GATE CS Topic wise Questions

Programming and Data Structure

YEAR 2001

Question. 1

What is printed by the print statements in the program P1 assuming call by reference parameter passing ?

```
Program P1( )
{
        1

         1(      )
      r    x;
      r    y;
}
func1(x,y,z)
{
     y=y+4
     z=x+y+z;
}
```

(A) 10, 3

(B) 31, 3

(C) 27, 7

(D) None of the above

SOLUTION

Since the function `fun 1` doesn't return the values of x & y and x & y are not passed by reference. So in program `P1( )` would print $x=10$ & $y=3$. So 10, 3

Hence (A) is correct option.

Question. 2

Consider the following three functions :

```
[P1]            int *g(void)
          {
                intx=10;
                return (& x);
          }
[P2]            int *g(void)
          {
                int *px;
                *px=10;
                return px;
          }
[P3]            int *g(void)
          {
                int *px
                px=(int*)malloc (size of (int));
                *px=10;
                return px;
          }
```

Which of the above three functions are likely to cause problems with pointers ?

(A) Only P3
(B) Only P1 and P3
(C) Only P1 and P2
(D) P1, P2 and P3

SOLUTION

P1 : Here the function is returning address of the variable x (& x) but the return type is pointer to integer not address. So incorrect.

P2 : $*px = 0$ directly assigned a value but still px doesn't point to any memory location, so memory initialization or allocation should be done before. So incorrect.

P3: Correction made in P2, memory pre allocated, So correct.

Hence (C) is correct option.

Question. 3

Consider the following program

```
Program P2
```

```
Var n:int:
procedure W (var x:int)
begin
     X=X+1
     Print x;
end
Procedure D
Begin
          var n:int;
          n=3;
          W(n);
     End
Begin     \\begin P2
     n=10;
     D;
end
```

If the language has dynamic scooping and parameters are passed by reference, what will be printed by the program ?

(A) 10

(B) 11

(C) 3

(D) None of the above

SOLUTION

$n = 10$ given but not passed to D. In D, $n = 3$ & $W(n)$ increments by 1. So $n = n + 1 = 4$.
Hence (D) is correct option.

YEAR 2002

Question. 4

The results returned by function under value-result and reference parameter passing conventions

(A) Do not differ

(B) Differ in the presence of loops

(C) Differ in all cases

(D) May differ in the presence of exception

SOLUTION

The results returned by function under value & reference parameter passing may differ in presence of loops.
Hence (B) is correct option.

Question. 5

Consider the following declaration of a two-dimensional array in C :

```
Char a[100][100]
```

Assuming that the main memory is byte-addressable and that array is stored starting form memory address 0, the address of a [40] [50] is

(A) 4040 (B) 4050

(C) 5040 (D) 5050

SOLUTION

Char $a[100][100]$
1 char require 1 byte
Total required 10000 bytes.
Memory format is byte addressable

$$\begin{array}{ccccc} a[0][0] & \cdots & a[0][50] & \cdots & a[0][99] \\ \vdots & \cdots & \cdots & \cdots & \vdots \\ a[50][0] & a[40][50] & \cdots & \cdots & \vdots \\ \vdots & \cdots & \cdots & \cdots & \vdots \\ a[99][0] & \cdots & \cdots & \cdots & a[99][99] \end{array}$$

100 bytes per row. I.e $40 \times 100 = 4000$
1 byte per column I. e. $50 \times 1 = 50$
Total 4050
Hence (B) is correct option.

YEAR 2003

Question. 6

Consider the following C function.

```
float f(float x, int y){
     float p, s; int i;
     for (s=1, p=1, i=1, i<y; i++)
```

```
    {
          p*=x/i;
          s+=p;
    }
    return s;
}
```

For large values of `y`, the return value of the function `f` best approximates

(A) x^y (B) e^x

(C) $\ln(1+x)$ (D) x^x

SOLUTION

The function is rewritten as

```
    s = ;p = ;
    for i = ;i y;i++
    {    p = p * x/i;
         s = s + p;
    }
```

Here initial value of `s` increments every time with a factor of `p * x/i`

Initially `s =` `p =`

Loop counter (i)	P	S
1	x	$1+x$
2	$x * x/2 \cdot 1 = x^2/2 \cdot 1$	$1+x+x^2/2$
3	$x^2/2 * x/3 = x^3/3 \cdot 2 \cdot 1$	$1+x+x^2/2!+x^3/3!$
4	$x^3/3! \times x/4 = x^4/4!$	$1+x+x^2/2!+x^3/3!+x^4/4!$

Thus it can be checked for every value.

Here the assumption is that the value of y is very large so $y \to \infty$

So the series $1+x+x^2/2!+x^3/3!................\infty$ will have infinite terms & from our previous knowledge we know that this ∞ series is expansion of e^x (exponential series) so.

$$1+x+x^2/2!+x^3/3!...........\infty = e^x$$

Hence (B) is correct option.

Question. 7

Assume the following C variable declaration

`int*A[10], B[10][10];`

Of the following expressions

(1) A[2]

(2) A[2][3]

(3) B[1]

(4) B[2][3]

Which will not give compile-time errors if used as left hand sides of assignment statements in a C program ?

(A) 1, 2, and 4, only (B) 2, 3, and 4, only

(C) 2 and 4 only (D) 4 only

SOLUTION

We have `int *` which is an array of 10 integer value pointer whereas $B[10][10]$ is an array which stores $10 \times 10 = 100$ integers

So let us try to solve it eliminating way.

→ Option 3 `B[1]` can't be at the left side since it is 2D array so can't use single index. We need not necessarily specify the size of first dimension for `B[][3]`

→ Option 4 B[2][3] is assignment to the array B value so possible.

→ Option 1 A [2] is also possible to assign some address os integer value

→ Option 2 this is some what tricky. Here `A[2][3]` becomes a 2D array if integer where A [2] means the 2^{nd} integer in this array and `A[2][3]`o means 3rd integer in this row. eg. $A[2][3] = 5$ means that at second row the third integer value is 5.

Hence (*) Is correct option.

Question. 8

Let $T(n)$ be the number of different binary search trees on n distinct elements.

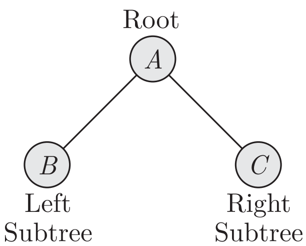
Then $T[n] = \sum_{k=1}^{n} \mathrm{T}(k-1)\,\mathrm{T}(x)$, where x is

(A) $n-k+1$ (B) $n-k$

(C) $n-k-1$ (D) $n-k-2$

SOLUTION

Binary search tree has a root node & its 2 subtrees. So for every node other than the leaves, all the elements smaller than the node are its left subtree & all the nodes which have value equal to or greater than that node are at right subtree.



Here the given expression.

$$T(n) = \sum_{K=1}^{n} T(k-1)\, T(X)$$

Figure

$$n(B) = \text{no. of nodes in left subtree}$$
$$n(C) \rightarrow \text{no. of nodes in right subtree}$$
$$T(n) = n(B) + n(C) + 1$$
$$T(n) = \sum_{K=1}^{n} T(X)\, T(k-1)$$

Expanding for $T(k-1)$ we get

$$T(n) = \sum_{K=1}^{n} T(X) \cdot \underbrace{[T(0) + T(1) + T(2) \ldots\ldots T(n-1)]}$$

no. of nodes in left subtree denoted by K

Total nodes $= n$

So remaining node $n-(k-1)$ i.e nodes in the right subtree.

So $= n - k + 1$

So overall we can say that the no. of different BST's on n different elements.

$$T(n) = \sum_{n=1}^{k} T(n-k+1)\, T(k-1)$$

Hence ( ) is correct option.

Question. 9

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 inorder transversal sequence of the resultant tree ?

