CS & IT ENGINEERING

Data Structure

Linked List

DPP

Discussion Notes

By- Pankaj Sharma sir

01 Question

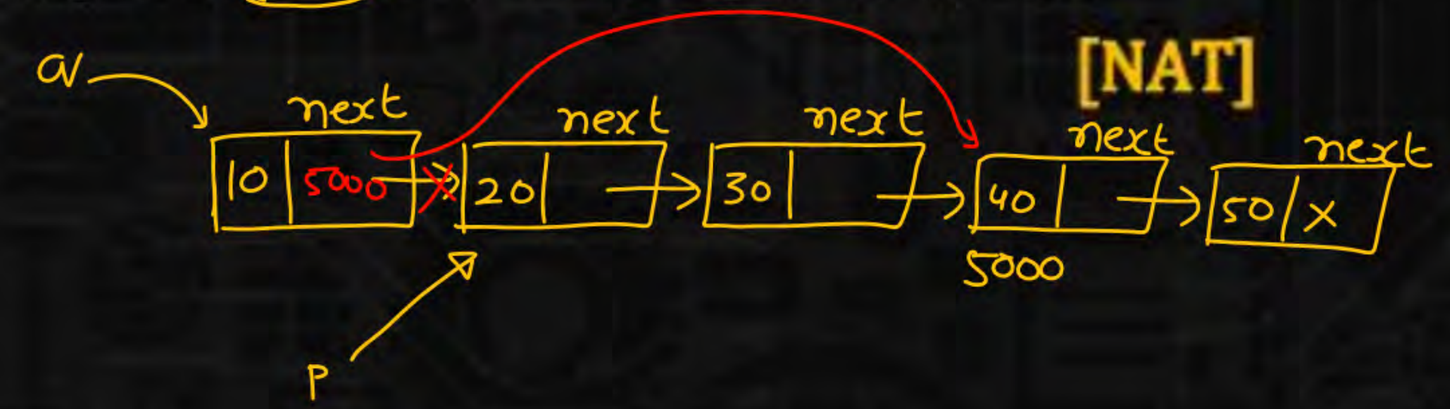
02 Discussion

Q.1

[NAT]

Consider a single linked list q with 2023 elements is passed to the following function:

```
struct node {
    int data;
    struct node *next;
};
void f(struct node *q){
    struct node *p;
    p=q->next;
    q->next=p->next->next;
}
```



The size of the linked list q after the execution of the function is _____.

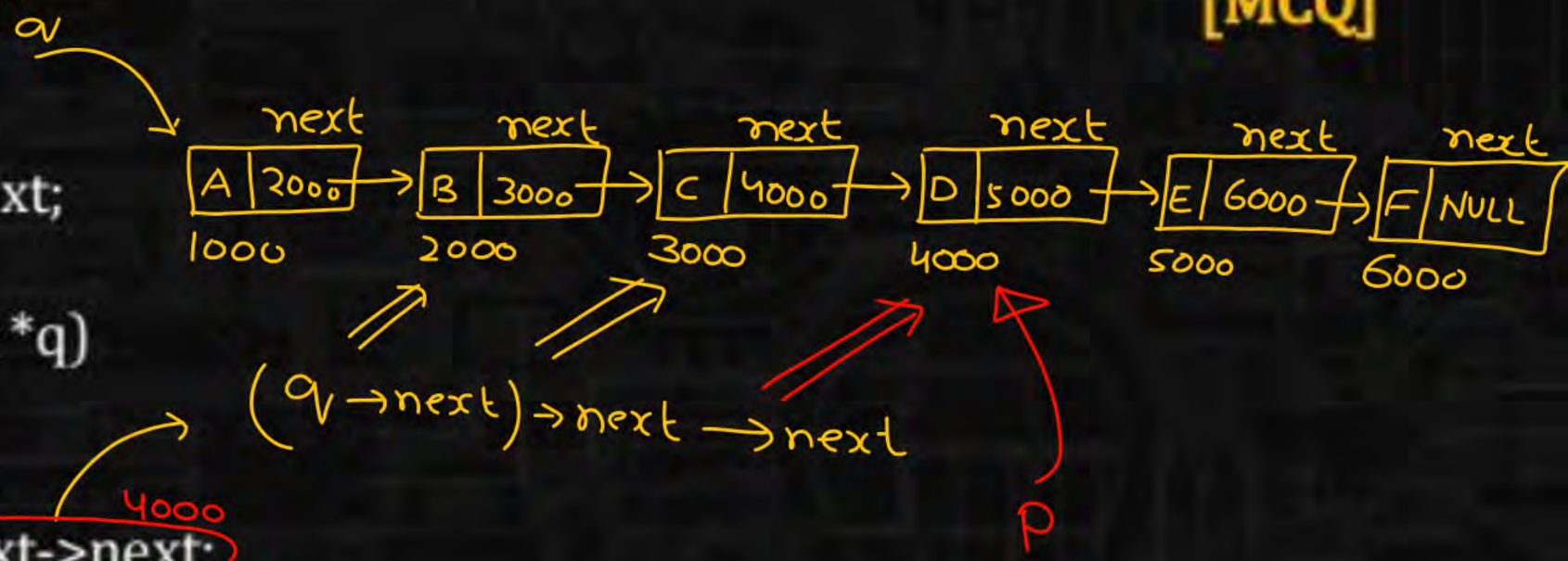
2023-2 = 2021

Q.2

Consider a single linked list q['A', 'B', 'C', 'D', 'E', 'F'] is passed to the following function:

[MCQ]

```
struct node {
    int data;
    struct node *next;
};
void f(struct node *q)
{
    struct node *p;
    p=q->next->next->next;
    q->next->next->next=p->next->next;
    p->next->next=q->next;
    printf("%c", p->next->next->next->data);
}
```



