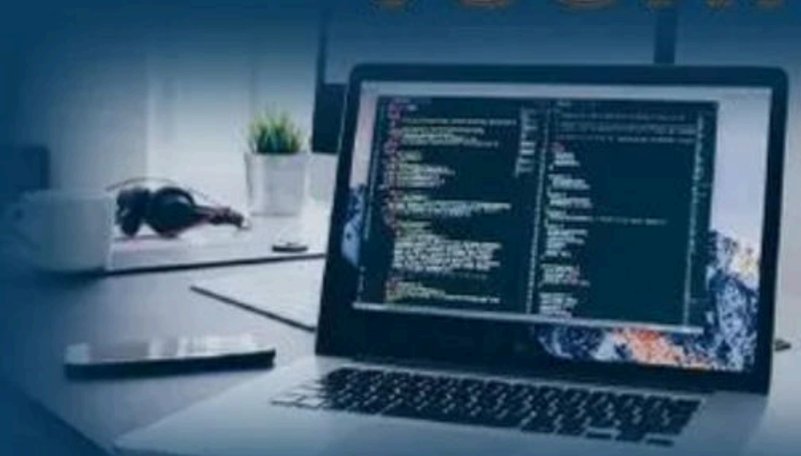
unacademy

Problem solving - Part III

Course on Data Structure

Pankaj Sharma • Lesson 37 • Aug 23, 2024

Computer Science And Information Technology



Lecture Number : 37

Data Structure

By- Pankaj sir

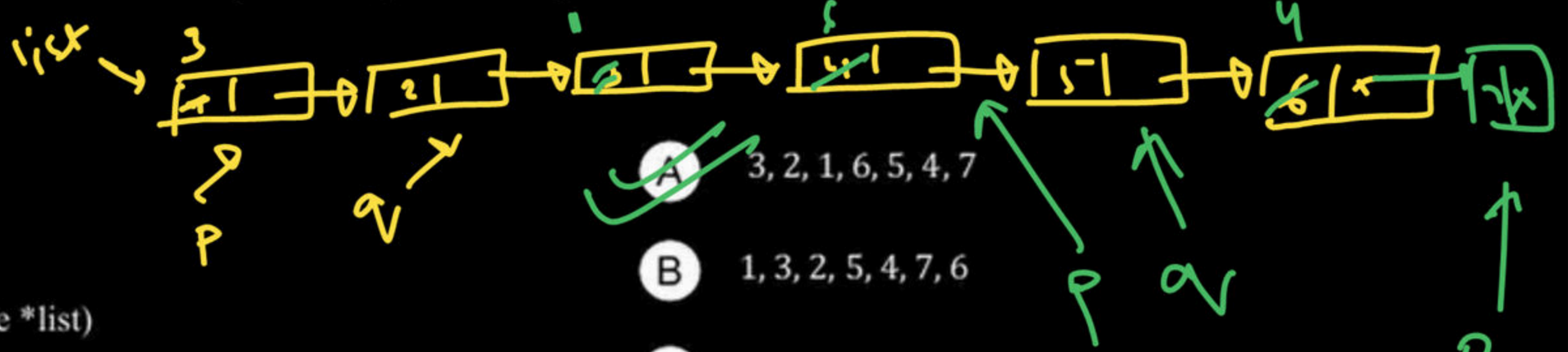
QUESTION

Q. The following C function takes a singly linked list of integers as a parameter and rearranges the elements of the list. The function is called with the list containing the integers in the given order. What will be the contents of the list after the function completes execution?

```
struct node
{
  int value;
  struct node *next;
};
void rearrange(struct node *list)
{
  struct node *p, *q;
  int temp;
  if (!list || !list -> next) return;
    p = list; q = list -> next;
  while(q && q->next) {
    temp = p -> value;
    p->value =  q->next -> value;
    q->next->value = temp;
    p = q -> next ->next;
    q = p ? p ->next : 0;
  } }
```

Initial List: 1, 2, 3, 4, 5, 6, 7

What will be the final contents of the list?



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

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

(C) 2, 1, 4, 3, 6, 5, 7

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

q = NULL

3 2 1 6 5 4 7

QUESTION

Q What will be the contents of the list after the function completes execution?

```
struct node {
    int value;
    struct node *next;
};
void rearrange(struct node *list)
{
    if (!list || !list->next || !list->next->next) return;
    struct node *p1 = list, *p2 = list->next, *p3 = list->next->next;
    while (p1 && p2 && p3)
    {
        int temp = p1->value;
        p1->value = p3->value;
        p3->value = temp;
       p1 = p3->next;
        p2 = p1 ? p1->next : 0;
        p3 = p2 ? p2->next : 0;
    }
}
```

Initial List: 1, 2, 3, 4, 5, 6, 7, 8, 9

What will be the final contents of the list?