(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

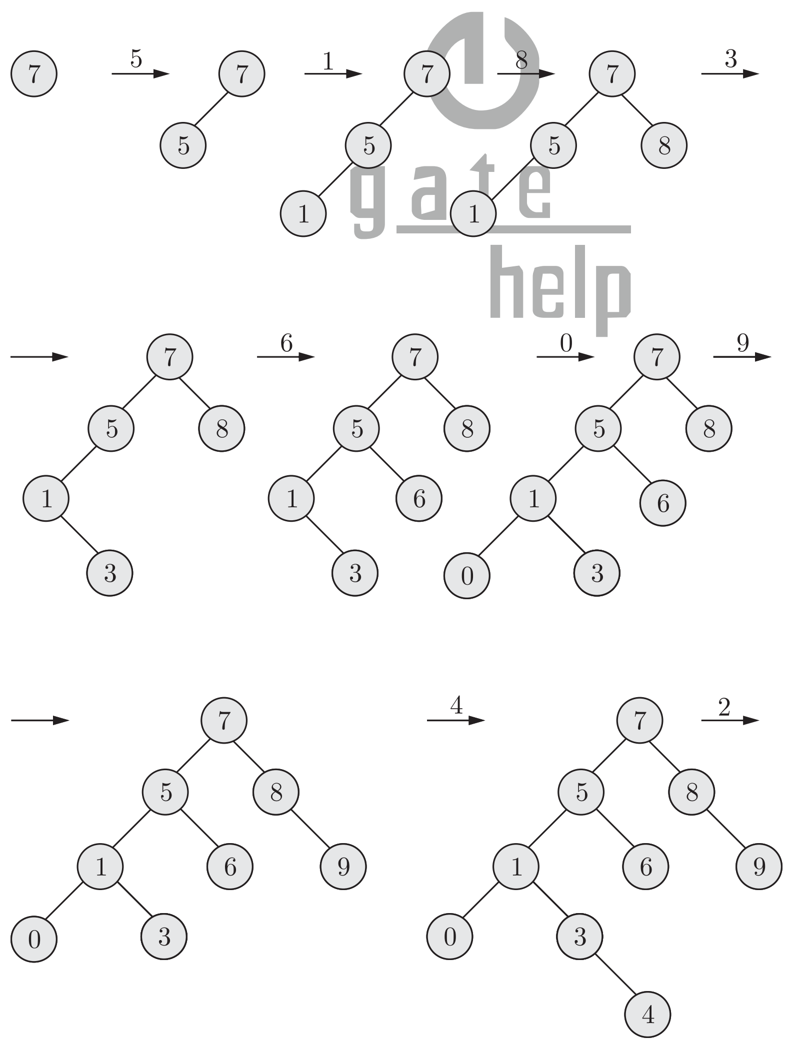
(D) 9 8 6 4 2 3 0 1 5 7

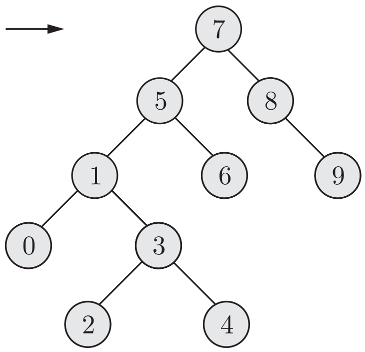
SOLUTION

We can solve it in shortcut that the first given element in 7, so we need to choose that particular option in which 7 is at the right place i.e. all the elements on its left should be smaller than it & all the elements on the right should be equal & greater than it.

So this rule is followed in option C only.

The method to make BST for given inputs is 7, 5, 1, 8, 3, 6, 0, 9, 4, 2.





To make in order of a binary search tree.

(i) Start with the root node.

(ii) Scan its left subtree,

(iii) If the node in subtree has any left child then store the node in stack & repeat this step for its left child unit no. left child of any node.

(iv) If leaf reached then print the node & pop the stack, print the poped value.

(v) Check its right subtree & repeat step (III) for it.

(vi) When stack empty then stop

So here inorder is 0 1 2 3 4 5 6 7 8 9. Actually a fact can be remembered that inorder traversal of a BST leads to a sorted sequence of elements. Hence (C) is correct option

Question. 10

A data structure is required for storing a set of integers such that each of the following operations can be done is $(\log n)$ time, where n is the number of elements in the set.

1. Delection of the smallest element.
2. Insertion of an element if it is not already present in the set.

Which of the following data structures can be used for this purpose ?

(A) A heap can be used but not a balanced binary search tree

(B) A balanced binary search tree can be used but not a heap

(C) Both balanced binary search tree and heap can be used

(D) Neither balanced binary search tree nor heap can be used

SOLUTION

Both the tasks can be performed by both the data structures but heap

is a data structure where to perform these function every element has to be checked so $O(n)$ complexity.
But the balance binary search tree is efficient data structure since at every decision it selects one of its subtree to no. of elements to be checked are reduced by a factor of 1/2 every time.

$$\frac{n}{2!} = x$$

$$x = \log n$$

Hence (B) is correct option.

Question. 11

Let S be a stack of size $n \geq 1$. Starting with the empty stack, suppose we push the first n natural numbers in sequence, and then perform n pop operations. Assume that Push and Pop operation take X seconds each , and Y seconds elapse between the end of the one such stack operation and the start of the next operation. For $m \geq 1$, define the stack-life of mcs the time elapsed from the end or Push (m) to the start of the pop operation that removes m from S. The average stack-life of an element of this stack is

(A) $n(X+Y)$ (B) $3Y+2X$

(C) $n(X+Y)-X$ (D) $Y+2X$

SOLUTION

Here each of PURSH & POP operation take X seconds & Y seconds are elapsed between two consecutive stack operations.

m is the life time of element in stack.

So m X is time for push.

m X is time for pop.

m Y is time for intermediate

So total $m(2X+Y)$

$$\text{Average stack life} = \frac{m(2X+Y)}{m}$$

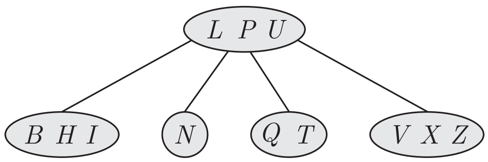
$$= 2X+Y$$

$$= Y+2X$$

Hence (D) is correct option.

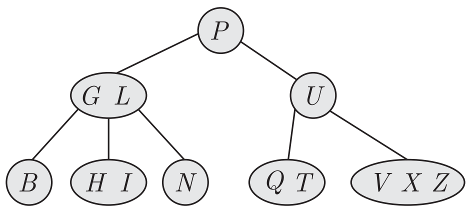
Question. 12

Consider the following 2-3-4 tree (i.e., B-tree with a minimum degree of two) in which each data item is a letter. The usual alphabetical ordering of letters is used in constructing the tree

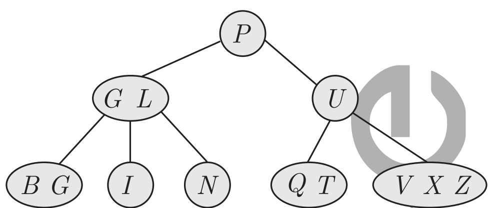


What is the result of inserting G in the above tree ?

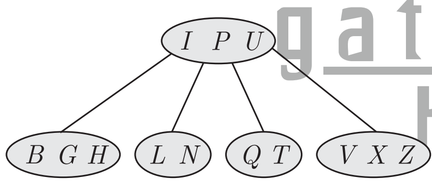
(A)



(B)



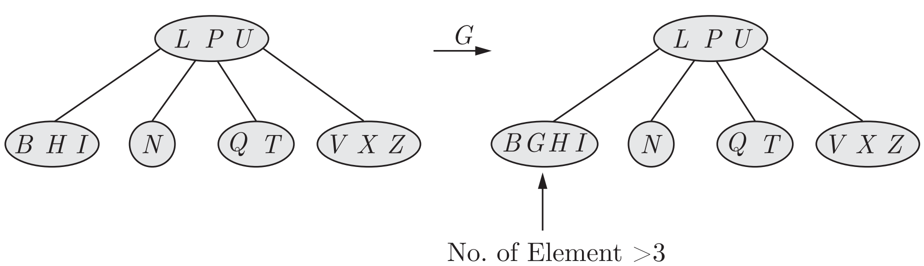
(C)



(D) None of the above

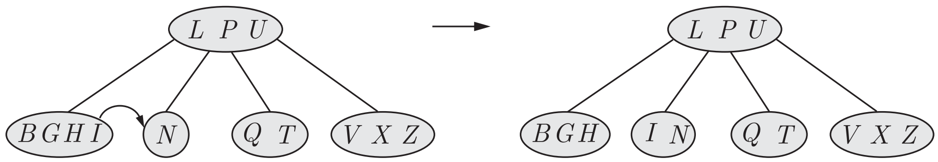
SOLUTION

2-3-4 B-tree means the min degree of a node is two & it can be max 4 So maximum of 3 elements can be there in a node.



Here in this node the no. of element >3.
So we need a split or rotation.

Since the adjacent child has no. of element $\leq \frac{n}{2} = \frac{4}{2} = 2$ so we apply a right rotation.
So here.



Hence (C) is correct option.

Question. 13

In the following C program fragment, `j`, `k`, `n` and `TwoLog_n` are integer variables, and `A` is an array of integers. The variable n is initialized to an integer ≥ 3, and `TwoLog_n` is initialized to the value of $2^*\lfloor \log_2(n) \rfloor$

```
for (k=3;k<=n;k++)
     A[k]=0;
for (k=2;k<=TwoLog_n;k++)
     for (j=k+1;j<=n;j++)
          A[j]=A[j]||(j%k);
for(j=3;j<=n;j++)
     if (!A[j])printf(“%d”,j);
```

The set of number printed by this program fragment is

(A) $\{m \mid m \leq n, (\exists\, i)\,[m = i!]\}$ (B) $\{m \mid m \leq n, (\exists\, i)\,[m = i^2]\}$

(C) $\{m \mid m \leq n, m \text{ is prime}\}$ (D) $\{m \mid m \leq n, m \text{ is odd}\}$

SOLUTION

Question. 14

Consider the C program shown below.

```
#include <stdio.h>
#define print(x)printf(“%d”,x)
int x;
void Q (int z){
     z+=x; print(z);
}
void p (int*y){
     int x=*y+2;
```

```
    Q(x);*y=x-1;
    print(x);
}
main (void){
    x=5;
    p(&x);
    print(x);
}
```

The output of this program is

(A) 12 7 6 (B) 22 12 11

(C) 14 6 6 (D) 7 6 6

SOLUTION

Figure

Here X is the global variable so still 5.

