+(2)]$$

$$[2-(c-d)]+3$$

Min

$$[2-(-1)]+3 = 3+3 = 6$$

Q.6

A Binary Search Tree (BST) stores values in the range 37 to 573. Consider the following sequence of keys. [MCQ]

I. 81, 537, 102, 439, 285, 376, 305

II. 52, 97, 121, 195, 242, 381, 472

III. 142, 248, 520, 386, 345, 270, 307

IV. 550, 149, 507, 395, 463, 402, 270

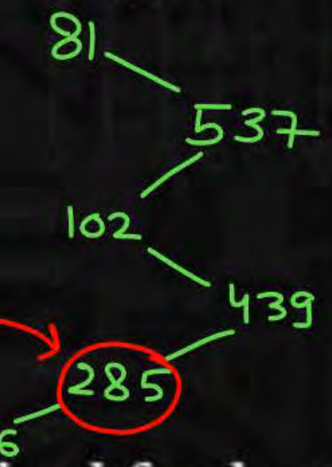
Suppose the BST has been unsuccessfully searched for key 273. Which all of the above sequences list nodes in the order in which we could have encountered them in the search?

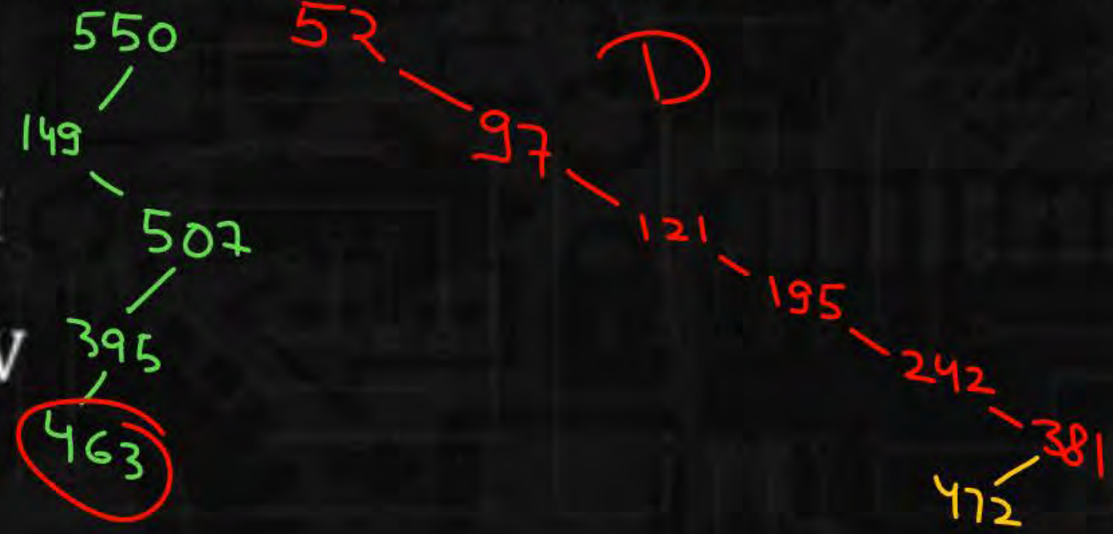
A. I and III

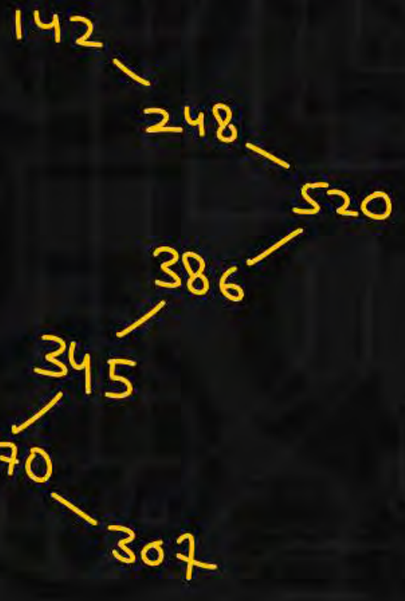
B. II and III

C. III and IV

D. III only



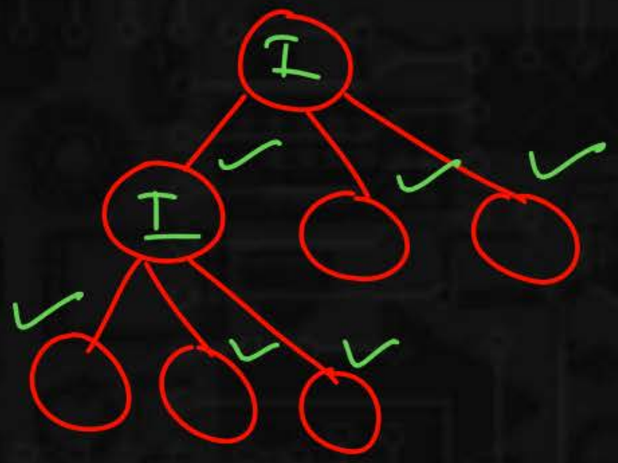




Q.7