The output is-

A. C B. D C. E D. B

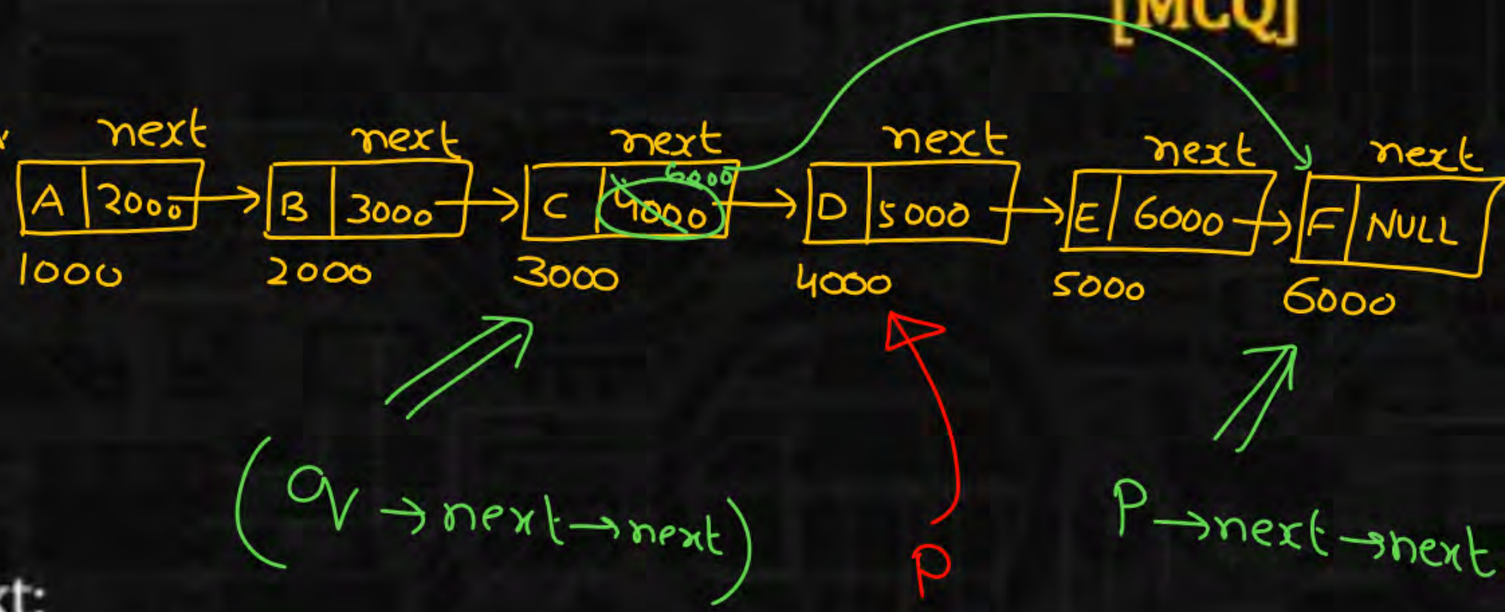
Q.2

[MCQ]

Consider a single linked list q['A', 'B', 'C', 'D', 'E', 'F'] is passed to the following function:

```
struct node {
    int data;
    struct node *next;
};
void f(struct node *q)
{
    struct node *p;
    p=q->next->next->next;
    q->next->next->next=p->next->next;
    p->next->next=q->next;
    printf("%c", p->next->next->next->data);
}
```

The output is-



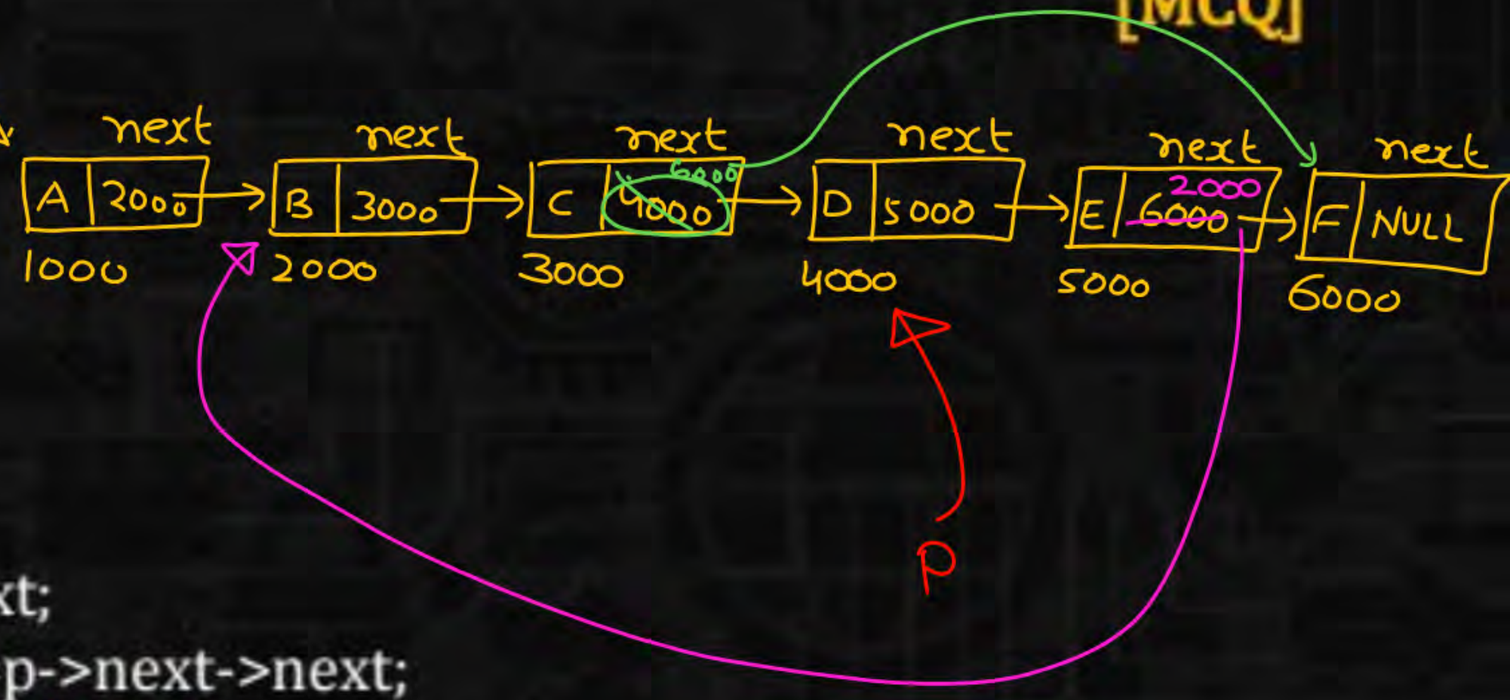
A. C B. D C. E D. B

Q.2

[MCQ]

Consider a single linked list q['A', 'B', 'C', 'D', 'E', 'F'] is passed to the following function:

```
struct node {
    int data;
    struct node *next;
};
void f(struct node *q)
{
    struct node *p;
    p=q->next->next->next;
    q->next->next->next=p->next->next;
    p->next->next=q->next;
    printf("%c", p->next->next->next->data);
}
```



(A)

(2000)

C

The output is-

A. C B. D C. E D. B

Q.3

[MCQ]

Consider the following statements:

P: Linked Lists supports linear accessing of elements Correct

Q: Linked Lists supports random accessing of elements. Incorrect

Which of the following statements is/are INCORRECT?