Figure

Here this is global X whose xy has been changed to 6 so 6 is printed
12 66
Hence (A) is correct option.

```
First x=5
Then by function p(&x)
              X =5+2=7
Then by function Q(x)
              z =z+x
                =7+5=12
Here x is global variable so still it is 5.
Return to function p(&x)
              Y =7-1=6
print         x =7
return to main
Print         x =6
Here this is global x whose *y ahs been changed to
6 so 6 is printed.
```

Question. 15

Consider the function – defined below.

```
struct item {
      int data;
      struct item*next;
};
int f (struct item *p){
      return ((p==NULL)||(p->next==NULL)||
            ((p->data<=p->next->data)&&
                  f(p->next)));
}
```

For a given linked list p, the function f return 1 if and only if

(A) the list is empty or has exactly one element

(B) the elements in the list are sorted in non-decreasing order of data value

(C) the elements in the list are sorted in non-increasing order of data value

(D) not all elements in the list have the same data value

SOLUTION

Here the return 1 any 1 of the following should be correct.
(A) $P == NULL$ i.e the list is empty (ends)
(B) $P \rightarrow next = NULL$ i.e have one element.
(C) $P \rightarrow data <= p \rightarrow next \rightarrow data$ i.e the element is smaller than its next element also. This is true for whole list. Since `&&f(p →next)` is also there.
So overall it gives that the elements should be in sorted order.
Hence (B) is correct option.

YEAR 2004

Question. 16

The goal of structured programming is to

(A) have well indented programs

(B) be able to infer the flow of control from the compiled code

(C) be able to infer the flow of control form the program text

(D) avoid the use of GOTO statements

SOLUTION

Structured programming :- It is way of programming using the sub structure method, i.e splitting the programs into sub sections.
Structured programming prevents confusing transfer of control of avoiding the use of GOTO statements.
Hence (D) is correct option.

Question. 17

Consider the following C function

```
void swap (int a, int b)
     {int temp;
     temp =a;
     a    =b;
     b    =temp;
     }
```

In the order to exchange the values of two variables x and y.

(A) call swap (x, y)

(B) call swap $(\&x, \&y)$

(C) swap (x, y) cannot be used as it does not return any value

(D) swap (x, y) cannot be used as the parameters are passed by value

SOLUTION

Here the function takes the arguments by value.

→ Option (A) sends parameter by value but only the local variable a & b will be exchanged but not the actual variables x & y so incorrect.

→ Option (B) is incorrect sending address of x & y.

→ Option (C) swap (x, y) is usable there is no need to return.

→ Option (D) is the opposite statement of option (A), it says that the values are passed by value so won't swap so the option is correct.

Hence (D) is correct option.

Question. 18

A single array A [1........MAXSIZE] is used to implement two stacks. The two stacks grow from opposite ends of the array. Variables top

1 and top 2 (top 1<top 2) point to the location of the topmost element in each of the stacks. If the space is to be used efficiently, the condition for "stack full" is

(A) (top 1=MAXSIZE/2) and (top 2=MAXSIZE/.2+1)

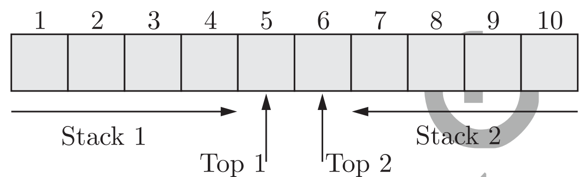
(B) top 1+top2=MAXSIZE

(C) (top 1=MAXSIZE/2) or (top2=MAXSIZE)

(D) top 1=top 2−1

SOLUTION

Let take maxsize =10



Here the stack will be fuel if both top 1 & top 2 are at the adjacent index values i.e. their difference is 1.

So top 1 = top 2 - 1

Here (D) is correct option.

Question. 19

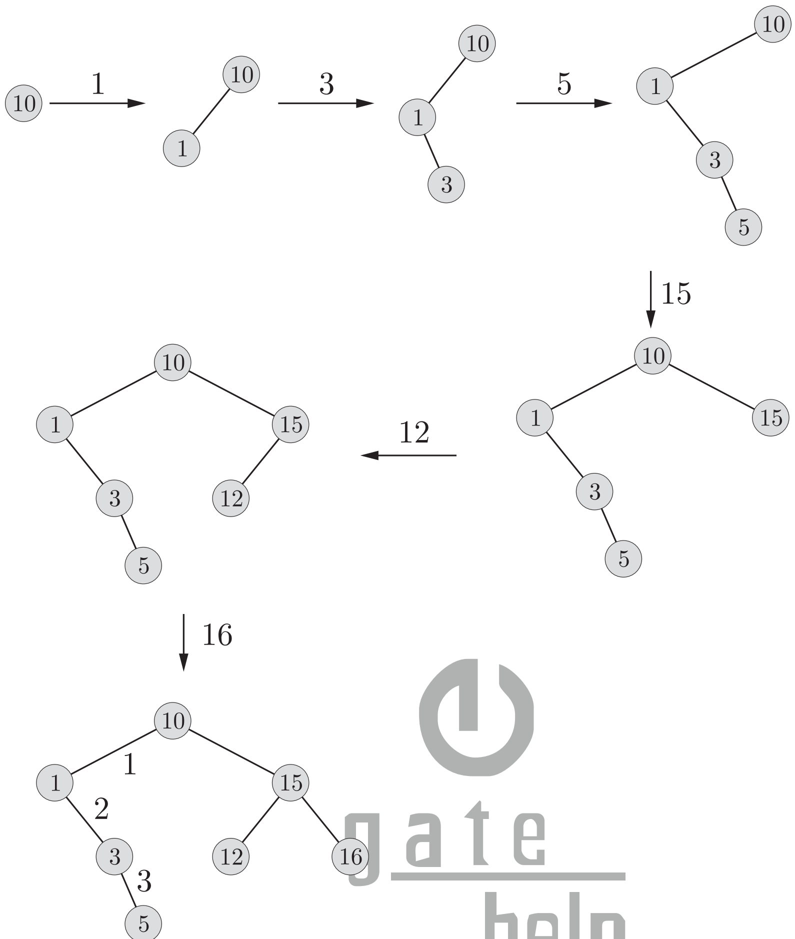
The following numbers are inserted into an empty binary search tree in the given order: 10, 1, 3, 5, 15, 12, 16. What is the height of the binary search tree (tree height is the maximum distance of a leaf node from the root) ?

(A) 2 (B) 3

(C) 4 (D) 6

SOLUTION

Given are 10, 1, 3, 5, 15, 12, 16



The height of the leaf node (5) is high 3.
Hence (B) is correct option.

Question. 20

The best data structure to check whether an arithmetic expression has balanced parenthesis is a

(A) queue (B) stack

(C) tree (D) list

SOLUTION

Balanced parenthesis in an equation are such that the no. of opening and closing parenthesis and in correct order should be there.
We can check balancing using stack. When we get any opening parenthesis then we push that in the stack & if we get a closing one then we pop the stack. After the complete scanning of input string if stack is found empty then the arithmetic expression is balanced.

Hence (B) is correct option.

Question. 21

Consider the following C function

```
int f(int n)
{static int i=1;
     if (n>=5) return n;
     n=n+i;
     i++;
     return f(n);
}
```

The value returned by `f(1)` is

(A) 5 (B) 6

(C) 7 (D) 8

SOLUTION

Here `i` is an static variable, so if it is once initialized it can't be initialized again during its scope
n is incremented by 1 & `f(n)` is called then.
The final return is when `n>=5` i.e. `n` returned then

Step	Call	n	i	
(1)	1	1	1	condition false $n < 5$
(2)		$1 + 1 = 2$	2	
(3)	2	2	2	false $n < 5$
(4)		$2 + 2 = 4$	3	
(5)	3	4	3	false $n < 5$
(6)		$4 + 3 = 7$	4	
(7)	4	7	4	true return $n = 7$

So return value is 7.
Hence (C) is correct option.