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

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

C 1, 3, 2, 5, 4, 7, 6, 9, 8

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

Same as Previous

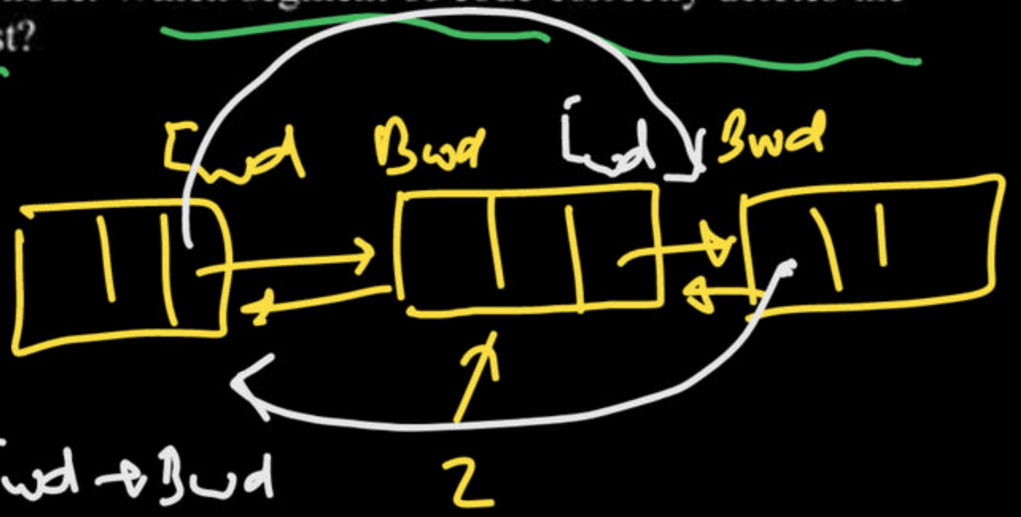
QUESTION

Q. Consider a doubly linked list defined as follows:

```
struct Node
{
   int Value;
   struct Node *Fwd;
   struct Node *Bwd;
};
```

In this list, Fwd points to the next node and Bwd points to the previous node. Which segment of code correctly deletes the node pointed to by Z, given that Z is neither the head nor the tail of the list?

Not first / last node
⇒ Any 'intermediate node



A. `Z->Bwd->Fwd = Z->Fwd; Z->Fwd->Bwd = Z->Bwd;`

B. `Z->Bwd->Fwd = Z; Z->Fwd->Bwd = Z;`

C. `Z->Bwd->Fwd = Z->Fwd; Z->Fwd->Bwd = Z->Fwd;`

D. ~~`Z->Bwd->Fwd = Z->Fwd; Z->Fwd->Bwd = Z->Bwd;`~~

Z → Fwd → Bwd = Z → Bwd

Z → Bwd → Fwd = Z → Fwd

None of these.

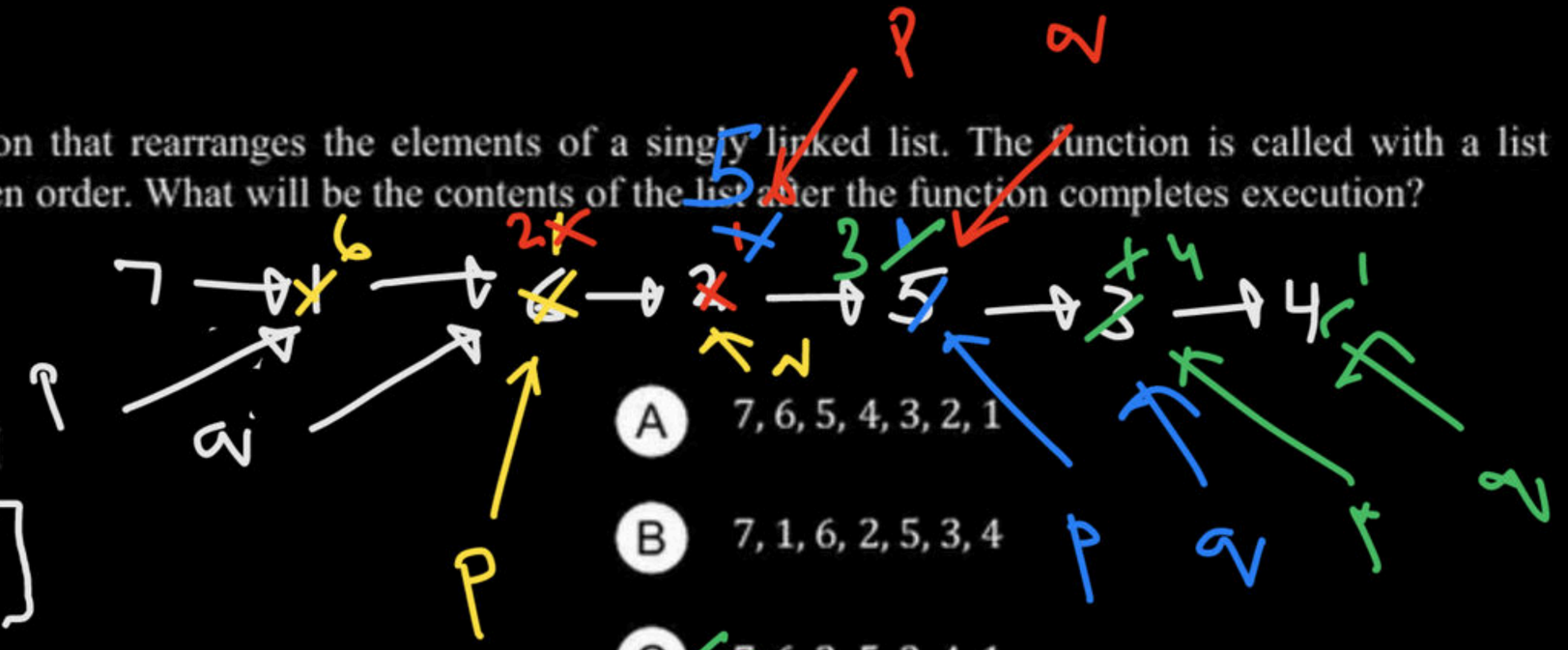
QUESTION

Q. Consider the following C function that rearranges the elements of a singly linked list. The function is called with a list containing the integers in the given order. What will be the contents of the list after the function completes execution?

```
struct node {
   int value;
   struct node *next;
};
void rearrange(struct node *list) {
   struct node *p, *q;
   int temp;
   if (!list || !list -> next) return;
   p = list; q = list -> next;
   while(q) {
      if (p->value < q->value) {
         temp = p -> value;
         p->value = q -> value;
         q->value = temp;
      }
      p = q;
      q = q->next;
      }
}
```

Initial List: 7, 1, 6, 2, 5, 3, 4

What will be the final contents of the list?



