CS & IT ENGINEERING

Data Structure & Programming

Lec No.- 01

1500 Series

By- Pankaj Sharma Sir

Topics to be Covered

Topic Problem Practice

[NAT] [1 Mark]

$i = 0$ $j = 0, 1, 2$

$i = 1$ $j = 0, 1, 2$

$i = 2$ $j = 0, 1, 2$

```
#Q. int main ( ) {
        int A[3] [3];
        int*p=&A[0] [0];
        for(int i =0; i< 3; i++){
            for(int j =0, j <3, j ++){
                switch (i) {
                case 0 :  A[i] [j]; = i*j+3;
                case 1 : * (p+j*3+i) = j*i+1;
                           break
                case 2 : * (p+j*3+i) = j*i+1;
                           break
                }
            }
    }
    printf("%d", A[2][1]-A[2][0]);
    }
```

$A_{00} = 3$

$A_{00} = 1$

$*(p+0+0) = 1$

$*(p) = 1$

The output give 'C' code is_____.

[NAT] [1 Mark]

$i=0$ $j=0,1,2$

$i=1$ $j=0,1,2$

$i=2$ $j=0,1,2$

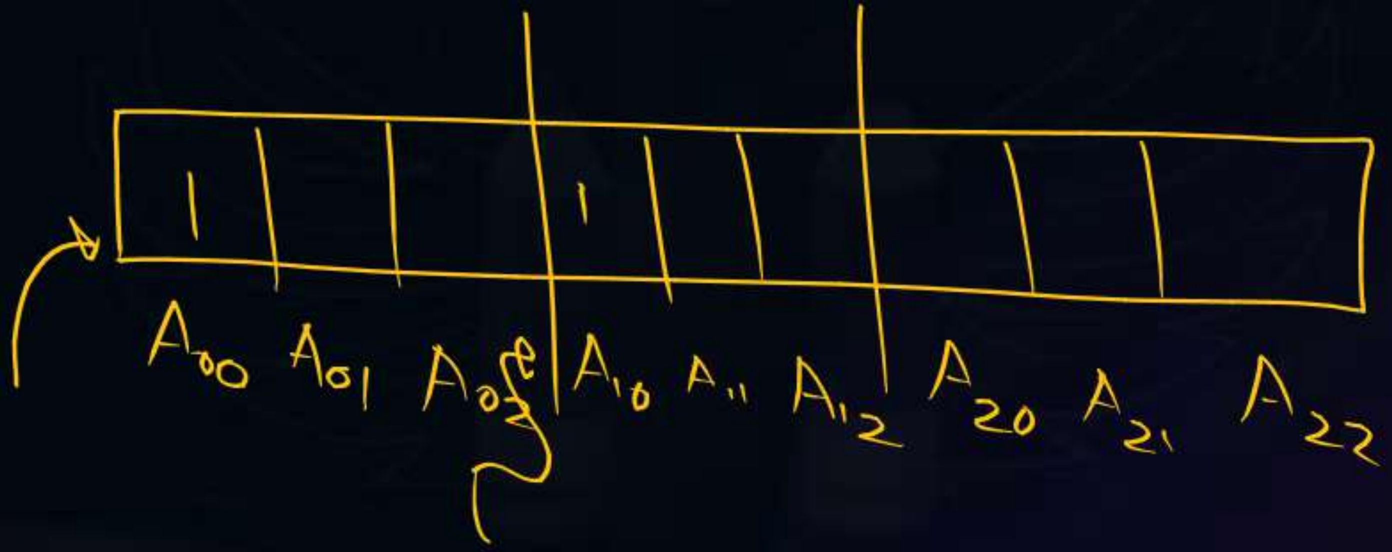
```
#Q. int main ( ) {
       int A[3] [3];
       int*p=&A[0] [0];
       for(int i =0; i< 3; i++){
            for(int j =0, j <3, j ++){
                 switch (i) {
                 case 0 :  A[i] [j]; = i*j+3;
                 case 1 : * (p+j*3+i) = j*i+1;
                             break
                 case 2 : * (p+j*3+i) = j*i+1;
                             break
                 }
            }
       }
       printf(“%d”, A[2][1]–A[2][0]);
       }
```

0

$A_{01} = 3$

$*(p+3) = 1$

The output give ‘C’ code is_____.



[NAT]

[1 Mark]

i=0
j=0,1,2

i=1
j=0,1,2

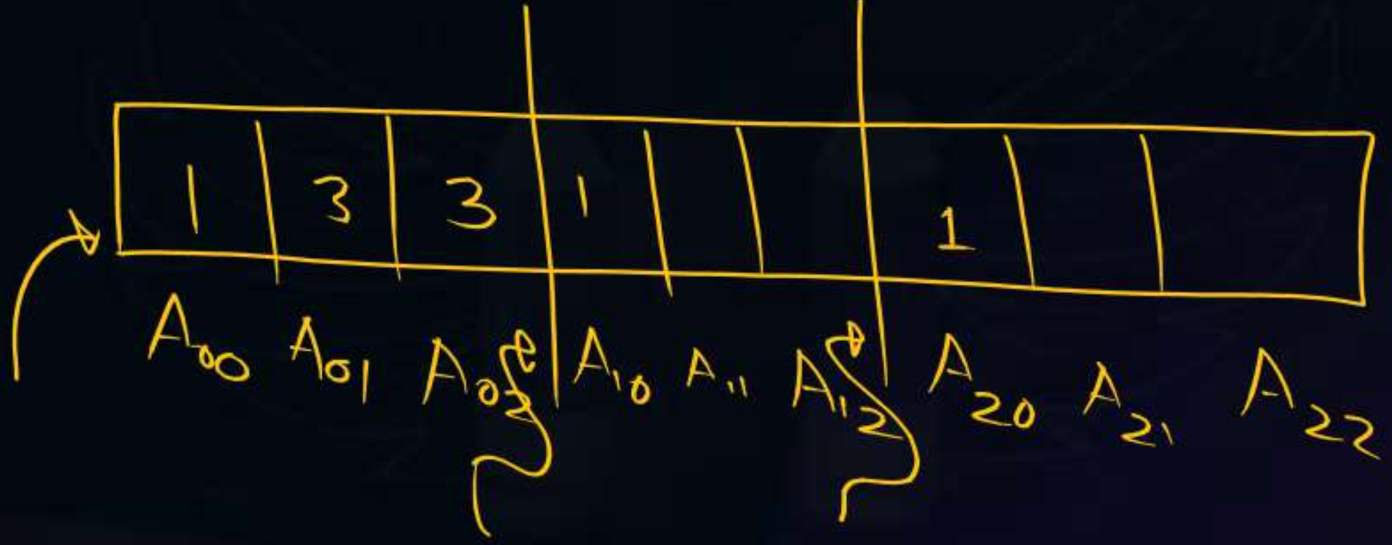
i=2
j=0,1,2

```
#Q. int main ( ) {
        int A[3] [3];
        int*p=&A[0] [0];
        for(int i =0; i< 3; i++){
            for(int j =0, j <3, j ++){
                switch (i) {
                case 0 :  A[i] [j]; = i*j+3;
                case 1 : * (p+j*3+i) = j*i+1;
                        break
                case 2 : * (p+j*3+i) = j*i+1;
                        break
                }
            }
        }
        printf("%d", A[2][1]-A[2][0]);
    }
```