Question. 22

Consider the following program fragment for reversing the digits in a given integer to obtain a new integer. Let $n = d_1 d_2 ....d_m.$

```
int n, rev;
rev=0;
while(n>0){
     rev=rev*10+n%10;
```

```
        n=n/10;
}
```

The loop invariant condition at the end of the i^{th} iteration is

(A) $n = d_1 d_2 ...... d_{m-i}$ and rev $= d_m d_{m-1} ...... d_{m-i+1}$

(B) $n = d_{m-i+1} ..... d_{m-1} d_m$ or rev $= d_{m-i} ..... d_2 d_1$

(C) $n \neq$ rev

(D) $n = d_1 d_2 .... d_m$ or rev $= d_m ...... d_2 d_1$

SOLUTION

Here after every iteration one digit is reduced from n since $n = n/10$ so unit place is removed.
This unit place is then added into the previous reverse sum (rev) after multiplying rev by 10. So 1 digit is incremented every iteration.
So at the i^{th} iteration n should have $m - i$ digits $d_1 d_2 ..... d_{m-i}$ & rev have $d_m d_{m-1} .... d_{m-i+1}$

i	n	rev
1	$d_1 d_2 .... d_{m-1}$	d_m
2	$d_1 d_2 ...... d_{m-2}$	$d_m d_{m-1}$

So on.
Hence (A) is correct option.

Question. 23

Consider the following C program segment:

```
char p[20];
char*s= "string";
int length=strlen(s);
for  (i=0;i<length; i++)
     p[i]=s[length i];
printf("% s", p);
```

The output of the program is

(A) gnirts (B) string

(C) gnirt (D) no output is printed

SOLUTION

In line 8 `p[i]=S[length-i];`
Here p is a character pointer variable so we can't assign the value of a pointer variable into character variable so no output is printed.

The block of code shown here is actually outputs the reversal of string given in S. Which is a char type pointer. But the mistake during the loop execution is done. The statement is accessing the `s[length - i]` & loop starts from 0

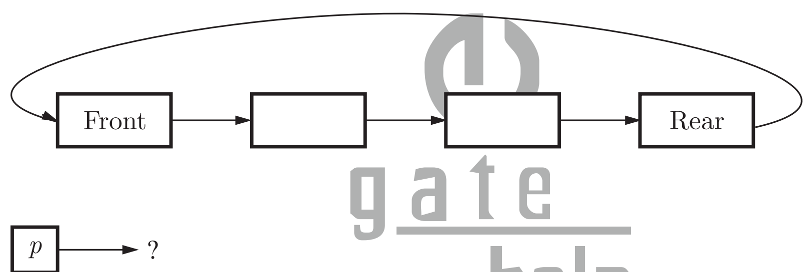
When `i = 0`, `s[length - 0]` $\Rightarrow$ `s[length]`.

So this value for string is always P will start with null pointer. So the string p will start with null pointers and nothing will be printed.

Hence (D) is correct option.

Question. 24

A circularly linked list is used to represent a Queue. A single variable p is used to access the Queue. To which node should p point such that both the operations enQueue and deQueue can be performed in constant time ?



(A) rear node

(B) front node

(C) not possible with a single pointer

(D) node next to front

SOLUTION

Here due to circular connection the rear & front are connected. Here if we point P to rear the $P \rightarrow$next point to front node & $P \rightarrow$ data will point to rear value while inserting at rear following sequence of operations done.

$$\begin{cases} \text{P} \rightarrow \text{data} = \text{inserted value} \\ \text{P} \rightarrow \text{next} = \text{P} \rightarrow \text{next} \rightarrow \text{next} \end{cases}$$

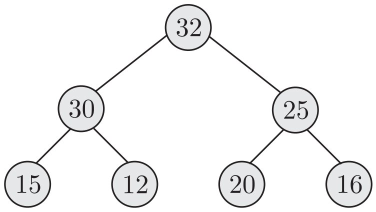
These operation done is $0(1)$ time

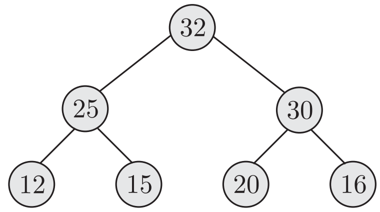
So constant complexity.

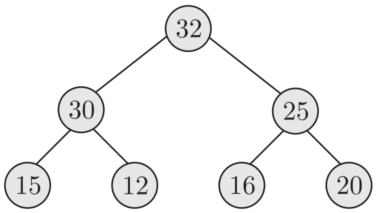
Hence (A) is correct option.

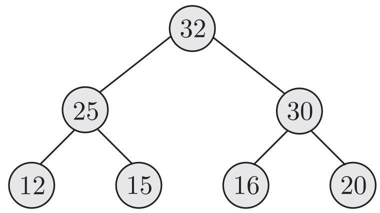
Question. 25

The elements 32, 15, 20, 30, 12, 25, 16 are inserted one by one in the given order into a maxHeap. The resultant maxHeap is

(A)


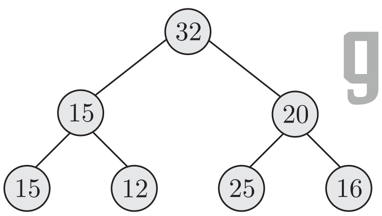
(B)


(C)


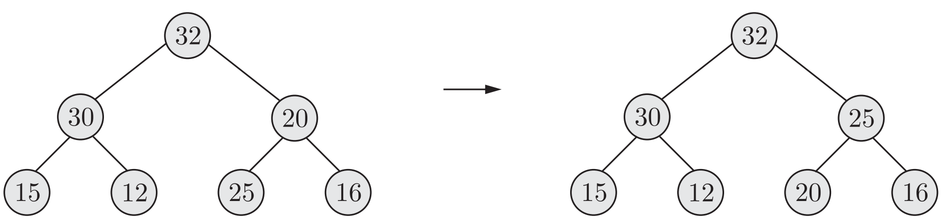
(D)


SOLUTION

Given elements are 32, 15, 20, 30, 12, 25, 16



Here $n = 7$.
This is the heap of elements.
Now for max heap property is that every root node should be larger than its child nodes. The root node on the tope is largest of all.
Step 1 take $\lceil n/2 \rceil$ node for start $\lceil 7/2 \rceil = 4$ if it is at right place i.e it is smaller than its parent & greater than its children then OK otherwise swap them.



So option (A) is correct since it satisfy max heap property.
Hence (A) is correct option.

Question. 26

Assume that the operators +, -, $\times$ are left associative and ^ is right associative .The order of precedence (from highest to lowest) is ^, $\times$, +, -. The postfix expression corresponding to the infix expression a + b $\times$ c - d ^ e ^ f is

(A) abc $\times$+def ^^−

(B) abc $\times$+de^f^

(C) ab+c$\times$d−e^f^

(D) −+a$\times$bc^^def

SOLUTION

Given expression $a + b * c - d \wedge e \wedge f$
parenthesizing the expression as per given rules.

$$= ((a + (b * c)) - (d \wedge (e \wedge f)))$$
$$= ((a + (bc *)) - (d \wedge (ef \wedge)))$$
$$= ((abc * +) - (def \wedge \wedge))$$
$$= (abc * + def \wedge \wedge -)$$

So option (A) is correct
You can also solve using stack.
Hence (A) is correct option.

Question. 27

Consider the following C program

```
main ()
{     int x, y, m, n;
      scanf("%d%d", &x,&y);
      /*Assume x>0 and y>0*/
      m=x;       n=y;
      while      (m!=n)
                 {     if (m>n)
                       m=m−n;
                       else
                       n=n−m;
                 }
      printf("%d",n);
}
```

The program computers

(A) $x \div y$, using repeated subtraction

(B) $x \bmod y$ using repeated subtraction

(C) the greatest common divisor of x and y

(D) the least common multiple of x only

SOLUTION

Here if $m > n$ then $m = m - n$

$m < n$ then $n = n - m$

Let take $X = 24$ $Y = 9$

Then $m = 24$ $n = 9$

iteration	m	n
1	$24 - 9 = 15$	9
2	$15 - 9 = 6$	9
3	6	$9 - 6 = 3$
4	$6 - 3 = 3$	3

Here $m = n$ so n returned

Which is GCD (Greatest common divisor) of X & Y

Hence (C) is correct option.

Question. 28

What does the following algorithm approximate ? (Assume m>1, $\in > 0$).

```
x=m;
y=1;
while (x-y>∈)
            {     x=(x+y)/2;
            y=m/x;
            }
print (x);
```

(A) log m

(B) m^2

(C) $m^{1/2}$

(D) $m^{1/3}$

SOLUTION

Here we take let $x = 16$

Loop will stop when $x - y = 0$ or >0

Iteration	X	Y
1	$\frac{(16+1)}{2} = 8$	$\frac{16}{8} = 2$

2	$\frac{8+2}{2}=5$	$\frac{16}{5}=3$
3	$\frac{5+3}{2}=4$	$\frac{16}{4}=4$

Here $X = Y$
Then take X. which is 4.

$$(m)^{1/2} = 4 = (16)^{1/2}$$

Hence (C) is correct option.