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

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

(C) 7, 6, 2, 5, 3, 4, 1

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

(E) None

7 6 2 5 3 4 1

QUESTION

Q. The following C function takes a singly linked list as input and prints its elements iteratively. Some part of the code is left blank.

```
typedef struct node
{
    int value;
    struct node *next;
}
Node;
void printlist(Node *head)
{
    Node* current = head;
    // ---- Blank ----
}
```

Choose the correct alternative to replace the blank line:

A
```
while (current != NULL) {
printf("%d", current->value);
current = current->next;
}
```

B
```
while (current != NULL) {
printf("%d", current->next->value);
current = current->next;
}
```

C
```
while (current != NULL) {
current = current->next;
printf("%d", current->value);
}
```

D None of these

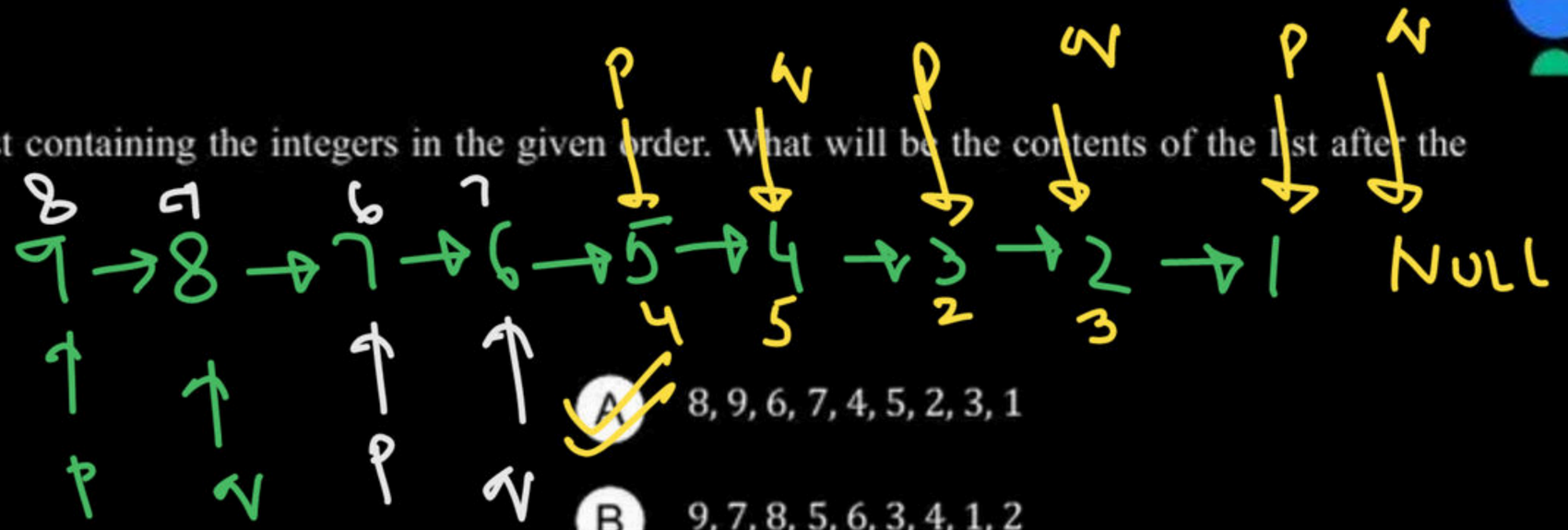
QUESTION

Q. The function is called with a list containing the integers in the given order. What will be the contents of the list after the function completes execution?

```
struct node {
   int value;
   struct node *next;
};
void rearrange(struct node *list) {
   struct node *p, *q;
   int temp;
   if (!list || !list -> next) return;
   p = list; q = list -> next;
  while(q) {
      temp = p -> value;
      p->value =  q -> value;
      q->value = temp;
      p = q ->next;
      q = p ? p ->next : 0;
   }
}
```

Initial List: 9, 8, 7, 6, 5, 4, 3, 2, 1

What will be the final contents of the list?