$A_{02} = 3$

$*(p+6) = 1$

The output give 'C' code is_____.



[NAT] [1 Mark]

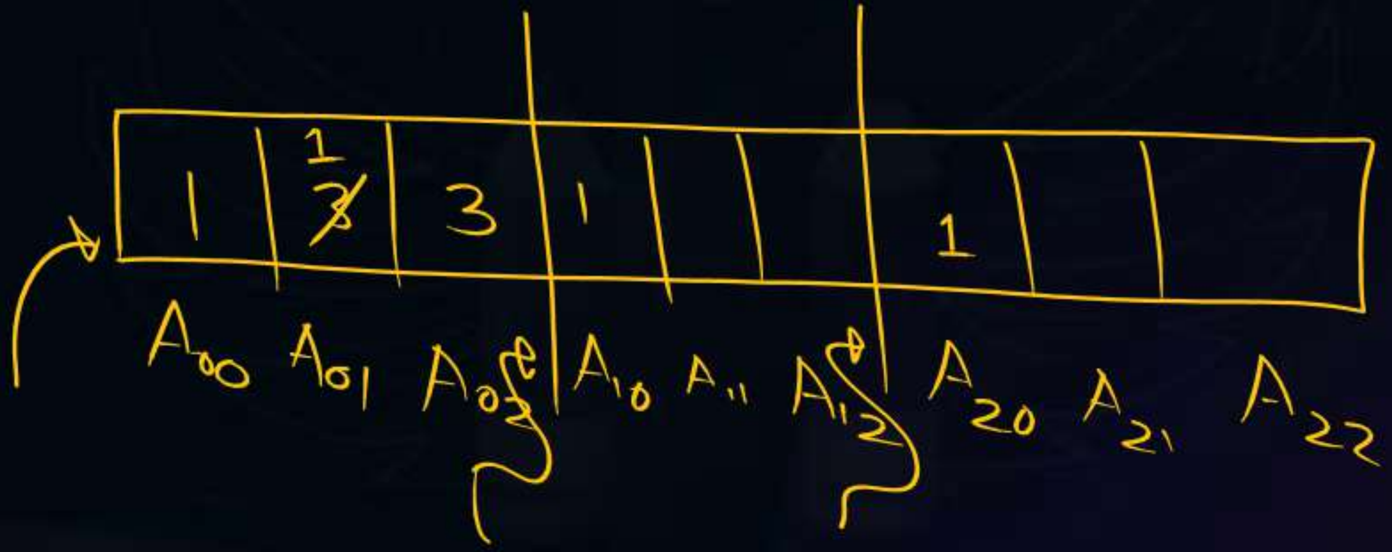
#Q.
```
int main ( ) {
    int A[3] [3];
    int*p=&A[0] [0];
    for(int i =0; i< 3; i++){
        for(int j =0, j <3, j ++){
            switch (i) {
            case 0 : A[i] [j]; = i*j+3;
            case 1 : * (p+j*3+i) = j*i+1;
                     break
            case 2 : * (p+j*3+i) = j*i+1;
                     break
            }
        }
    }
    printf("%d", A[2][1]–A[2][0]);
}
```

i=1 j=0,1,2

i=2 j=0,1,2

$*(p+1) = 1$

The output give 'C' code is_____.