Question. 29

Consider the following C program segment

```
struct Cellnode {
     struct CellNode     *leftChild;
     int element;
     struct CellNode     *rightChild;
     }
int DoSomething (struct CellNode *ptr)
{
     int value=0;
     if(ptr!=NULL)
     {    if (ptr->leftChild !=NULL)
          value=1+DoSomething   (ptr->leftChild);
          if (ptr->rightChild!=NULL)
          value=max(value,1+DoSomething(ptr->
right child));
     return (value);
     }
```

The value returned by the function DoSomething when a pointer to the proof of a non-empty tree is passed as argument is

(A) The number of leaf nodes in the tree

(B) The number of nodes in the tree

(C) The number of internal nodes in the tree

(D) The height of the tree.

SOLUTION

Value initialized by 0
If any root node has left child then it adds 1 to the value & move to

left child & if any mode has right child also then also calculated the value using recursion & take maximum of both left & right value is taken.
So we know that height is the largest distance between root node & leaf.
So this program calculates heights.
Hence (D) is correct option.

Question. 30

Choose the best matching between the programming styles in Group 1 and their characteristics in Group 2.

Group-I	Group-2
P. Functional Q. Logic R. Object-oriented S. Imperative	1. Command-based, procedural 2. Imperative, abstract data types 3. Side-effect free, declarative, expression evaluation 4. Declarative, clausal representation, theorem proving

(A) P-2, Q-3, R-4, S-1 (B) P-4, Q-3, R-2, S-1
(C) P-3, Q-4, R-1, S-2 (D) P-3, Q-4, R-2, S-1

SOLUTION

p. Functional Programming is declarative in nature, involves expression evaluation, & side effect free.
q Logic is also declarative but involves theorem proving.
r. Object oriented is imperative statement based & have abstract (general) data types.
s Imperative :- The programs are made giving commands & follows definite procedure & sequence.

Hence (D) is correct option.

YEAR 2005

Question. 31

What does the following C-statement declare?

```
int    int
```

(A) A function that takes an integer pointer as argument and returns an integer

(B) A function that takes an integer pointer as argument and returns an integer pointer

(C) A pointer to a function that takes an integer pointer as argument an returns

(D) A function that takes an integer pointer as argument returns a function pointer

SOLUTION

Given statement `int(*f)(int*)`
This is not the declaration of any function since the `f` has * (pointer symbol) before it. So `f` is a pointer to a function also the argument type is * & the return type is
So overall we can say that `f` is a pointer to a function that takes an integer pointer as argument and returns an integer.
Hence (C) is correct option.

Question. 32

An Abstract Data type (ADT) is

(A) same as an abstract class

(B) a data type that cannot be instantiated

(C) a data type for which only the operations defined on it can be used, but none else

(D) all of the above

SOLUTION

Abstract Data type :- It is defined as a user defined data type, specified by keyword 'abstract' & defines the variables & functions, these operations can only use the variables of this data type.
So option (C) which says that Abstract data type for which only operations defined on it can be used is correct.
Eg. stack data type
Here operations defined are push & pop. So we can apply only these 2 operations on it.
Hence (C) is correct option.

Question. 33

A common property of logic programming languages and functional languages is

(A) both are procedural language

(B) both are based on λ −calculus

(C) both are declarative

(D) all of the above

SOLUTION

λ-calculus→ It provides the semantics for computation with functions so that properties of functional computation can be studied.

Both the languages require declaration before use of any object.

Both are procedural

So option (D) is correct.

Both the languages are based on λ calculus, procedural & declarative

Hence (D) is correct option.

Question. 34

Which of the following are essential features of an object-oriented programming languages?

1. Abstraction and encapsulation
2. Strictly-typedness
3. Type-safe property coupled with sub-type rule
4. Polymorphism in the presence of inheritance

(A) 1 and 2 only (B) 1 and 4 only

(C) 1, 2 and 4 only (D) 1, 3 and 4 only

SOLUTION

Object oriented programming languages necessarily have features like. Abstraction Encapsulation, inheritance with polymorphism but OOPL are also strongly-typed since there are restrictions on how operations involving values having different data types can be intermixed.

Eg. two integers can be divided but one integer & one string can't.

Hence (B) is correct option.

Question. 35

A program P reads in 500 integers in the range (0, 100) representing the scores of 500 students. It then prints the frequency of each score above 50. What be the best way for P to store the frequencies?

(A) An array of 50 numbers

(B) An array of 100 numbers

(C) An array of 500 numbers

(D) A dynamically allocated array of 550 numbers

SOLUTION

Here the no. readable are range 0 to 100 but the output of the program is interested in scores above 50 so there are 50 values (51 to 100) in this range.
So only an array 50 integers required as we get any no. we increment the value stored at index.
Array $[x-50]$ by 1.
Hence ( ) is correct option.

Question. 36

Consider the following C-program

```
void foo(int n, int sum ){
      int     ,
if (n    ) return;
   n   ;   n   ;
      sum = sum    ;
      foo ( , sum);
printf("%d," );
      }
      int main(){
int a =     , sum = 0;
      foo (a, sum);
printf(%d n ,sum);
      }
```

What does the above program print?

(A) 8, 4,0, 2, 14 (B) 8, 4, 0, 2, 0

(C) 2, 0, 4, 8, 14 (D) 2, 0, 4, 8, 0

SOLUTION

Here $K = n\%10$ takes unit digit of n $j = n/10$ reduces 1 digit from n.

Recurssion	n	Local Sum	Print		
1	2048	0	$\uparrow$ 8	204	8
2	204	8	$\uparrow$ 4	20	4
3	20	12	$\uparrow$ print 0	2	0
4	2	12	$\uparrow$ print 2	0	2
5	0	14	$\rightarrow$ return		

Here $n = 0$ so return
2 0 4 8 are printed then
Print sum. Sum = 14 but in foo function only the sum variable in main is still 0.
So 2 0 4 8 0 printed.
Hence (D) is correct option.

Question. 37

Consider the following C-program

```
double foo (double); /* Line 1*/
     int main(){
     double da db;
     // input da
     db  foo(da);
     }
double foo(double a){
     return a;
     }
```

The above code complied without any error or warning. If Line 1 is deleted, the above code will show

(A) no compile warning or error

(B) some complier-warning not leading to unitended results

(C) Some complier-warning due to type-mismatch eventually leading to unitended results

(D) Complier errors

SOLUTION

Here if line 1 which is prototype declaration of the function `foo`, in C compilation process this would give an compile warning , due to type-mismatch. Since then compiler won't know what is the return type of `foo`.
So unintended results may occur.
Hence (C) is correct option.

Question. 38

Postorder traversal of a given binary search tree, T produces the following sequence of keys

10, 9, 23, 22, 27, 25, 15, 50, 95, 60, 40, 29

Which one of the following sequences of keys can be the result of an inorder traversal of the tree T?
(A) 9, 10, 15, 22, 23, 25, 27, 29, 40, 50, 60, 95
(B) 9, 10, 15, 22, 40, 50, 60, 95, 23, 25, 27, 29
(C) 29, 15, 9, 10, 25, 22, 23, 27, 40, 60, 50, 95
(D) 95, 50, 60, 40, 27, 23, 22, 25, 10, 0, 15, 29

SOLUTION

When we are given any no elements & even any order (preorder or post order) & we need to calculate inorder, then inorder is simply sorted sequence of the elements.
Here 9, 10, 15, 22, 23, 25, 27, 29, 40, 50, 60, 95
Hence (A) is correct option.

YEAR 2006

Question. 39

An implementation of a queue Q, using two stacks S1 and S2 , is given below

```
void insert(Q,x){
push (S1,x);
}
void delete(Q, x){
if (stack-empty (S2))then
```

```
        if (stack-empty (S1))then{
        print("Q is empty");
        return;
    }
    else while (! (stack-empty)(S1)){
    x=pop(S1);
    push(S2,x);
    }
    x=pop(S2);
    }
```

Let n insert and $m(\leq n)$ delete operations be performed in an arbitrary on an empty queue Q, Let x and y be the number of push and pop operations performed respectively in the processes. Which one of the following is true for all m and n ?

(A) $n+m \leq x < 2n$ and $2m \leq n+m$

(B) $n+m \leq x < 2n$ and $2m \leq y \leq 2n$

(C) $2m \leq x < 2n$ and $2m \leq y \leq n+m$

(D) $2m \leq x < 2n$ and $2m \leq y \leq 2n$

SOLUTION

Question. 40

Consider the following C-function in which a[n] and b[n] are two sorted integer arrays and c[n+m] be another integer array.

```
    void xyz (int a[],int b[],int c[]){
        int i, j, k;
        i=j=k=0;
        while((i<n))&&(j<m)
           if (a[i]<b[j]c[k++]=a[i++];
           else c[k++]=b[j++];
        }
```

Which of the following condition (s) hold (s) after the termination of the while loop ?

```
I    j<m, k=n+j-1, and a [n-1]<b[j] if i=n
II   i<n, k=m+j-1, and b[m-1]≤a[i] if j=m
```

(A) only (I)

(B) only (II)

(C) either (I) or (II) but not both

(D) neither (I) nor (II)

SOLUTION

While loop will terminate $i \geq n$ & $j \geq m$ program is to merge a & b arrays into C.
While loop terminates after merging then either (I) or (II) should hold but not both at the same time.
(I) says $j < m$ & (II) say $j = m$ vice versa
Hence (C) is correct option.