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

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

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

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

unacademy

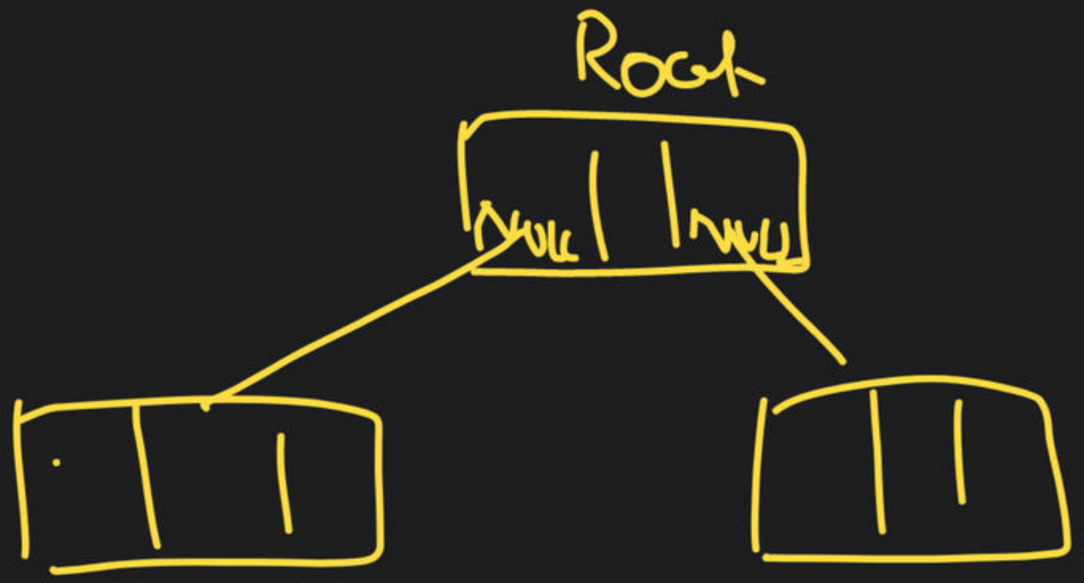
Q A binary tree has 1000 leaves. Then the no. of nodes having 2 childs is 999

$$n_0 = n_2 + 1$$

unacademy

Q/ A linked list rep. is used to store a binary tree with 500 nodes. The no. of NULL pointers present is _____

$n+1$



Null

2 ~~-1~~ 2-1+2

2 + 1 - 1 + 2

Q The no. of leaf nodes in a rooted tree of n nodes, with each node having 0 or 2 childs f.w is

a) $\frac{n+1}{2}$ b) $\frac{n-1}{2}$ c) $\frac{n}{2}$ d) $n-1$

$2L = n+1$

$L = \left(\frac{n+1}{2}\right)$

Total nodes $= 2 \cdot I + 1 \Rightarrow n = 2I + 1$

$n = 2(n-L) + 1 \Rightarrow n = 2n - 2L + 1$

Q A Comp K-ary tree T is a tree in which a node has 0 or k childs exactly. The no. of leaf nodes in tree T if there are exactly 'p' internal nodes is -

a) $(k-1)p+1$

b) $p(k+1)$

c) $pk+p+1$

d) $p(k+1)-p$

e) $p(k-1)+1$

Incorrect

a) S1 b) S2 c) S1, S3 d) S3

j) Consider the statements:

S1 : It is possible to construct a binary tree uniquely whose post-order & pre-order traversal are given

S2 : It is possible _____ a binary tree uniquely whose in-order & post-order trav. are given.

S3 : It is possible _____ _____ . _____ whose post order traversal & level order traversal is given

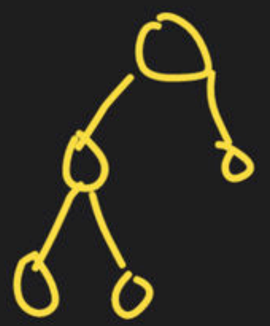
unacademy

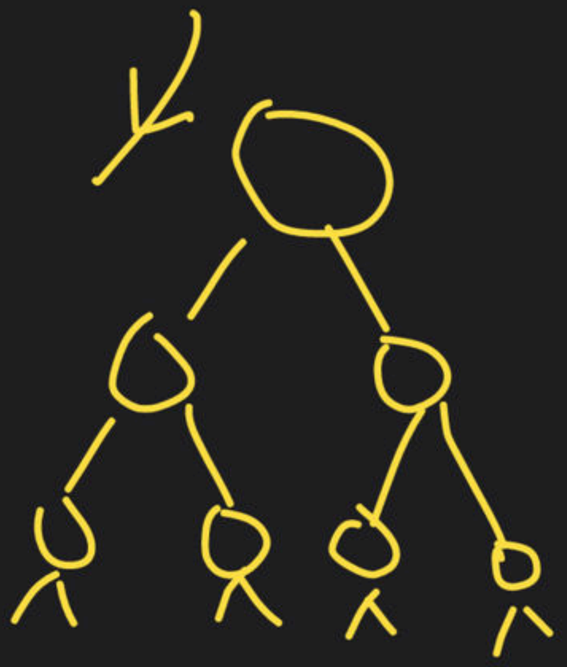
In, Pre → binary tree

In, Post → binary tree

FBT | CBT

only Preorder:
only level-order:
only in-order:
only post order:





Q) The no. of unlabelled binary trees with 4 nodes is

$$\Rightarrow \frac{{}^{2n}C_n}{n+1}$$

Q) The no. of labelled binary trees possible with nodes 10, 30, 20, 40

Q Consider the statements:

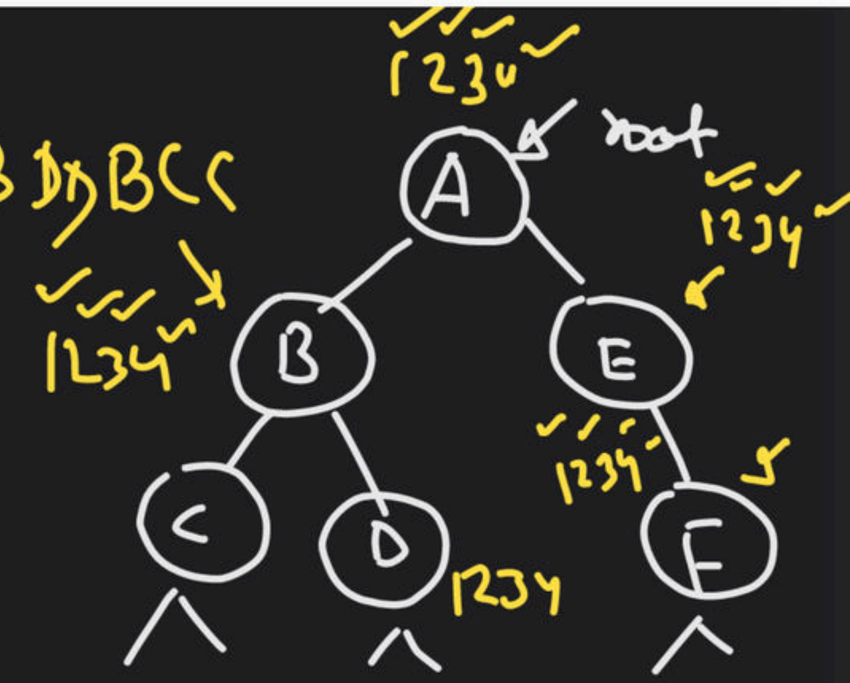
S1: The last element in the post order and in-order traversal of a BST are always same. → False

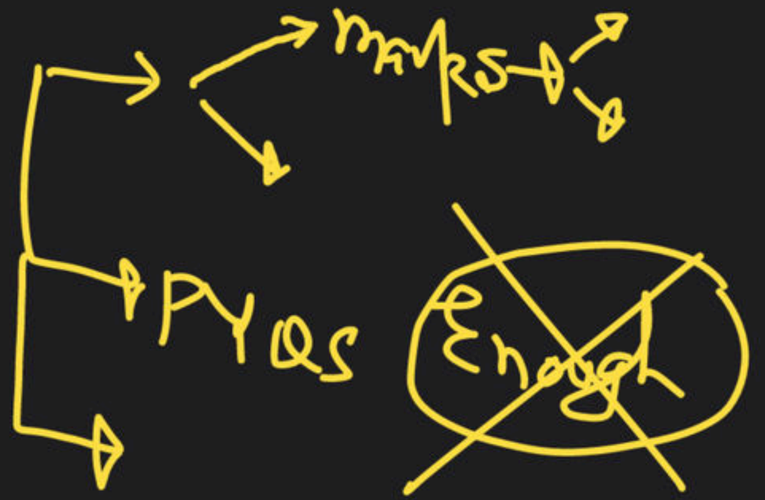
S2: The last element in the Pre order and In-order traversal of a binary tree are always same.

unacademy

```
void func(struct Node *p)
{
    if(p) {
        1. pf("%d", p->data);
        2. func(p->right);
        3. pf("%d", p->data);
        4. func(p->left);
           }
}
```

AEFFEAB BBCC





unacademy

Q

```
int func(struct Node *ptr)
{
  if (ptr == NULL) return 0;
  else if (ptr->Left == NULL && ptr->Right == NULL)
    return 1;
  else
    return 1 + func(ptr->Left) + func(ptr->Right);
}
```

No. of nodes Count

unacademy

Q1

```
Void func(struct Node *Ptr){
    while (Ptr->Left != NULL)
        Ptr = Ptr->Left;
    printf("%d", Ptr->data);
}
```

{ add of root node of BST is passed

{ Non-Empty BST

Q

S1: The min. no. of nodes in a CBT of height h is 2^{h+1} ✗

S2: A BST is always a CBT. ✗

S3: The min no. of nodes in a CBT of height h is 2^h ✓

Correct

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

Q S1: An AVL tree is a node balanced CBT ✗

S2: A heap is necessarily a CBT.

height balanced

unacademy

Q — No. of max-heaps possible with keys 10, 20, 30, 40, 50, 60, 70, 80.

(root) 1 | 7

$F(8) = {}^{7}C_4 \times F(4) \times F(3)$

$\frac{7 \times 6 \times 5}{3!} \times 3 \times 2 \Rightarrow 7 \times 30$

210

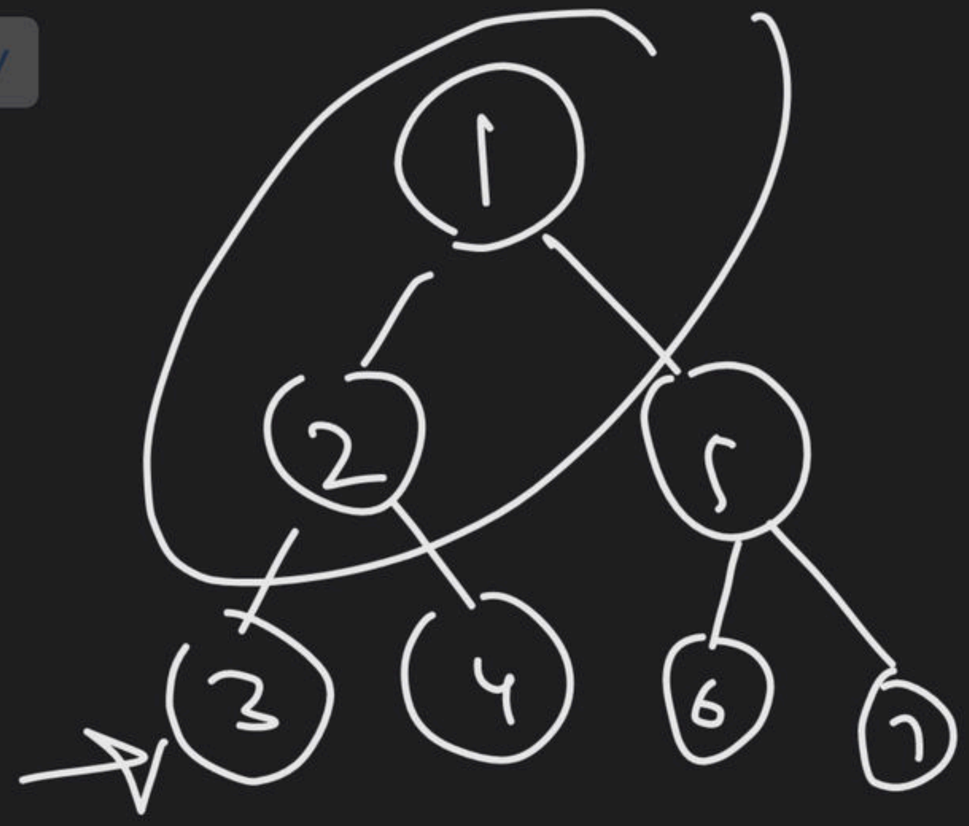
$1 \times {}^{3}C_2 \times F(2) \times F(1) = 3^{\times(1)}$