A. P only

B. Q only

C. Both P and Q

D. Neither P nor Q

(B)

Q.4

Consider a single linked list q['A', 'B', 'C', 'D'] is passed to the following function:

[MCQ]

```
void f(struct node *q)
{
    if(q==NULL) return;
1.  f(q->next);
2.  printf("%c ", q->data);
}
```

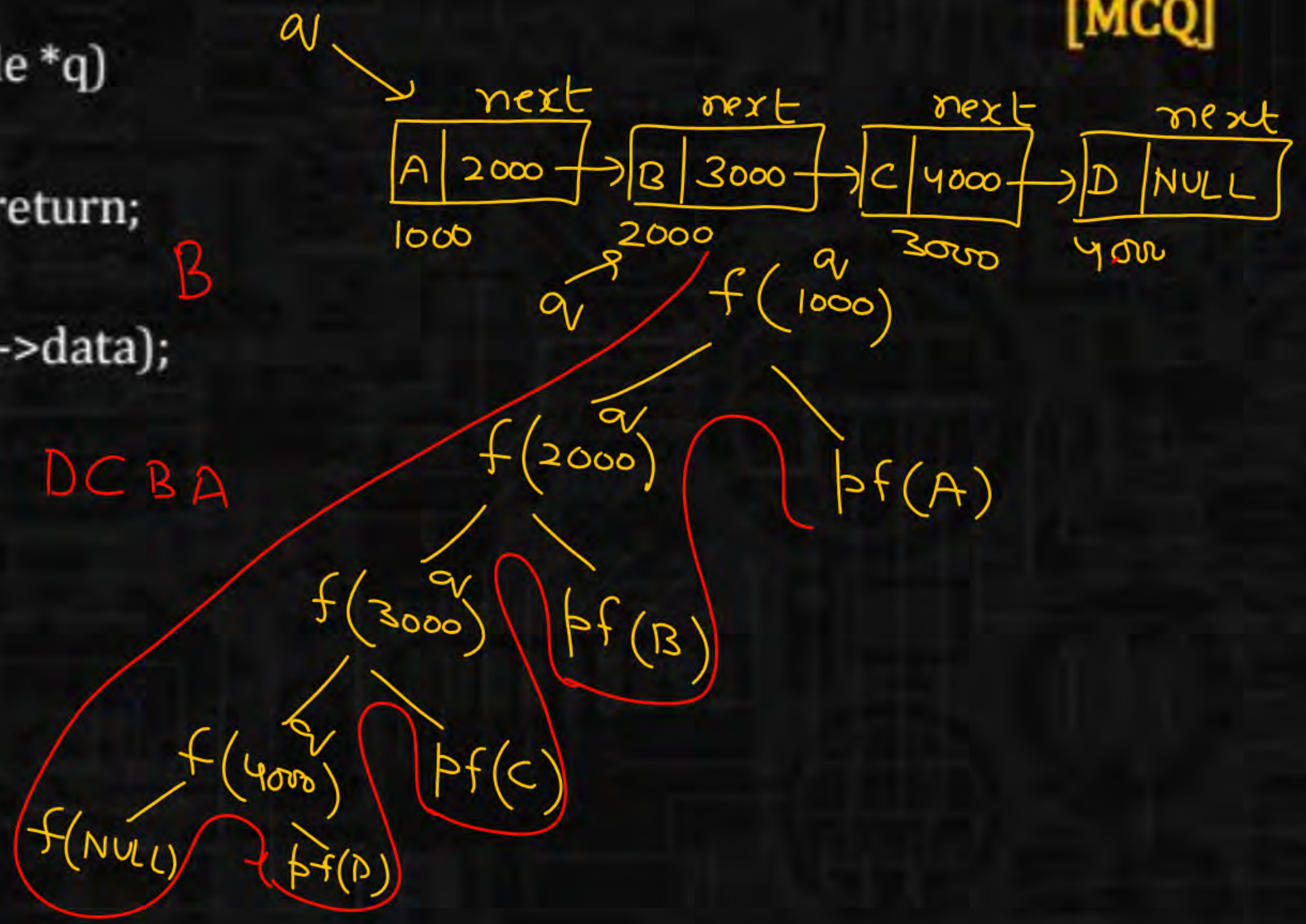
The output is-

A. C D B A

B. D C B A

C. A B C D

D. B C D A



Q.5

[MCQ]

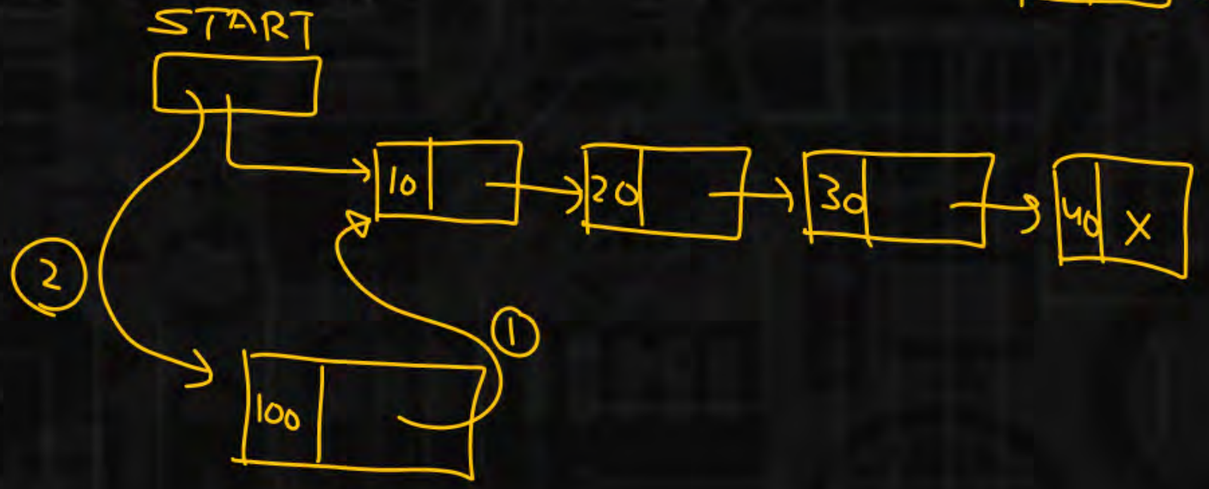
Consider the following statements:

Correct

P: Insertion at the end of the linked list is difficult than insertion at the beginning of the linked list.

Q: Deletion at the beginning of linked list is easier as compared to deletion at the end of the linked list.

Which of the following statements is/are CORRECT?



A. Both P and Q

B. P only

C. Q only

D. Neither P nor Q

Q.5

[MCQ]

Consider the following statements:

Correct

P: Insertion at the end of the linked list is difficult than insertion at the beginning of the linked list.

Correct

Q: Deletion at the beginning of linked list is easier as compared to deletion at the end of the linked list.