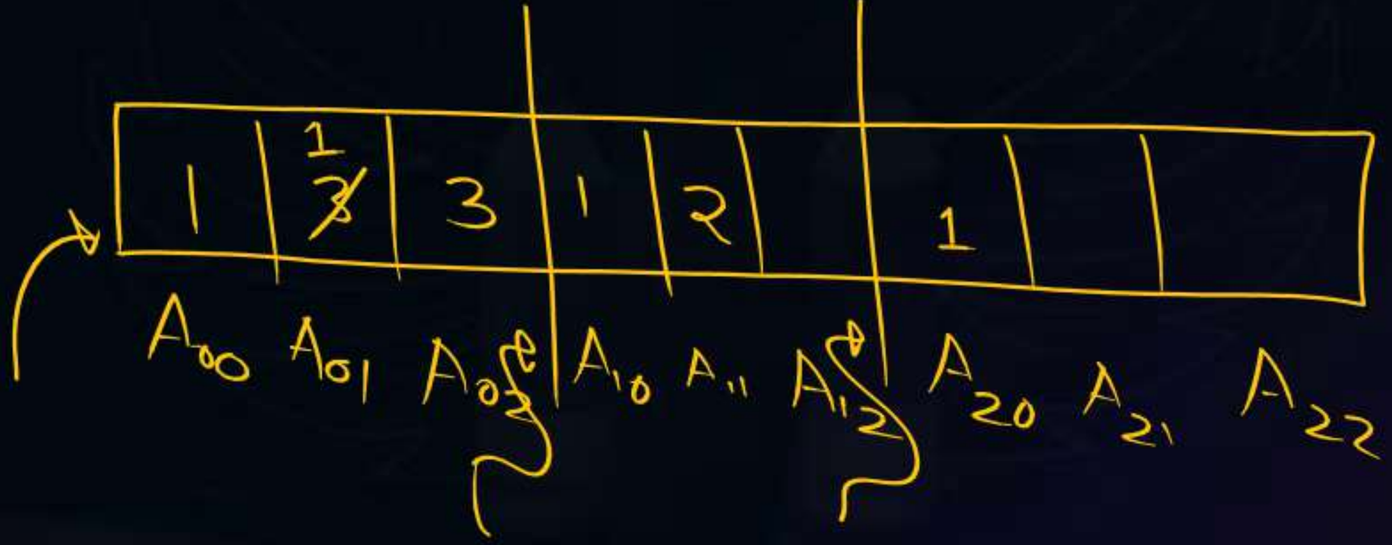
[NAT] **[1 Mark]**

i=1
j=0,1,2

i=2
j=0,1,2

```
#Q. int main ( ) {
        int A[3] [3];
        int*p=&A[0] [0];
        for(int i =0; i< 3; i++){
            for(int j =0, j <3, j ++){
                switch (i) {
                ~~case 0 : A[i] [j]; = i*j+3;~~
*(p+4)=2        case 1 : * (p+j*3+i) = j*i+1;
                         break
                case 2 : * (p+j*3+i) = j*i+1;
                         break
                }
            }
        }
        printf("%d", A[2][1]–A[2][0]);
}
```

The output give 'C' code is______.



[NAT] [1 Mark]

```
#Q. int main ( ) {
        int A[3] [3];
        int*p=&A[0] [0];
        for(int i =0; i< 3; i++){
            for(int j =0, j <3, j ++){
                switch (i) {
                ~~case 0 : A[i] [j]; = i*j+3;~~
                case 1 : * (p+j*3+i) = j*i+1;
                        break
                case 2 : * (p+j*3+i) = j*i+1;
                        break
                }
            }
        }
        printf(“%d”, A[2][1]–A[2][0]);
}
```

The output give ‘C’ code is_____.