Question. 41

Consider these two functions and two statements S1 and S2 about them.

```
int work1(int *a,int i,int j)
{
     int    a i
     a j     1
     return a i
}
```

```
int work2(int *a,int i,int j)
{
     int t1=i+2;
     int t2=a[t1];
     a[j]=t2+1
     return t2-3;
}
```

S1: The transformation from work 1 to work 2 is valid, i.e., for any program state and input arguments, work 2 will compute the same output and have the same effect on program state as work 1

S2: All the transformations applied to work 1 to get work 2 will always improve the performance (i.e. reduce CPU time) of work 2 compared to work 1

(A) S1 is false and S2 is false

(B) S1 is false and S2 is true

(C) S1 is true and S2 is false

(D) S1 is true and S2 is true

SOLUTION

During the optimizations of code phase variables are reduced to temporary variables which are used to store results of unitary

operations so later if this expression again evaluated then in optimized code it is replaced by temporary variable.
Both work 1 & work 2 produces same state & since operation $i+2$ in work 1 is performed only 1 time in work 2 due to use of t so it is also optimized.
So both are true.
Hence (D) is correct option.

Question. 42

Consider the C code to swap two integers and these five statements: the code

```
void swap(int *px,int *py){
*px=*px−*py;
*py=*px+*py;
*px=*py−*px;
}
```

S_1: will generate a compilation error
S_2: may generate a segmentation fault at runtime depending on the arguments passed
S_3: correctly implements the swap procedure for all input pointers referreing to integers stored in memory locations accessible tot he process
S_4: implements the swap procedure correctly for some but not all valid input pointers
S_5: may add or subtract integers and pointers

(A) S_1 (B) S_2 and S_3
(C) S_2 and S_4 (D) S_2 and S_5

SOLUTION

Here pointers are used without initialization also the address pointed by then may be out of segment of program, so segmentation.

→ Fault may be there so. S_2 correct.
→ Here no compiler error S_1 false.
→ Correctly done swap procedure but not all valid import pointers so S_4 also true.

S_2 & S_4 are correct.
Hence (C) is correct option.

Data for Q. 43 & 44 are given below.

A 3-ary max heap os like a binary max heap, but instead of 2 children, nodes have 3 children, A 3-ary heap can be represented by an array as follows: The root is stored in the first location, a [0], nodes in the next level, from left to right, is stored form a[1] to a[3]. The nodes from the second level of the tree from left to right are stored from a[4] location onward.

An item x can be inserted into a 3-ary heap containing n items by placing x in the location a [n] and pushing it up the tree to satisfy the heap property.

Question. 43

Which one of the following is a valid sequence of elements in an array representing 2-ary max heap ?

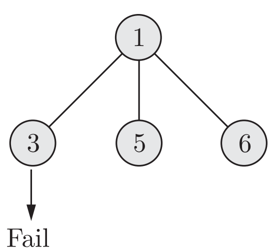
(A) 1, 3, 5, 6, 8, 9 (B) 9, 6, 3, 1, 8, 5

(C) 9, 3, 6, 8, 5, 1 (D) 9, 5, 6, 8, 3, 1

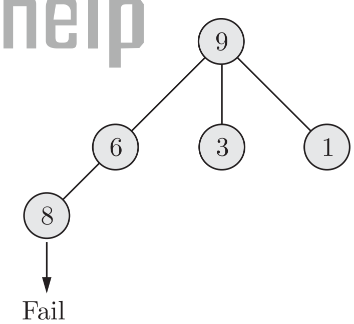
SOLUTION

Create max heap for options.

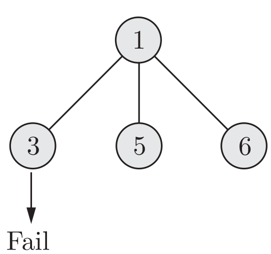
(A)



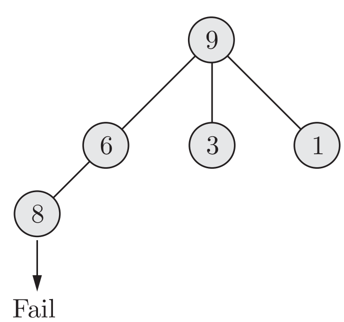
(B)



(A)



(B)



Here in option (A), (B) and (C), value present at node is not greater then all its children.

Hence (D) is correct option.

Question. 44

Suppose the elements 7, 2, 10, and 4 are inserted, in that order, into the valid 3-ary max heap found in the above question, Q. 33. Which on of the following is the sequence of items in the array representing the resultant heap ?

(A) 10, 7, 9, 8, 3, 1, 5, 2, 6, 4

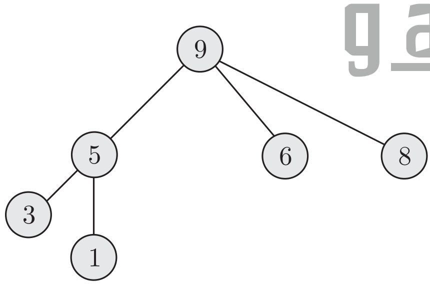
(B) 10, 9, 8, 7, 6, 5, 4, 3, 2, 1

(C) 10, 9, 4, 5, 7, 6, 8, 2, 1, 3

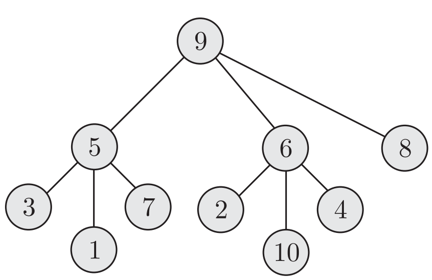
(D) 10, 8, 6, 9, 7, 2, 3, 4, 1, 5

SOLUTION

Given heap is as follows

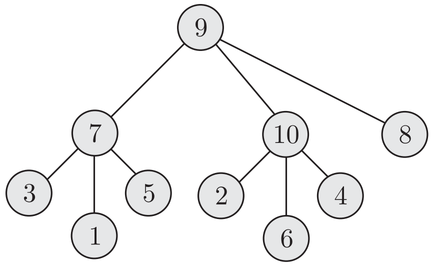


To add 7, 2, 10, 4 we add the node at the end of array

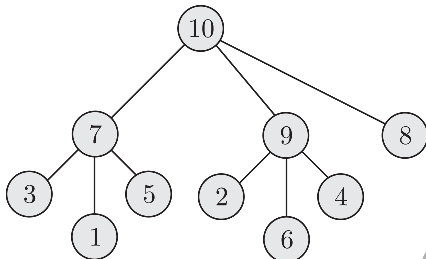


We keep if at right place in the heap tree.

Compare elements with its parent node. Since 10 > 6 and 7 >5, we interchange



Since 10 >9, we interchange and we get



$$\frac{n}{2} = \frac{10}{2} = 5$$

3 is at right position
4 is at right position
Figure Figure

Figure
Order
10 7 9 8 3 1 5 2 6 4

Hence (A) is correct option.

YEAR 2007

Question. 45

Consider the following segment of C-code

```
int, ,n

while      n
```

The number of comparisons made in the execution of the loop for any $n > 0$ is

(A) $\lceil \log_2 n \rceil + 1$ (B) n

(C) $\lceil \log_2 n \rceil$ (D) $\lfloor \log_2 n \rfloor + 1$

SOLUTION

Iteration	j	Comparison
Initially 0	$1 = 2^0$	
1	$2 = 2^1$	
2	$4 = 2^2$	
3	$8 = 2^3$	

Here condition is $2^i <$ $= n$

In i^{th} iteration & it requires $i + 1$ comparisons.

$$i = \log_2 n$$

$$i + 1 = \lceil \log_2 n \rceil + 1$$

Hence (A) is correct option.