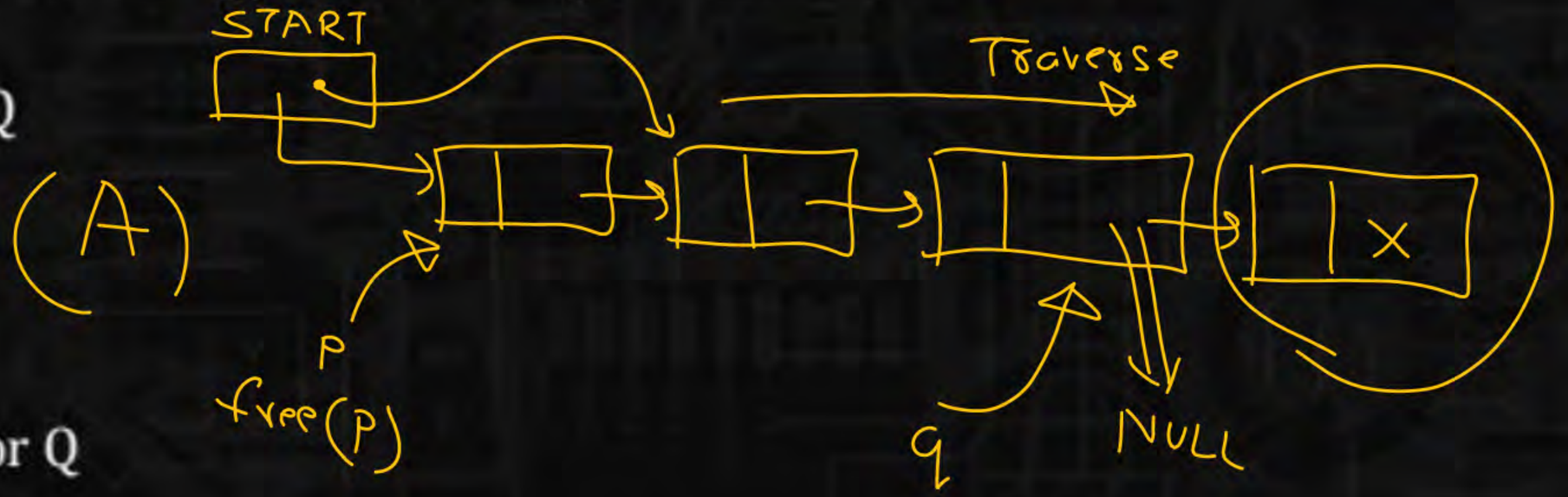
Which of the following statements is/are CORRECT?

A. Both P and Q

B. P only

C. Q only

D. Neither P nor Q

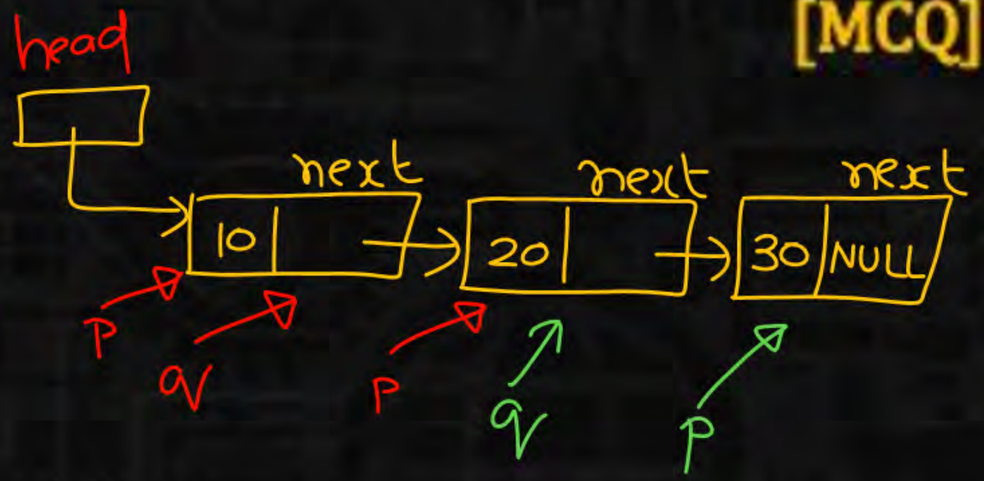


Q.6

The following C function takes a single-linked list p of integers as a parameter. It deletes the last element of the single linked list. Fill in the blank space in the code:

[MCQ]

```
struct node {
        int data;
        struct node *next;
};
void delete_last(struct node *head)
{
        struct node *p=head, *q;
        if(!head) return;
        if(head->next==NULL){free(head);head=NULL;
        return;}
        while(_____a_____){
          q = p;
          p=p->next;
        }
        _____b_____;
        free(p);
        q=NULL; p=NULL;
        }
```



(C)

p->next != NULL

q->next = NULL

q->next = NULL

free(p)