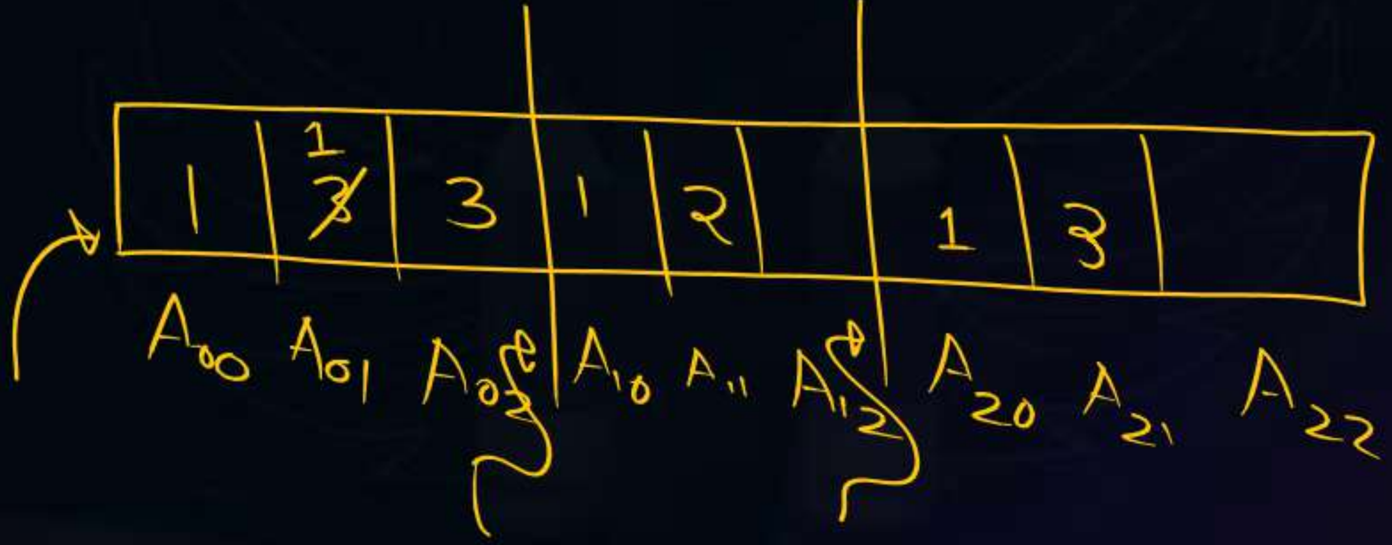
i=1 j=0,1,2

i=2 j=0,1,2

2

$*(p+6+1) = 3$

$*(p+7)=3$

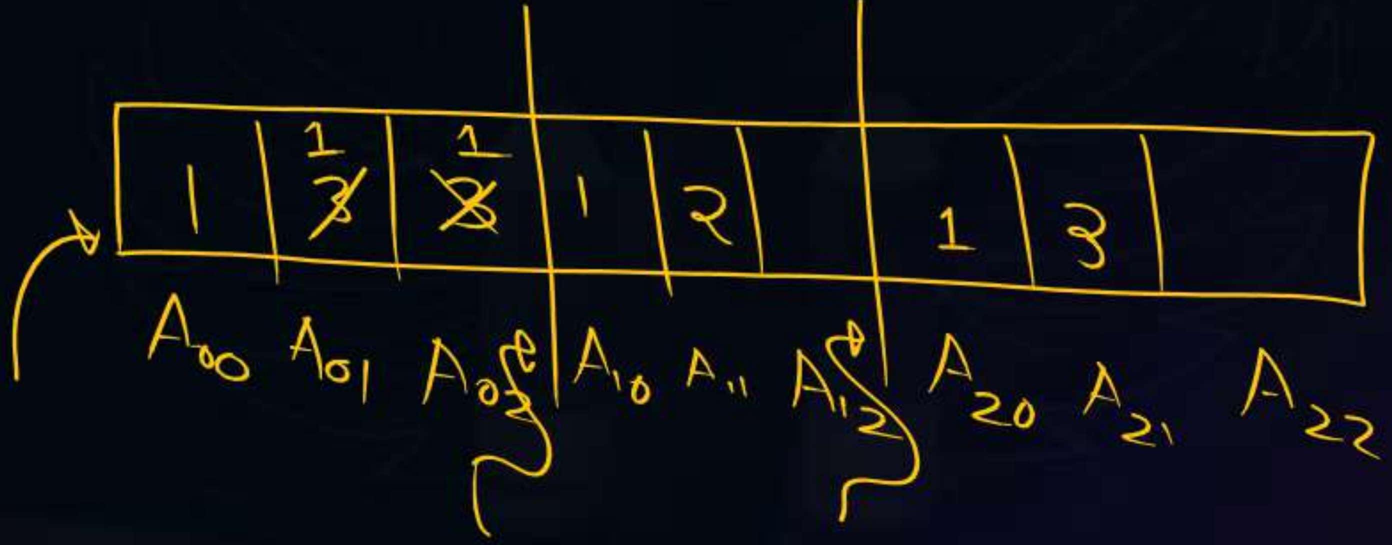


[NAT] [1 Mark]

i=1
~~j=0,1,2~~
i=2
j=0,1,2

```
#Q. int main ( ) {
        int A[3] [3];
        int*p=&A[0] [0];
        for(int i =0; i< 3; i++){
            for(int j =0, j <3, j ++){
                switch (i) {
                ~~case 0 :  A[i] [j]; = i*j+3;~~
*(P+2) = 1      ~~case 1 : * (p+j*3+i) = j*i+1;~~
                    ~~break~~
                case 2 : * (p+j*3+i) = j*i+1;
                    break
                }
            }
        }
        printf("%d", A[2][1]–A[2][0]);
    }
```

The output give 'C' code is_____.



[NAT]

[1 Mark]

i=1 j=0,1,2

i=2 j=0,1,2

```
#Q. int main ( ) {
        int A[3] [3];
        int*p=&A[0] [0];
        for(int i =0; i< 3; i++){
            for(int j =0, j <3, j ++){
                switch (i) {
                ~~case 0 :  A[i] [j]; = i*j+3;~~
                ~~case 1 : * (p+j*3+i) = j*i+1;~~
                            ~~break~~
                case 2 : * (p+j*3+i) = j*i+1;
                            break
                          }
            }
    }
    printf("%d", A[2][1]–A[2][0]);
}
```

i=2 j=1

*(P+3+2)=3

The output give 'C' code is_____.

A_{00}	A_{01}	A_{02}	A_{10}	A_{11}	A_{12}	A_{20}	A_{21}	A_{22}
1	1 ~~3~~	1 ~~3~~	1	2	3	1	3	

[NAT] [1 Mark]

#Q.

```
int main ( ) {
    int A[3] [3];
    int*p=&A[0] [0];
    for(int i =0; i< 3; i++){
        for(int j =0, j <3, j ++){
            switch (i) {
            case 0 :  A[i] [j]; = i*j+3;
            case 1 : * (p+j*3+i) = j*i+1;
                    break
            case 2 : * (p+j*3+i) = j*i+1;
                    break
            }
        }
    }
    printf("%d", A[2][1]-A[2][0]);
}
```

i=1 j=0,1,2

i=2 j=0,1,2

i=2 j=2

*(P+6+2) = 5

3 - 1 ⇒ 2

The output give 'C' code is 2.

1	~~3~~ 1	~~3~~ 1	1	2	3	1	3	5
A_{00}	A_{01}	A_{02}	A_{10}	A_{11}	A_{12}	A_{20}	A_{21}	A_{22}

[MCQ] [1 Mark]

#Q. What do the following declaration signify:

```
int(*(*f[5])()[9];
```

[✓ A] f is an array of 5 pointer to function returning pointer to array of 9 integers.

B F is a pointer of 9 integer array which pointer to 5 function of return type integer.

C f is a pointer to an array of 5 functions returning an array of 9 integers.

D F is a pointer to array of 5 elements which return an array of 9 class.

[MCQ] [1 Mark]

#Q. Which of the following declarations satisfy the explanation.

x is a pointer to a function that takes 2 arguments first is an array of 5 pointer to char and second argument is a character and function return pointer to float.

A float (*x) (char l [5],) char m);

B float **x(char l [5], char *m); x ⇒ function

[✓ C] float* (*x) (char* l [5], char m);

D float*(x)* (char * l[5], chart* m);

1 min

[NAT]