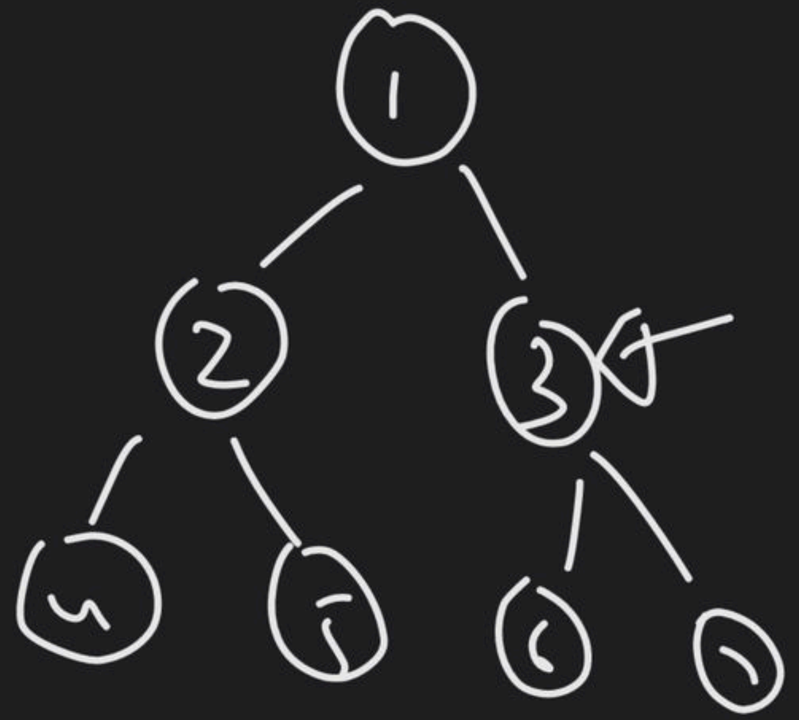
Q1 S1: If the root node of BST is deleted, it can be replaced by inorder predecessor.

S2: If the root node of BST is deleted, it can be replaced by preorder succ.

S3: In a min-heap, kth min can be present at level k.