A. a: !head ; b: q->next = NULL;

B. a: p->next ! = head ; b: q->next = q

C. a: p->next ! = NULL ; b: q->next = NULL

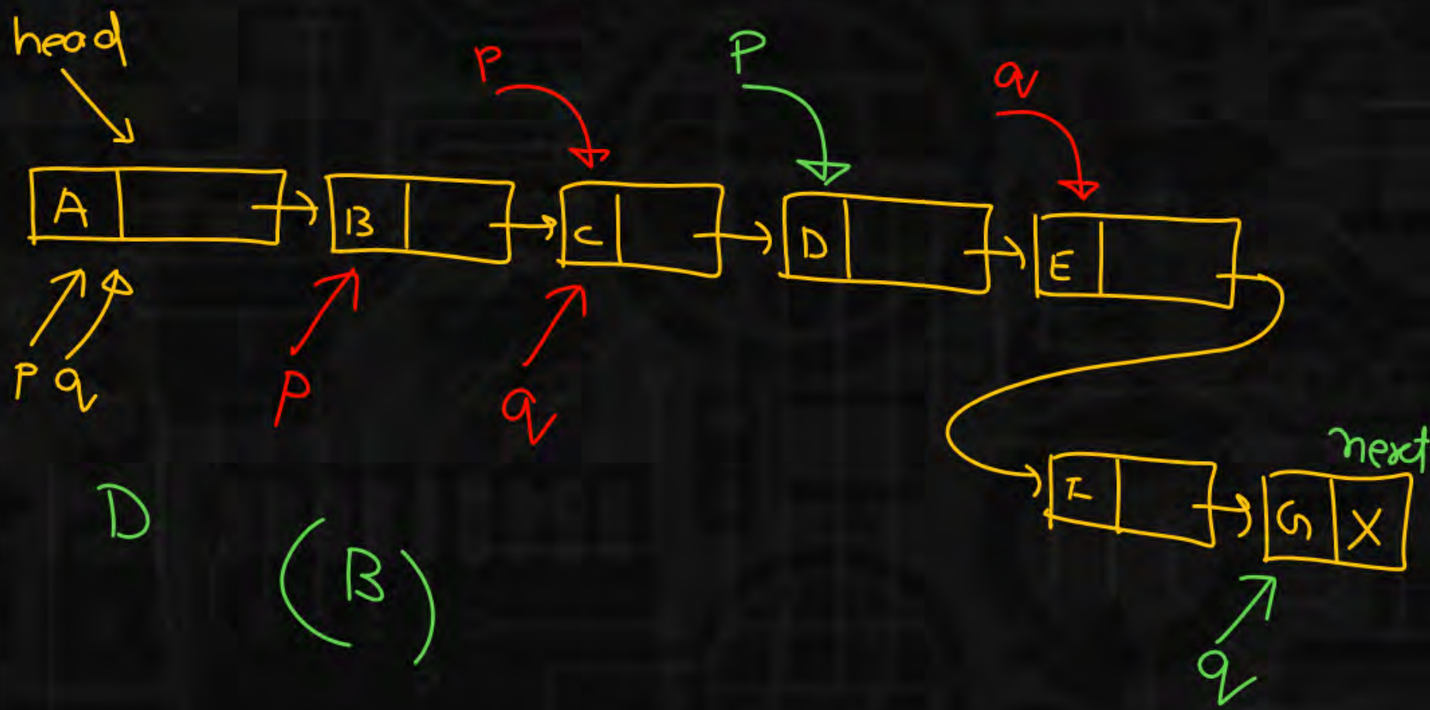
D. a: head->next ! = p ; b: q->next = p

Q.7

Consider a single linked list q[['A', 'B', 'C', 'D', 'E', 'F', 'G'] is passed to the following function:

[MCQ]

```
void func(struct node *head){
    struct node *p=head, *q=head;
    while(q!=NULL && q->next!=NULL && q->next->next != NULL){
        p=p->next;
        q=q->next->next;
    }
    printf("%c", p->data);
}
```



D (B)

The output is-

A. C

B. D

C. E

D. B

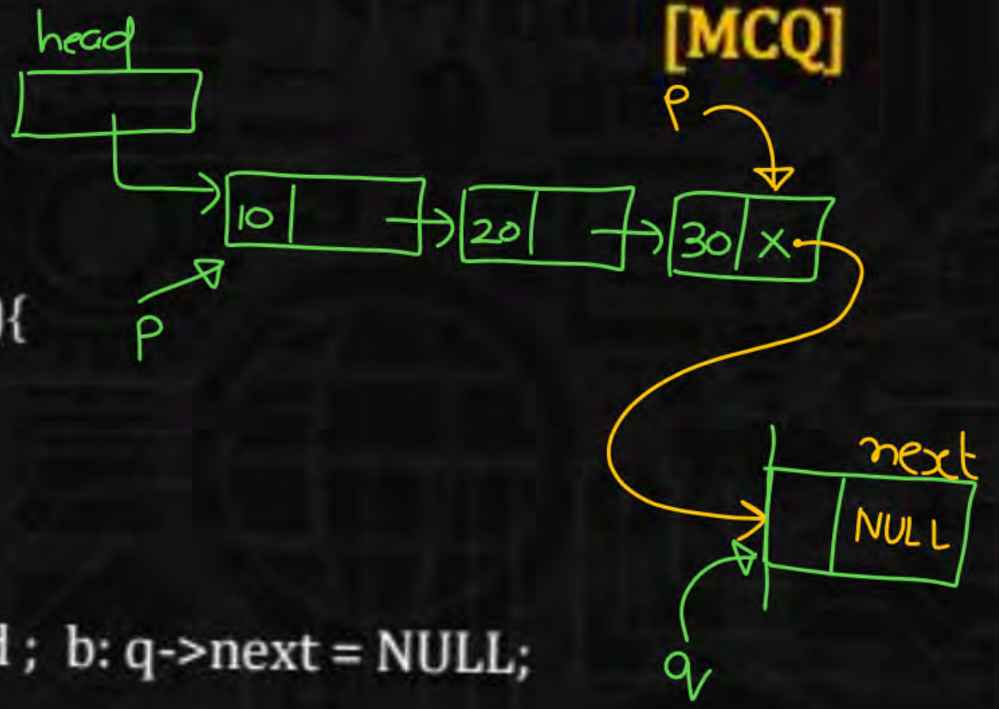
Q.8

[MCQ]

The following C function takes a single-linked list p of integers as a parameter. It inserts the element at the end of the single linked list. Fill in the blank space in the code:

```
struct node
{
    int data;
    struct node *next;
};
void insert_last(struct node *head, struct node *q){
struct node *p=head;
if(!head) return;
while(_____a_____)
        p=p->next;
        _____b_____;
        q=NULL;
        p=NULL;
}
```

Assume, q is the address of the new node to be added.



p->next != NULL

A. a: !head ; b: q->next = NULL;

B. a: q->next ! = NULL; b: p->next = q

C. a: p->next ! = NULL ; b: p->next = q

D. a: head->next ! = p ; b: q->next = p

Q.1

Consider a linked list [a, b, c, d, e]. ptr is a pointer pointing to the head/start node. Assume a node in the linked list is defined as:

```
struct node
{
int data;
struct node *next;
};
```

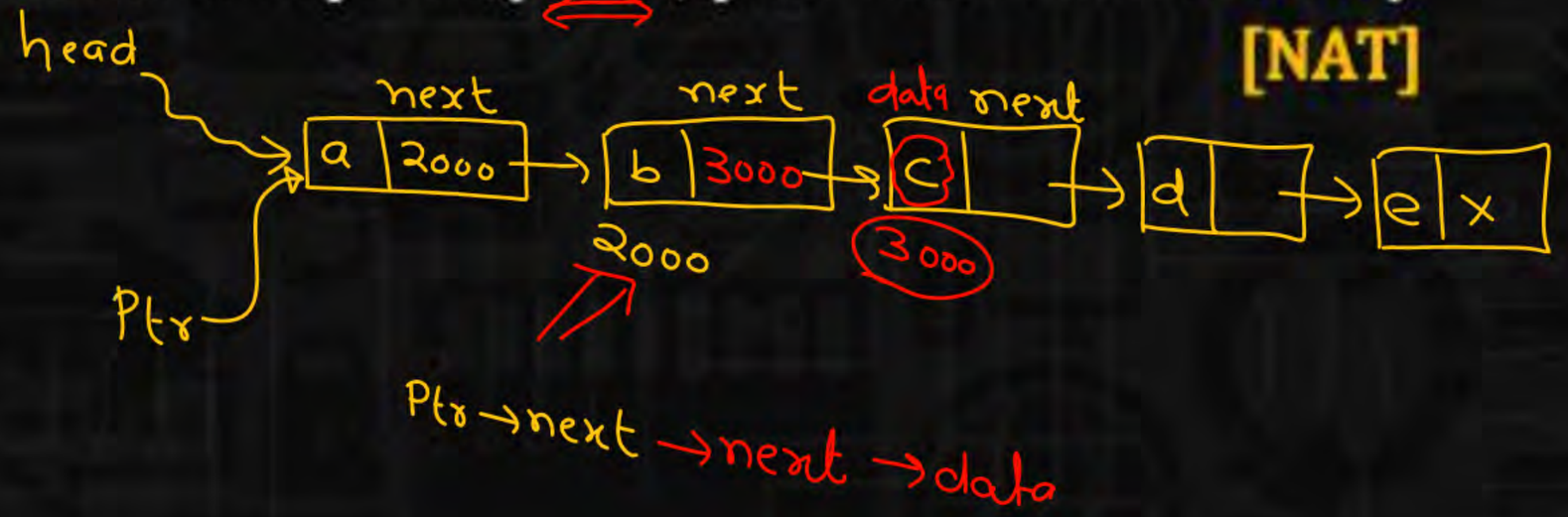
a – 97
b – 98
c – 99

Ascii

C

The output of the statement printf("%d", ptr->next->next->data) is 99.