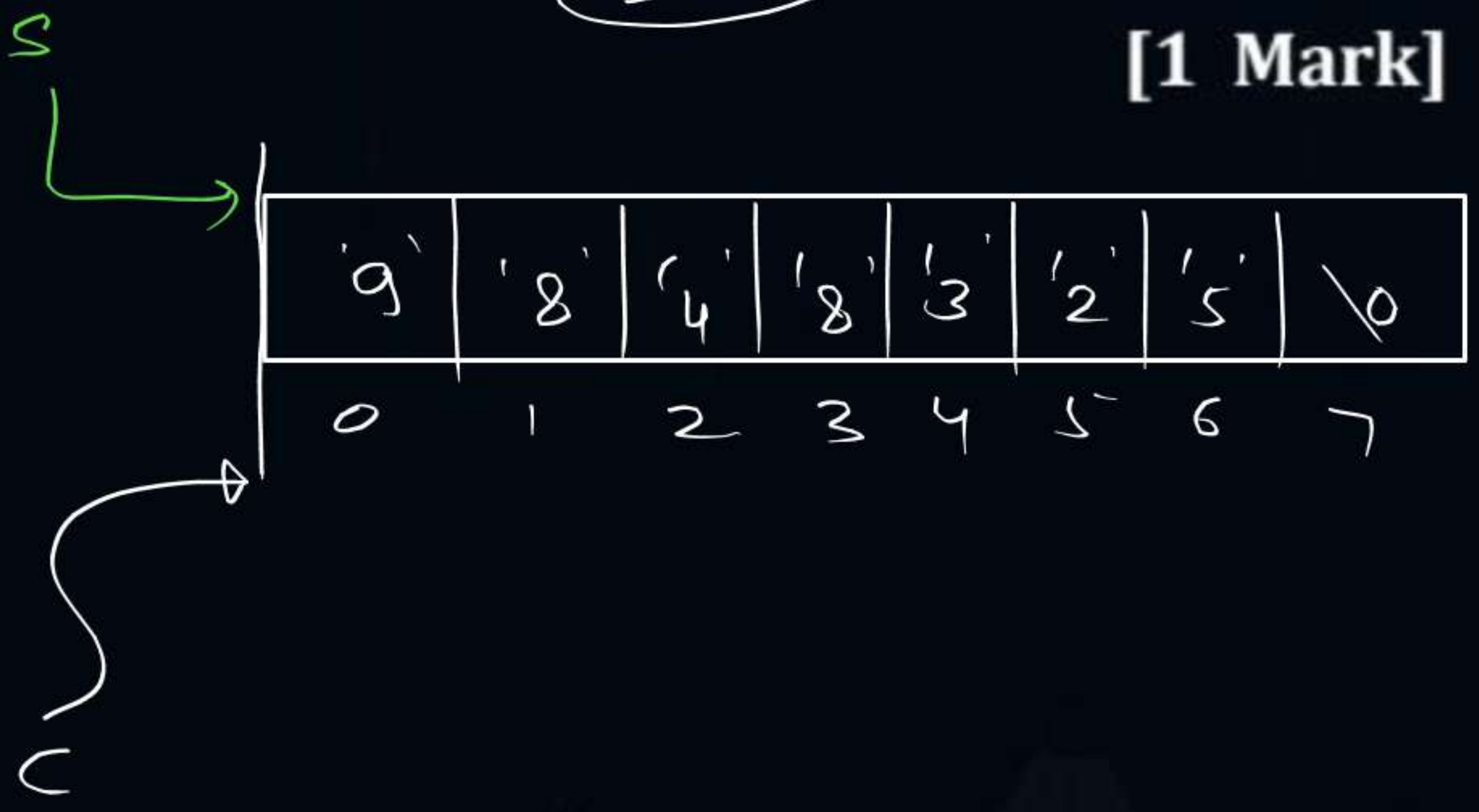
[1 Mark]

#Q. char s [] = "9848325";

char * c = s;

printf("%c", *(c+c[3] - 6[c]));

The output of snippet is 8.



'9'	'8'	'4'	'8'	'3'	'2'	'5'	\0
0	1	2	3	4	5	6	7

c[6]

c[3]   6[c]

'8' − '5'

c[3] − 6[c] ⇒ 3

c + c[3] − 6[c] ⇒ c + 3

*(c + c[3] − 6[c]) ⇒ *(c+3) ⇒ c[3]

[NAT] [1 Mark]

2

#Q. Consider a 'H' hash table of size 10 and the hash function $h(x) = x^2 \% 10$ is used, if collision occurs the elements will be replaced. If there are 1000 elements then the {(load on H[2])+(load on H[3])} is ___0___.

$0 + 0 = 0$

bucket no. 2

$$\begin{tikzcd} & 2 \\ x^2 \arrow[ur] \arrow[dr] & \\ & 3 \end{tikzcd}$$

[MCQ]

8 10 13

[1 Mark]

#Q. The following nodes are inserted into an AVL tree:

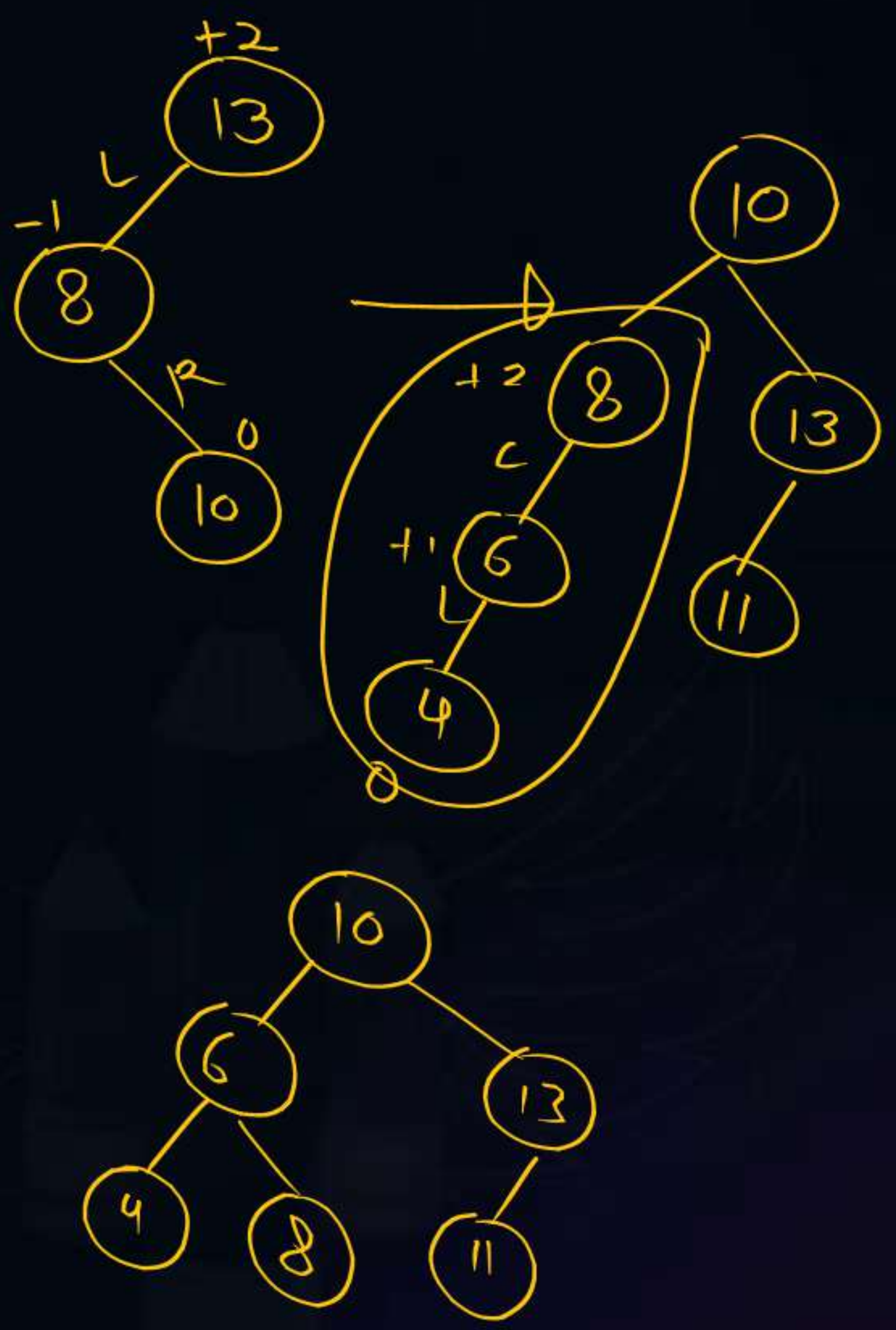
13, 8, 10, 6, 11, 4, then how many rotations did it take.

A 1 single rotation

B 2 double rotations

[✓ **C**] one double rotation, one single rotation

D 2 high rotations

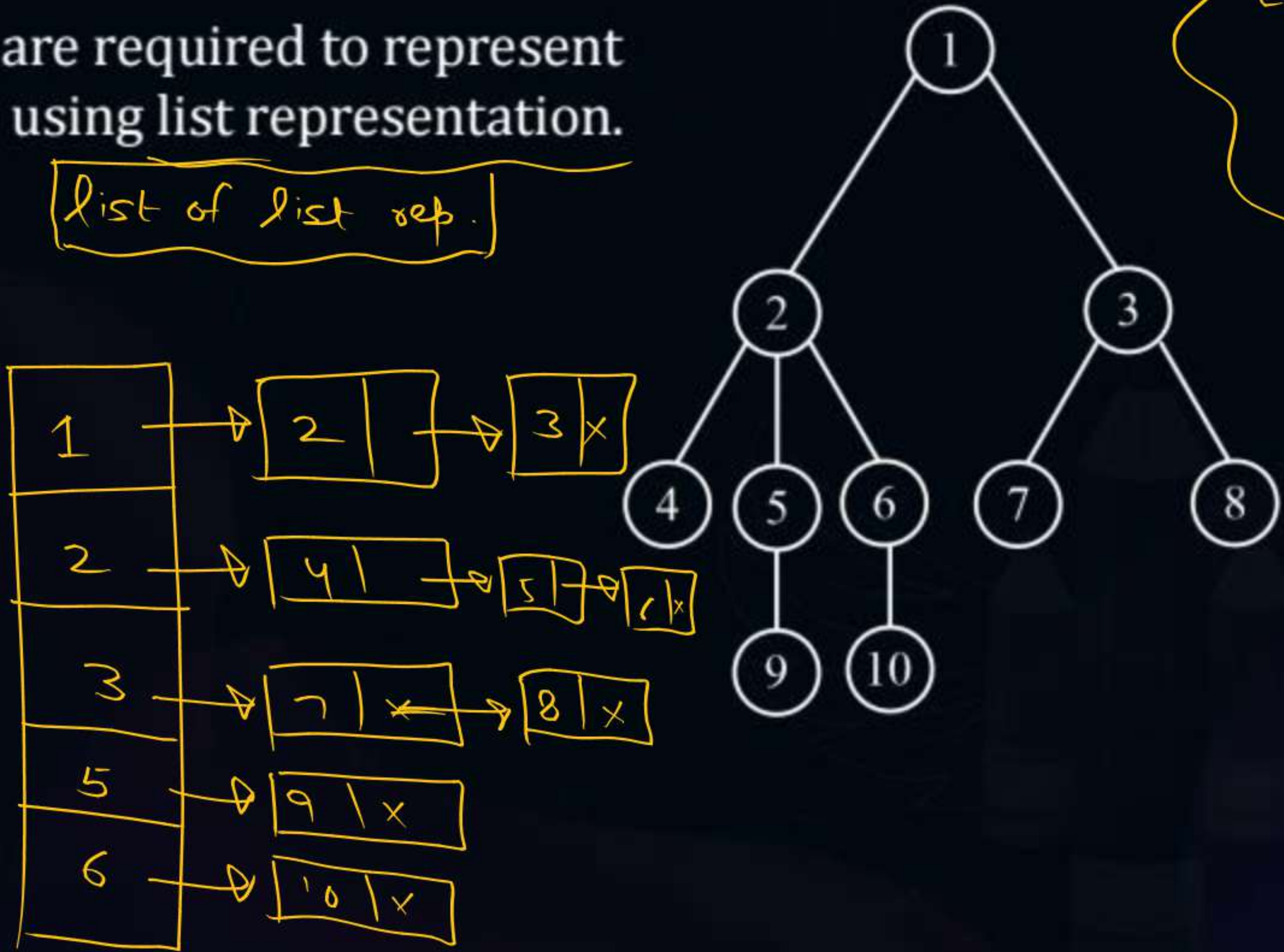


[MCQ]

[1 Mark]

#Q. How many nodes are required to represent the following tree using list representation.