S4: The accepted bal. factor in AVL tree are 0, +1, -1

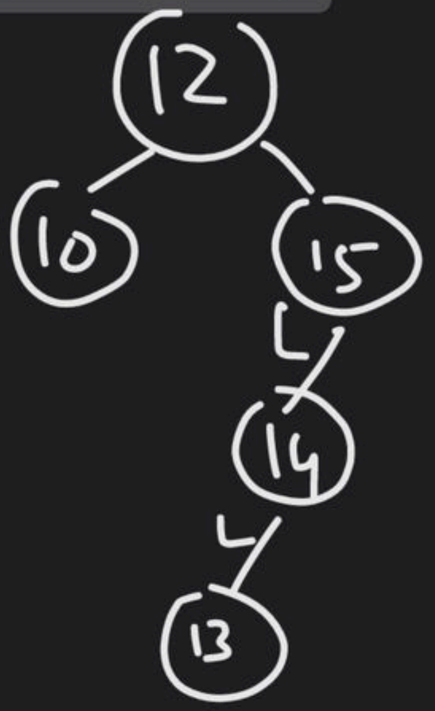


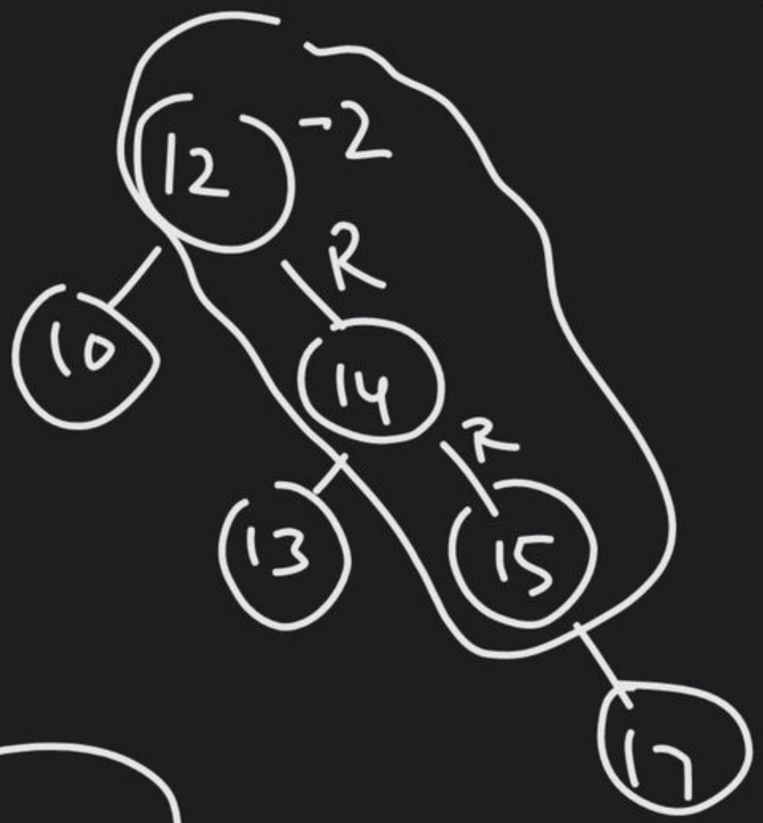


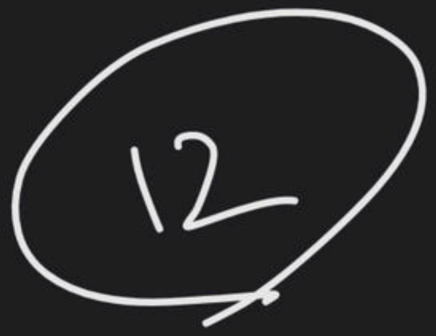
Q1 Construct an AVL tree with the following keys:

12, 10, 15, 14, 13, 17, 8

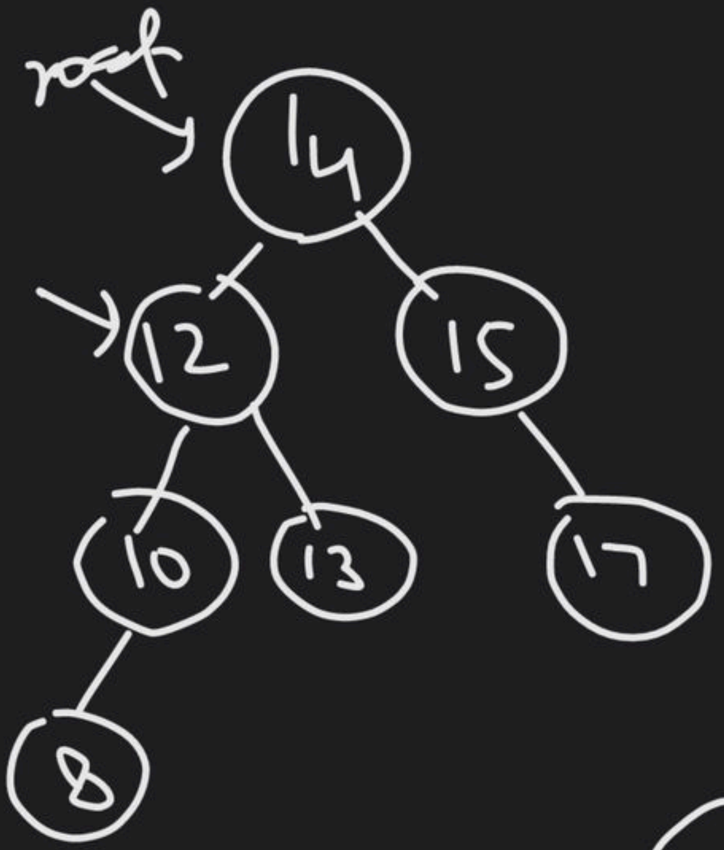
Left child key value of root node

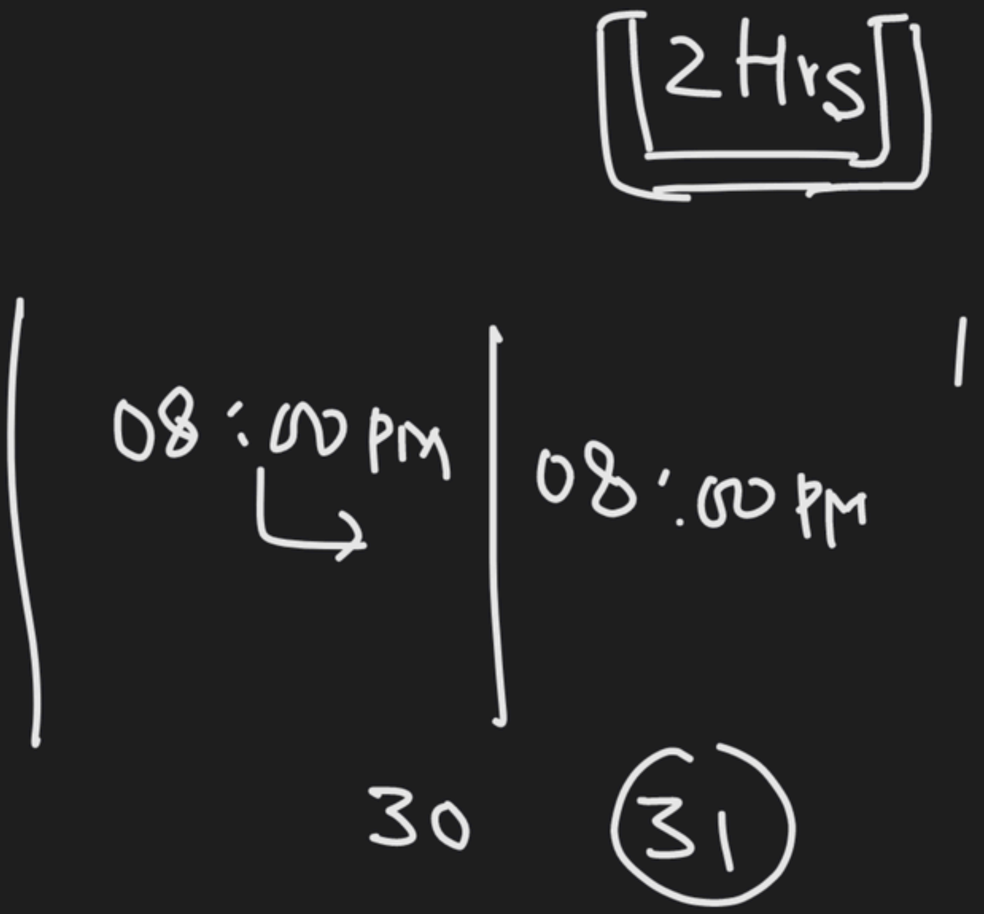


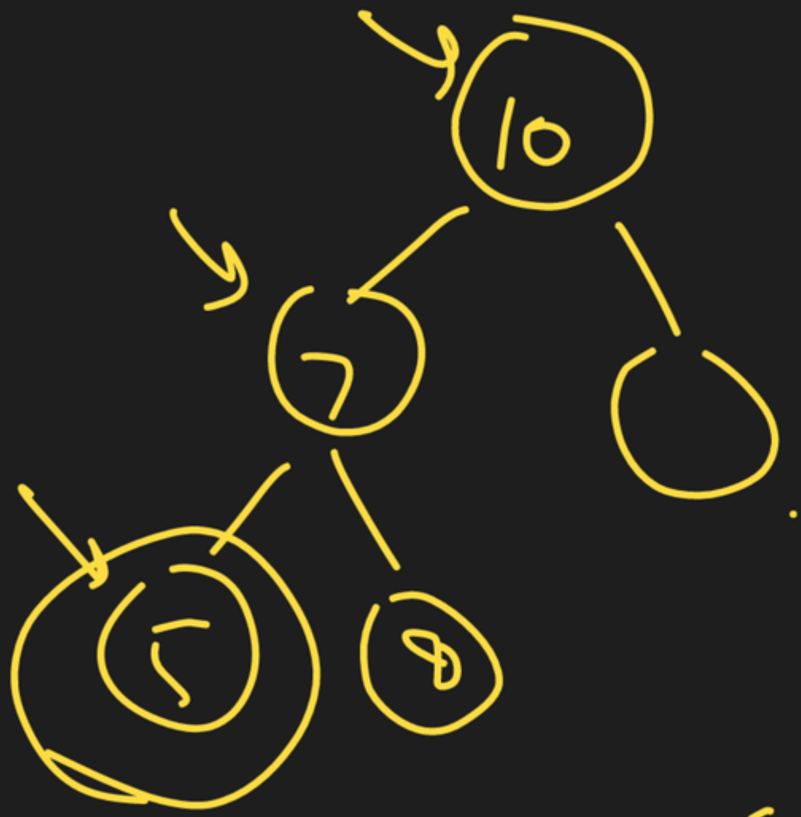


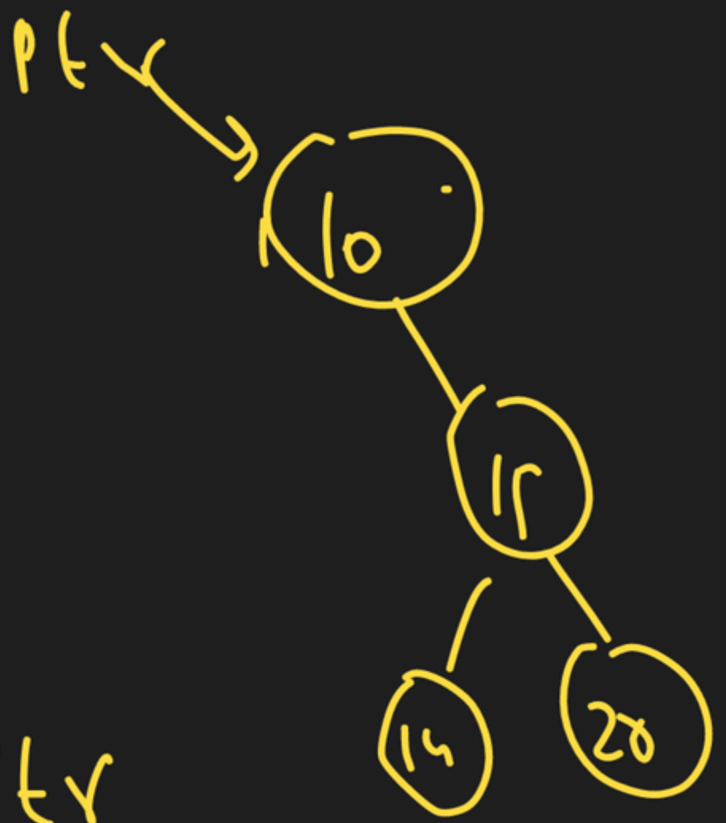


12, 10, 15, 14, 13, 17, 8









Ptr

NULL

unacademy

THANK YOU!

Here's to a cracking journey ahead!