Question. 46

The following postfix expression with single digit operands in evaluated using a stack

$\wedge$

Note that^is the exponentiation operator. The top two elements of the stack after the first* is evaluated are

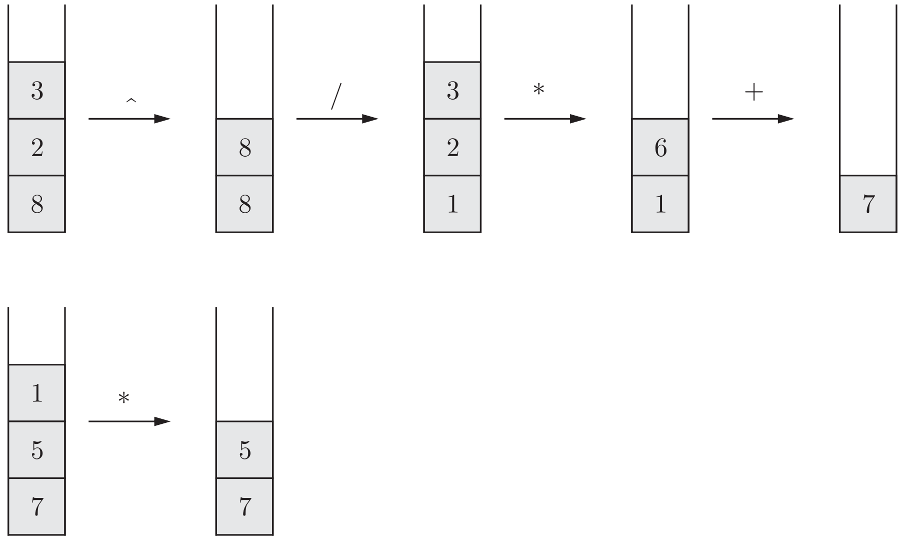
(A) 6, 1 (B) 5, 7

(C) 3, 2 (D) 1, 5

SOLUTION

Given postfix expression is $823 \wedge /23 * + 51 * -$

Scanning the string from left to right we push all the digits, & we get any operator we evaluate the operation between top 2 elements poped from stack. Then the result is pushed again into stack.



Stack contain 5, 7
Hence (B) is correct option.

Question. 47

Consider the following C function:

```
int f(int n)
    static int r=0;
if  (n  = 0) return 1;
if  (n   )
    {r = n;
    return f(n   )   ;
}
return f(n  1)  r;
}
```

What is the value of `f(5)` ?

(A) 5
(B) 7
(C) 9
(D) 18

SOLUTION

Given $f(5) = ?$

Recursion	n	r	Return	Final Return
1	5	$0 \rightarrow 5$	$f(3)+2$	$16+2=18$

Here r is a static int, so it will not be initialized again during recursive calls so the final return would be option (D) i.e. 18
Hence (D) is correct option.

Question. 48

Consider the following C program segment where Cell Node represents a node in a binary tree

```
struct CellNode {
struct CellNode*leftchild;
int element;
struct CellNode*rightchild;
};
int GetValue(struct CellNode * ptr) {
     int vlaue = 0;
if (ptr  =NULL) {
     if ((ptr   leftChild = = NULL)&&
        (ptr   rightChild = = NULL))
        Value = 1;
     else
        value = value + GetValue
                         (ptr   leftChild)
               +         Get Value
                         (ptr   rightChild);
}
return(value);
}
```

The value returned by Get Value when a pointer to the root of a binary tree is passed as its argument is

(A) the number of nodes

(B) the number of internal nodes in the tree

(C) the number of leaf nodes in the tree

(D) the height of the tree

SOLUTION

Here the structure cell Node represents a binary tree. Here the function Get value return 1 if for any node both left & right children are NULL or we can say if that node is a leaf node.
A variable value initialized to 0 is there to count these leaf nodes.
Hence (C) is correct option

YEAR 2008

Question. 49

Which combination of the integer variables x, y, and z makes the variable a get the value 4 in the following expression?

$$a = (x > y)?((x > z)?x: z):((y > z)?y: z)$$

(A) $x = 3, y = 4, z = 2$
(B) $x = 6, y = 5, z = 3$
(C) $x = 6, y = 3, z = 5$
(D) $x = 5, y = 4, z = 5$

SOLUTION

$$a = (x > y)?((x > z)?x: z):((y > z)?y: z)$$

Expr 1? expr 2 : expr 3 ;

Here Expr 1 is a comparison expression whose result may be true or false. If true returned the expr 2 is selected as choice otherwise expr3.
Here we want 4 to be printed which is only in option (A) & (D) for $y = 4$ to be printed.
$x > y$ should be false since y is in true part so this expr should be true.
So both the conditions are true in option (A) only so correct.
We can check.

$x = 3 \;\; y = 4 \;\; z = 2$

$a = (x > y)?((x > z)?x: z)((y > z)?y: z)$

First we can check $3 > 2 ? 3 : 2$ thus 3 is selected
Then $4 > 2 ? 4 : 2$ here 4 is selected
Hence $a = 3 > 4?3: 4 = 4$
Hence (A) is correct option.

Question. 50

What is printed by the following C program?

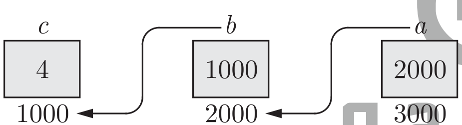
```
int f(int x, int *py, int **ppz) void main()
{                               {
     int y,z                    int  ,*  ,**a
     **ppz      z  *ppz                    a
     *py      y  *py            printf (" d",f( ,  ,a))
     x
     return x  y  z
```

(A) 18 (B) 19

(C) 21 (D) 22

SOLUTION

Here `C` is an integer variable, `b` has the address of `C` and `a` is double pointer i.e it contains address of pointer `b`.



Let c=4$c = 4$

$b = \& c = 1000$

$a \ \& b = 2000$

Call $f(4, 1000, 2000)$

$ppz = 2000 \ \ py = 1000 \ \ X = 4$

(1) $** ppz = ** ppz + 1$

$** ppz = 4 + 1 = 5 \ \ z = 5$

(2) $* py = * py + 2$

$= 5 + 2 = 7 \ \ y = 7$

(3) $X = X + 3 = 4 + 3 = 7 \ \ x = 7$

Return $(x + y + z) \Rightarrow 7 + 7 + 5 = 19$

Since both $** ppz$ and $* py$point to same memory location 1000 where C is stored.

Hence (B) is correct option.

Question. 51

Choose the correct option to fill `?1` and `?2` so that the program below prints an input string in reverse order. Assume that the input string is terminated by a newline character.

```
void reverse (void) {
      int c;
      if(?1) reverse();
      ?2
}
main () {
      printf("Enter Text"); printf("\n");
      reverse(); printf("\n");
}
```

(A) ?1 is (getchar () != '\n')
    ?2 is getchar (c);

(B) ?1 is (getchar ()) != '\n')
    ?2 is getchar (c);

(C) ?1 is (c != '\n')
    ?2 is putchar (c);

(D) ?1 is ((c = getchar ()) != '\n')
    ?2 is putchar (c);

SOLUTION

Here if and if the string comes we print the letter & do it recursively.
If C is not end to string then we move to next character in the string.
?1 should be to getchar in C & check if it is end of string.
Hence (C=getchar ()!= '\n')
?2 should be when'\n' reached so print. putchar (C);
Hence (D) is correct option.

Question. 52

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 1,2,3,4,5,6,7 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 *   *
     int temp;
     if (! list || ! list >next) return;
     p   list;     list >next;
     while ( ){
          temp=p  > value;p  >value =   > value;
            > value = temp ; p =   > next;
           = p p  > next : 0 ;
     }
}
```

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

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

SOLUTION

Here the code scans the whole list & exchanges two consecutive elements which are the nodes p & q. and then move to next two elements or $2 \longleftrightarrow 1 \quad 4 \longleftrightarrow 3 \quad 6 \longleftrightarrow 5 \quad 7$

Thus 2 1 4 3 6 5 7 is the output.

Hence (B) is correct option.

YEAR 2009

Question. 53

Consider the program below:

```
#include<stdio.h>
int fun(int n, int *f_p){
     int t,f;
     if (n<=1){
     *f_p=1
     return 1;
}
     t=fun(n-1,f_p);
     f=t+*f_p;
     *f_p=t;
     return f;
}
int main () {
     int x=15;
```