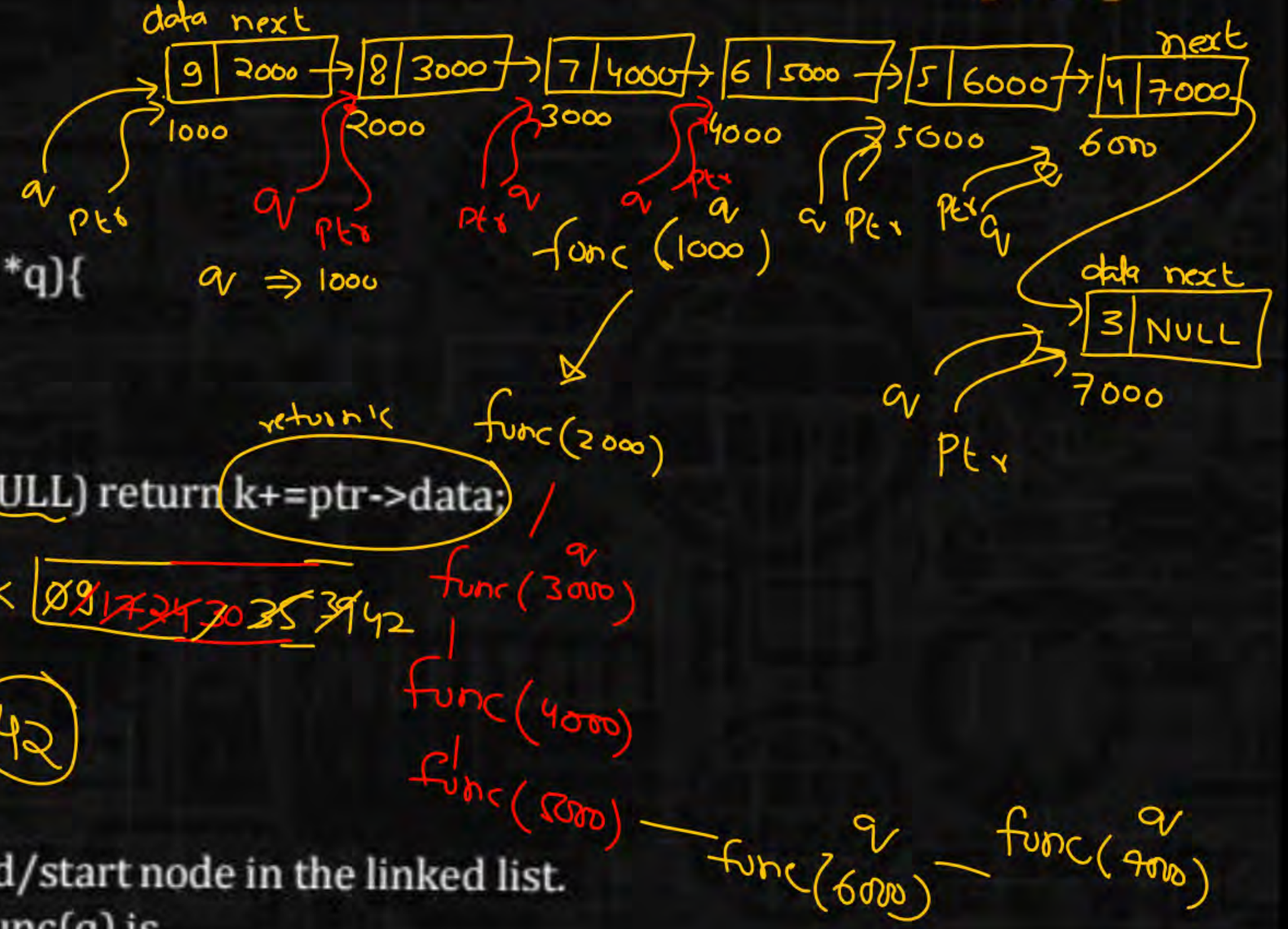
[NAT]



Q.2

Consider a single linked list of integers [9, 8, 7, 6, 5, 4,3] is passed to the following function: **[NAT]**

```
struct node
{
int data;
struct node *next;
};
int  func(struct node *q){
static int k=0;
struct node *ptr=q;
    if(!ptr) return 0;
else if(ptr->next==NULL) return k+=ptr->data;
else{
    k+=ptr->data;
 func(ptr->next);
    return k;
}
}
```



Assume, q points to head/start node in the linked list.

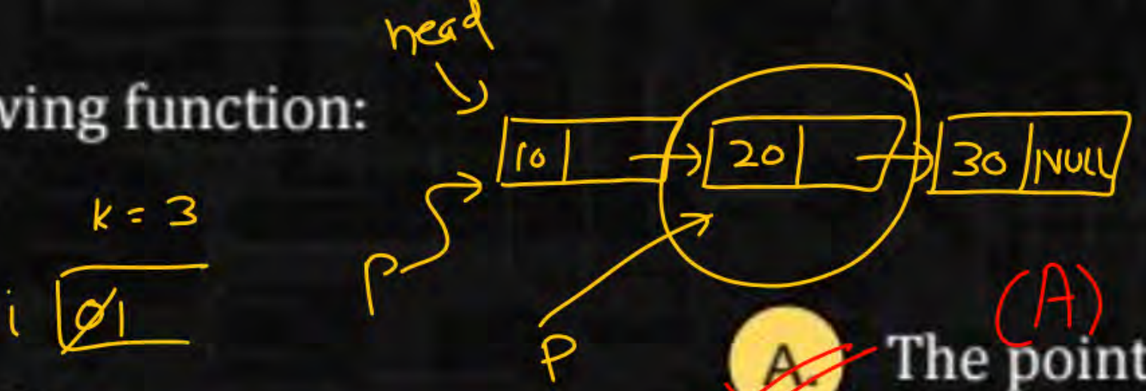
The value returned by func(q) is _____.