A complete n-ary tree is a tree in which each node has n children or no children. Let I be the number of internal nodes and L be the number of leaves in a complete n-ary tree. If L = 41, and I = 10, what is the value of n? _____. **[NAT]**

Each internal node ⇒ n child



2 × 3 + 1 (Root)

No. of internal node → #child (for Each internal node)

Total = I × n + 1

51 = 10 × n + 1

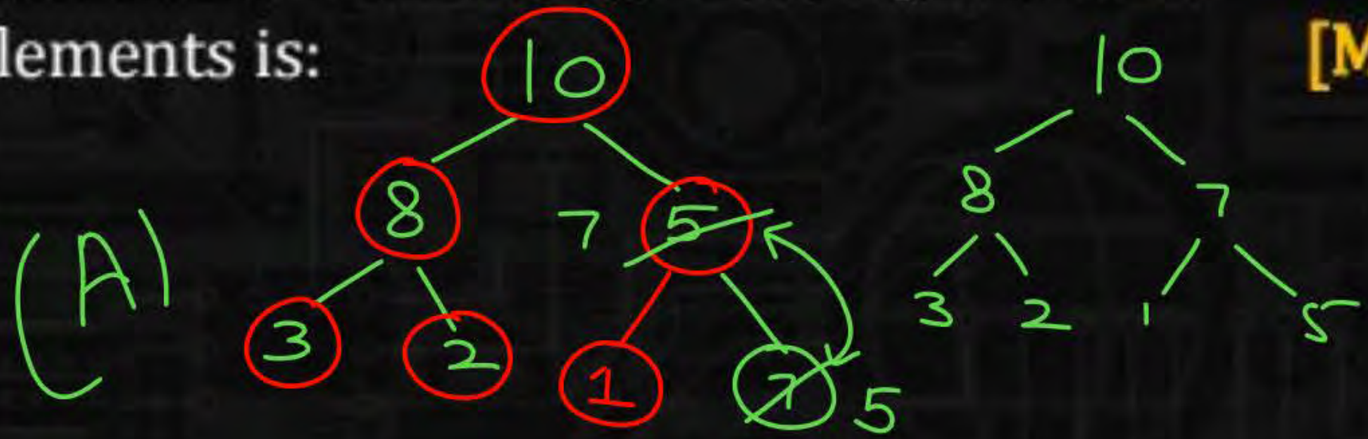
50 = 10 × n

n = 5

Q.8

A Priority-Queue is implemented as a Max-Heap. Initially, it has 5 elements. The level-order traversal of the heap is given below: 10, 8, 5, 3, 2. Two new elements '1' and '7' are inserted in the heap in that order. The level-order traversal of the heap after the insertion of the elements is:

[MCQ]



A. 10, 8, 7, 3, 2, 1, 5

B. 10, 8, 7, 2, 3, 1, 5

C. 10, 8, 7, 3, 2, 5, 1

D. None of the above


THANK YOU GW
SOLDIERS !