Left child Right Sibling representation



A 14

B 15

C 12

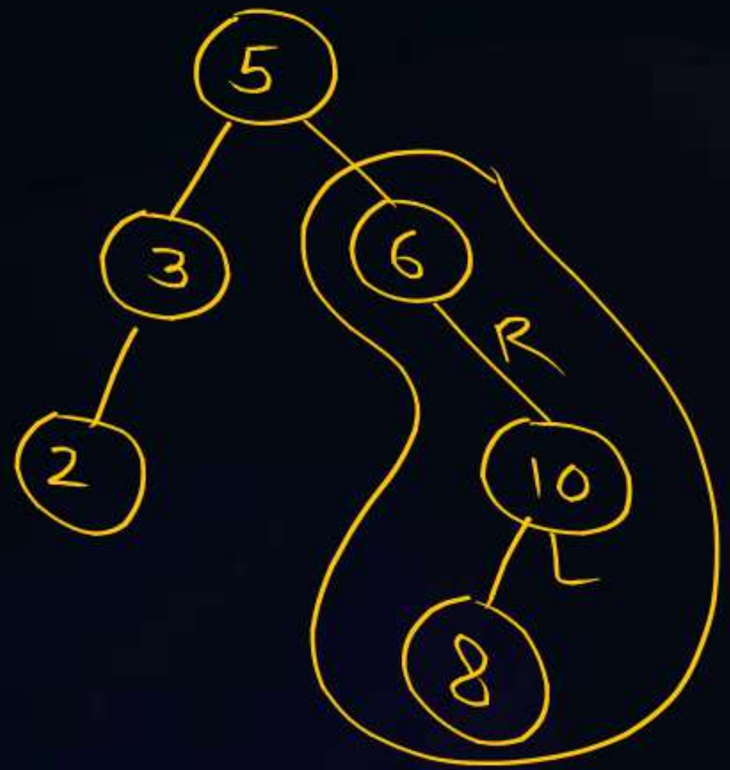
D 11

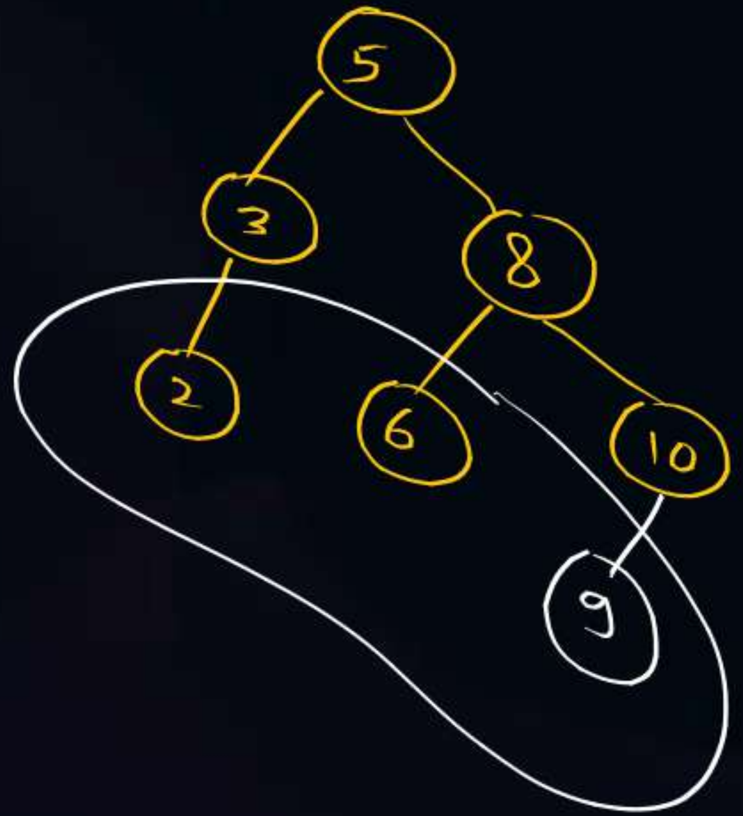
[NAT]

[1 Mark]

#Q. If the following elements are inserted into an empty AVL tree 5, 6, 3, 2, 10, 8, 9 then the sum of leaf nodes is______.

6 8 10





17

[MCQ]

[1 Mark]