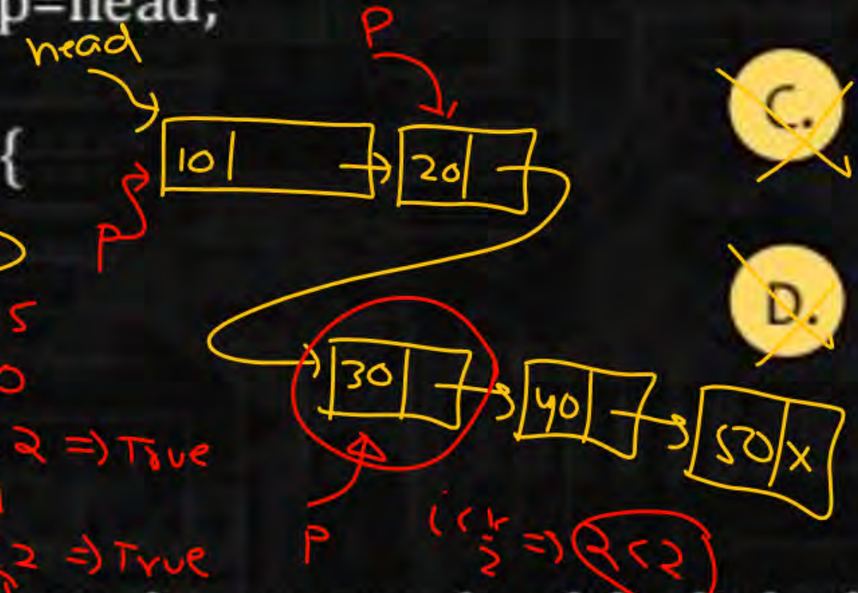
Q.3

[MCQ]

Consider the following function:

```
struct node
{
int data;
struct node *next;
};
struct node * f(struct node *head, int k){
    struct node *p=head;
        int i=0;
    while(i<k/2){
    p=p->next;
    i++;
    }
    return p;
}
```





(A)

A. The pointer to the middle element in the linked list.

B. The pointer to the second element in the linked list.

C. The middle element in the linked list.

D. The second last element in the linked list.

Assume head points to the start node of the linked list and k is the number of elements in the linked list, the function returns-

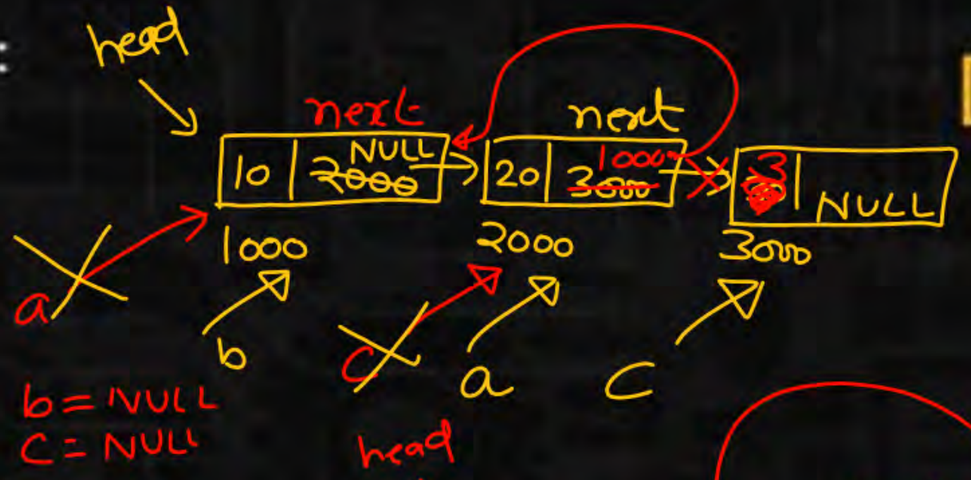
Q.4

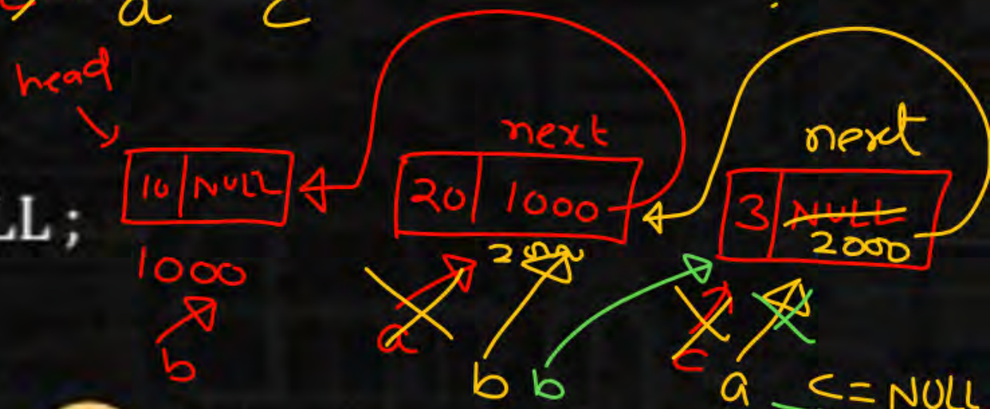
[MCQ]

Consider the following function:

```
struct node
{
    int data;
    struct node *next;
};
    void f(struct node *head){
    struct node *a=head, *b=NULL, *c=NULL ;
    while(a){
    c=a->next;
    a->next=b;
    b=a;
    a=c;
    }
    head=b;
}
```

(C)

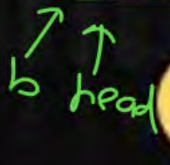




A. Sorts the linked list.

B. Interchanges every two consecutive elements in the list.

C. Reverses the list.



D. None of the above.

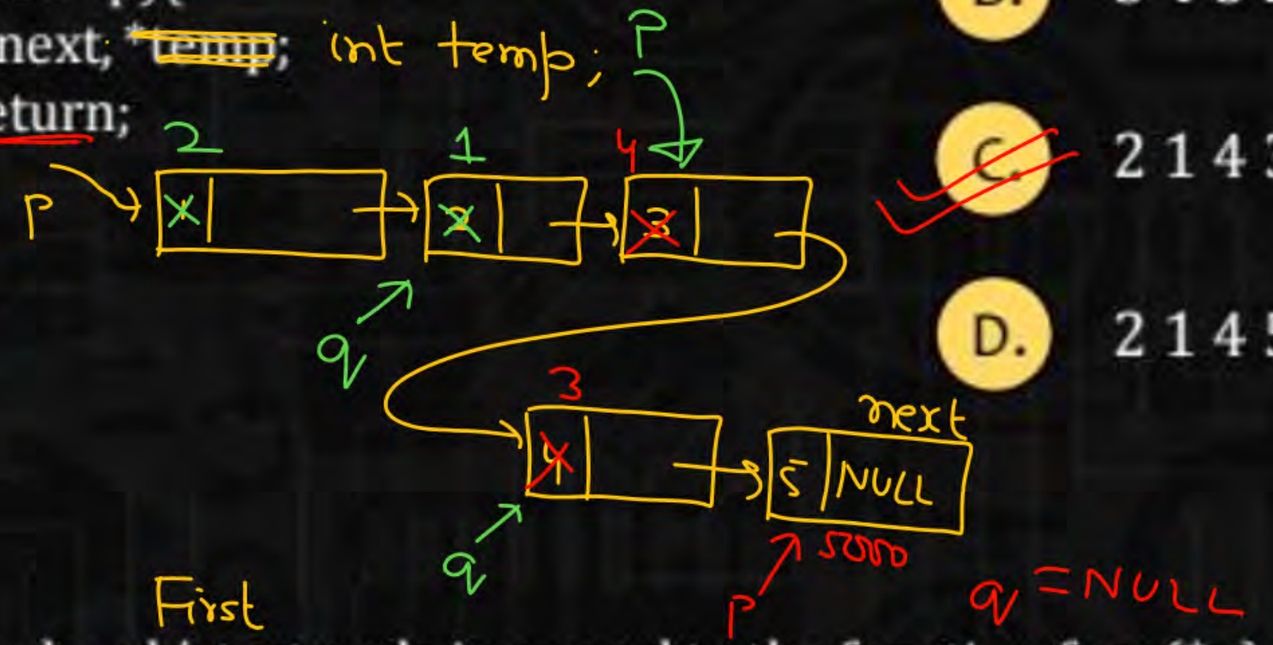
Assume head points to the start node of the linked list, the function-