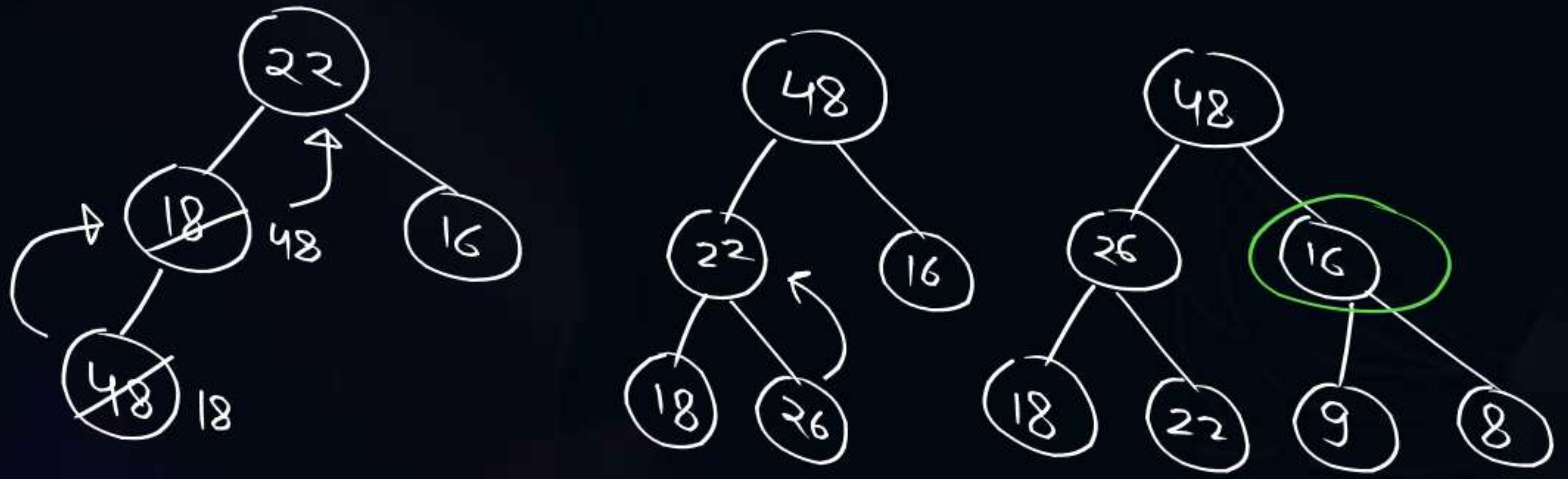
#Q. Construct max heap by inserting following elements 22, 18, 16, 48, 26, 9, 8 Note that after every insertion the heap should be heapified then what is the element which is right child of the root.

A 48

[✓ B] 16

C 9

D 26



[MCQ] [1 Mark]

#Q. A 4-Ary tree where every internal nodes has exactly 4 children then number of internal nodes are there if there are 21 nodes in total.

$$21 = 4 \times I + \overset{\text{Root}}{1}$$

$$20 = 4 \times I$$

$$\boxed{I = 5}$$

A 2

B ~~0~~ 5

C 1

D 3

[MSQ]

[1 Mark]

#Q. Which of the following hash function are best expected to have less number of collisions

$\boxed{n = 10}$

[✓ A] x mod n → uniformly

B x^2 mod n →

✓ ✓ ✗ ✗
2, 3, 7, 8

C 2x mod n →

✓ ✓ ✓ ✓ ✗
1, 3, 6, 7, 9

[✓ D] 3x mod n

3, 6, 9, 2, 5, 8, 1, 4, 7, 0

[NAT]

$(3+1^2) \bmod 10 = \boxed{4}$ coll.

$(3+2^2) \bmod 10 = (3+4) \bmod 10 = \boxed{7}$

[1 Mark]

#Q. Consider a hash table which stores string, hash table size is 10 and hash function h(x) = x%10 where x is XOR of all characters in the string.

Consider 2, 3, 4, 6, 8 places are filled in the hash table if quadratic probing is used then at what index 'ab' will be stored $\boxed{7}$.

'a' – 97
'b' – 98
→ XOR

$x \to 3$

$h(3) = 3 \% 10 \Rightarrow \boxed{3} \to$ coll.

97 → 64 + 32 + 1

98 → 64 + 32 + 2

$$\begin{array}{r} 01100001 \\ 01100010 \\ \hline 00000011 \end{array} \Rightarrow \boxed{3}$$

XOR

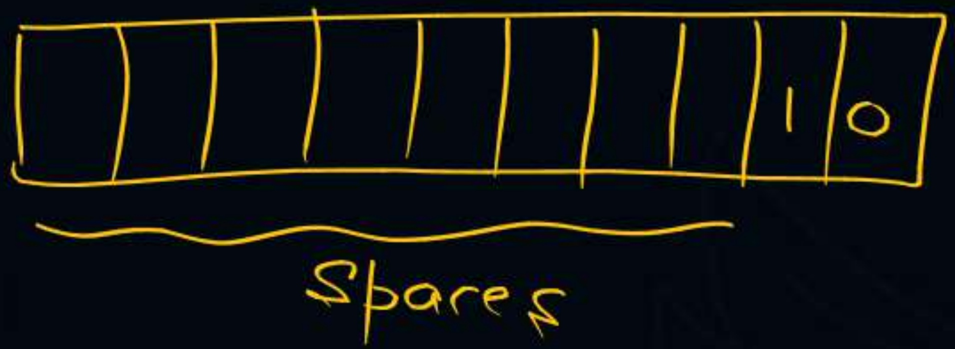
[MCQ] [1 Mark]

"%5d", x

"%10d", x

#Q. What is the output of the program

```
int main () {
        int x = 10;
        printf ("% *d", x, x);
        return 0;
}
```



A Runtime error

B Compilation error

[✓ C] ....10 ('...' is space)

D ....10 ('...' is null character)

[MCQ] [1 Mark]

return

#Q.
```
void fun (int x) {
        int i;
        if (x = =1) printf ("%d", x/2);
        i = 0;
        l :printf("%d", x –i);
        i + = 2
        if (i < x) goto l;
        fun (x/2);
}
```

What is the output of fun(5)

A infinite loop

B 5 3 1 2 0

[✓ C] stack overflow

D 5 3 2 1 0

[MCQ]

[1 Mark]

for loop based Question + cyclic Property

#Q.

```
main ( ) {
int x = 25;
do {
        printf ("%d", x)
        x - = 2;
        }
        while (x);
}
```

odd

25 → 23 → 21 → ... 1 → -1

0

What is the output of the program.

A 25

B None

[✓ C] infinite loop

D 25, 23, ...,1

[MCQ]

[1 Mark]

#Q.
```
main ( ) {
    int * p = (int *) 0 [redacted] ;
    if (P = =  (int *)0) {
            int a = 20;
              p = &a;
          printf (“%d”, *p)
          }
}
```

What is the output of the program.

A 20

B Garbage value

C Compile time error

D Segmentation fault.

2 mins Summary

Topic One -

Topic Two -

Topic Three

Topic Four

Topic Five

THANK - YOU

10 AM -12:00

Revise

2 Hrs

- linked list
- flow control statement
- Operators

⇒