Q.5

Consider a single linked list [1, 2, 3, 4, 5] is passed to the following function:

[MCQ]

```
struct node
{
int data;
struct node *next;
};
    void  func(struct node *p){
    struct node *q=p->next; *temp; int temp;
    if(!p||!(p->next)) return;
    else{
    temp=q->data;
q->data=p->data;
    p->data=temp;
func(p->next->next);
        }
}
```



A. 2 3 4 5 1

B. 5 4 3 2 1

C. 2 1 4 3 5

D. 2 1 4 5 3

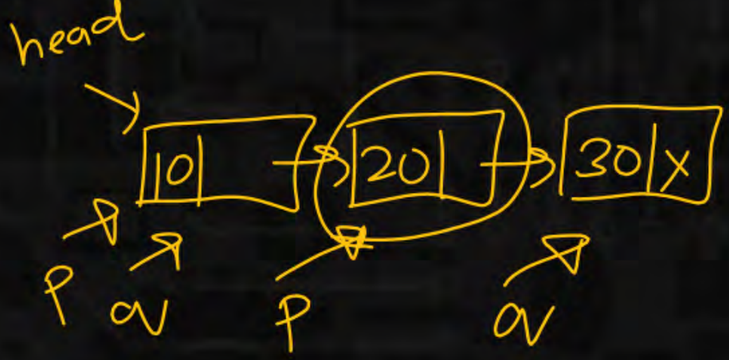
Initially, the address of the head/start node is passed to the function func(*p), the arrangement of the linked list after function execution is –

Q.6

[MCQ]

Consider the following function:

```
struct node
{
    int data;
    struct node *next;
};
    struct node * f(struct node *head){
    struct node *p=head, *q=head;
while(q!=NULL && q->next!=NULL && q-
>next->next != NULL){
    p=p->next;
q=q->next->next;
 if(p==q) break;
 }
return p;}
```



Assume head points to the start node of the linked list, the function-

Options ahead

A. Returns the pointer to the node where a cycle/loop ends (assume the leftmost node in a loop is the start node of a cycle).

B. Returns the pointer to the node where a cycle/loop starts (assume the leftmost node in a loop is the start of a cycle).

C. Reverses the list.

D. Detects a cycle in the list.

(E)

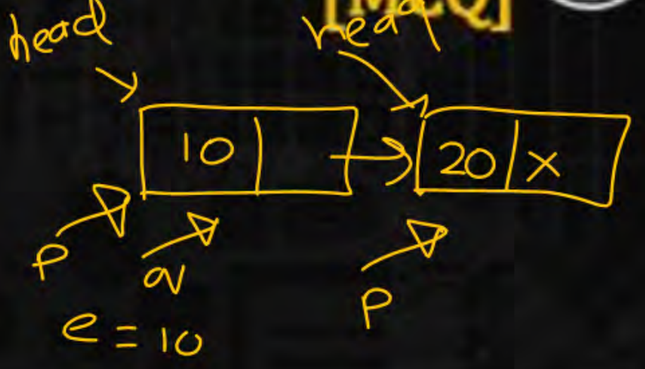
E. None of these ✓

Q.7

[MCQ]

Consider the following function:

```
struct node
{
int data;
struct node *next;
};
void f(struct node *head, int e){
    struct node *p, *q;
    if(head->data==e){
    q=p;
    p=p->next;
    free(q);
    head=p;
    return;
}
    q=head; p=head->next;
    while(p->next!=NULL){
    if(p->data==e){
    _____;
    free(p);
    return;
    }
    q=p;
    p=p->next;
    }
    if(p->data==e){
    q->next=NULL;
    free(p);
    }
    }
```



Assume there are at least two elements in the single linked list of integers. The starting node's address is contained in the head pointer passed to the function. The function f() searches for the element e in the list. If found, the function deletes the node. The missing statements are-

Options ahead

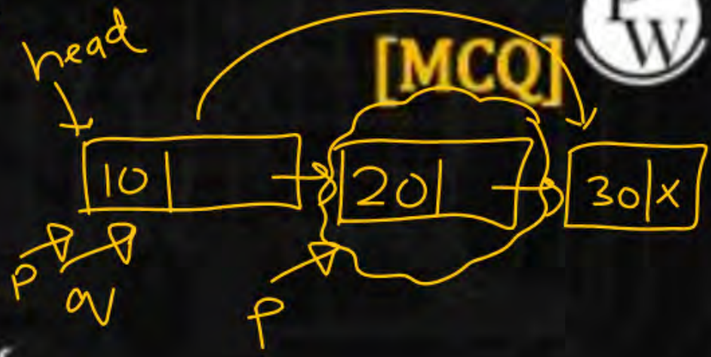
Q.7

Consider the following function:

```
struct node
{
int data;
struct node *next;
};
void f(struct node *head, int e){
    struct node *p, *q;
    if(head->data==e){
    q=p;
    p=p->next;
    free(q);
    head=p;
    return;
}
    q=head; p=head->next;
    while(p->next!=NULL){
    if(p->data==e){
    _____;
    free(p);
    return;
    }
    q=p;
    p=p->next;
    }
    if(p->data==e){
    q->next=NULL;
    free(p);
    }
    }
```

Assume there are at least two elements in the single linked list of integers. The starting node's address is contained in the head pointer passed to the function. The function f() searches for the element e in the list. If found, the function deletes the node. The missing statements are-

Options ahead



q->next = p->next

A. free(q), q->next=p->next

B. free(q), q=p->next

C. free(p), q=p->next

D. free(p), q->next=p->next

Q.8

A node of a linked list is to be created by calling malloc() function. The malloc() returns NULL if-

[MCQ]

A. Stack overflow occurs.

B. Memory leakage occurs.

C. Memory is full.

D. None of the above.


THANK YOU GW
SOLDIERS !