```
    printf("%d\n",fun(5,&x));
    return 0;
}
```

The value printed is:

(A) 6 (B) 8

(C) 14 (D) 15

SOLUTION

Recursion no.	n	$*f-p$	t	f
1	5	$15 \rightarrow 3$	5	8
2	4	$15 \rightarrow 2$	3	5
3	3	$15 \rightarrow 1$	2	3
4	2	$15 \rightarrow 1$	1	2
5	1	15		

Here the table column I, II, & III are during the forward calls of the recursive function fun &

The part after arrow in column III is updated value during return calls & column IV & V are the returned values.

In the end 8 is returned so only this will be printed

Hence (B) is correct option.

Question. 54

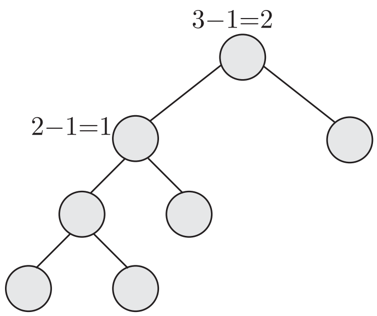
What is the maximum height of any AVL-tree with 7 nodes ? Assume that the height of a tree with a single node is 0.

(A) 2 (B) 3

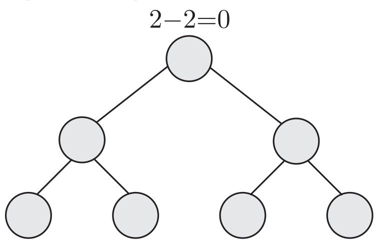
(C) 4 (D) 5

SOLUTION

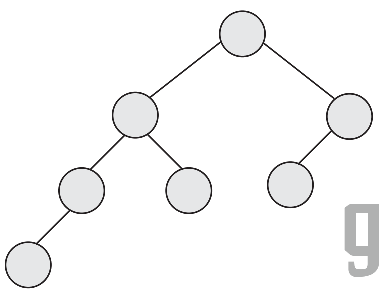
AVL tree is a partially balanced tree with the weights assigned to the nodes can be only $-1, 0$ or 1. This weight is assigned on the basis of difference of the no. of children in the left subtree & right subtree. If some other weight is there then we rotate the tree to balance it.



Here the weight is 2, so AVL tree require rotation.



This is balanced.



In this tree the height is 3 and it is Also AVL balanced so maximum height will be 3

Hence (B) is correct option.

Statement for Linked Answer Question 55 & 56

Consider a binary max-heap implemented using an array

Question. 55

Which one of the follow9ng array represents a binary max-heap?

(A) {25, 12, 16, 13, 10, 8, 14}

(B) {25, 14, 13, 16, 10, 8, 12}

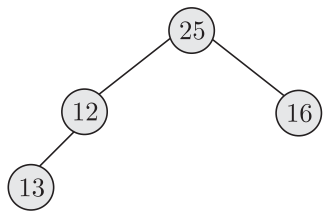
(C) {25, 14, 16, 13, 10, 8, 12}

(D) {25, 14, 12, 13, 10, 8, 16}

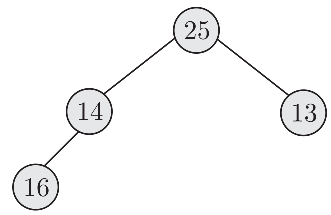
SOLUTION

If the value presented at any node is greater then all its children then the tree is called the **max heap**.

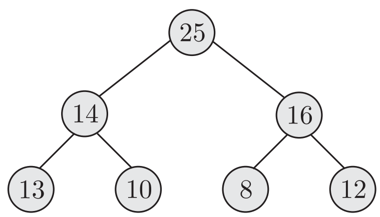
Here we need to draw heap for all options



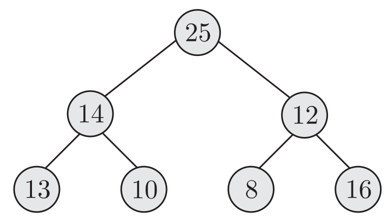
Failed at 13



Failed at 16



Sucessfull



Failed at 16

Hence (C) is correct option.

Question. 56

What is the content of the array after two delete operations on the correct answer to the previous question?

(A) {14, 13, 12, 10, 8}

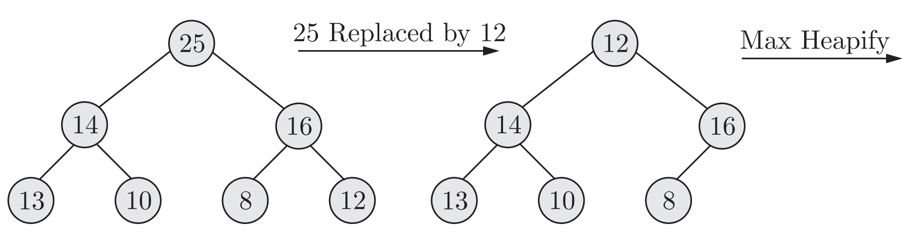
(B) {14, 12, 13, 8, 10}

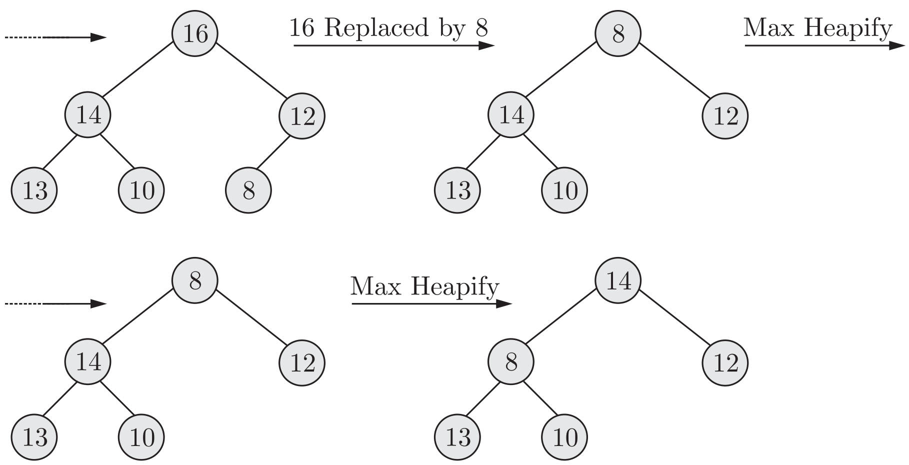
(C) {14, 13, 8, 12, 10}

(D) {14, 13, 12, 8, 10}

SOLUTION

Given Max heap





So array contents are

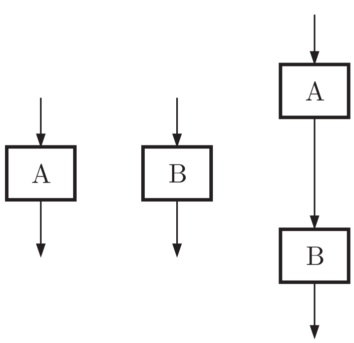
14 13 12 8 10

Hence (D) is correct option.

YEAR 2010

Question. 57

The cyclomatic complexity of each of the modules A and B shown below is 10. What is the cyclomatic complexity of the sequential integration shown on the right hand side ?



(A) 19 (B) 21

(C) 20 (D) 10

SOLUTION

Cyclomatic complexity is defined as the no. of independent paths form begin to exit in a module. If some dependent sub modules are there in the module then the individual cyclomatic complexities are added & overall sum is reduced by 1.

Here $\quad CC(\text{complete}) = CC(A) + CC(B) - 1$

So $\quad 10 + 10 - 1 = 19$

Hence (A) is correct option.

Question. 58

What does the following program print ?

```
#include<stdio.h>
void        f(int *p, int *q){
            p=q
            *p
}
int i=0,j=1;
int main(){
            f(&i,&j);
            printf("%d%d\n",i,j);
            return 0;
}
```

(A) 22 (B) 21

(C) 01 (D) 02

SOLUTION

Here in the function `p&q` are integer pointer and `p = q` statement means now `p` contains the same address as contained by `p*p=2` means now both locations contain 2. Since `p & q` both contain same location so value of `*p` & `*q` is 2. But the address `i` hasn't be modified so value of `i` is not changed. It is `a & j` has been changed to 2.

Hence (D) is correct option.

Question. 59

What is the appropriate paring of items in the two columns listing various activities encountered in a software life cycle ?

P.	Requirement Capture	1.	Module Development and Integration
Q.	Design	2.	Domain Analysis
R.	Implementation	3.	Structural and Behavioral Modeling
S.	Maintenance	4.	Performance Tuning

(A) P-3 Q-2, R-4 S-1 (B) P-2 Q-3 R-1 S-4

(C) P-3 Q-2 R-1 S-4 (D) P-2 Q-3 R-4 S-1

SOLUTION

All of these steps are part of a simple software development life cycle (SWDLC)

P. Requirement Capture : Considered as first step where we analyze the problem scenario, domain of input, range of output and effects.

P Design : Knowing the problem a systematic structure of the problem solution is designed and the behavior modelling refers to the functions of the module, which give certain output providing definite inputs.

R Implementation : After knowing behavior the modules are developed, converting the logics in the programming logics. The independent modules are then integrated.

S Maintenance : Successful implementation done but even then the performance might not optimal so some features or methods need to be change to tune the performance.

Hence (B) is correct option.

Question. 60

What is the value printed by the following C program ?

```
#include<stdio.h>
int f(int *a, int n)
{
     if (n<=0) return 0;
     else if (*a%2==0) return *a+f(a+1,n-1);
     else return *a-f(a+1,n-1);
}
```

```
int main()
{
     int a[]={12, 7, 13, 4, 11, 6};
     printf("%d",f(a,6));
     return 0;
}
```

(A) -9 (B) 5

(C) 15 (D) 19

SOLUTION

Recursion	$+a$	n	Return (x)	Final return
1	12	6	3	$*a+x=12+3=15$
2	7	5	4	$*a-x=7-4=3$
3	13	4	9	$*a-x=13-9=4$
4	4	3	5	$*a+x=4+5=9$
5	11	2	6	$*a-x=11-6=5$
6	6	1	0	$*a+x=6+0=6$
7		0	0	

Hence finally 15 will be returned.

Hence () is correct option.

Question. 61

The following C function takes a singly-linked list as input argument. It modified the list by moving the last element to the front of the list and returns the modified list. Some part of the code is left blank.

```
typedef struct node {
          int value;
          struct node *next
} Node;
Node *mode_to_front(Node *head){
Node*p,* ;
if((head==NULL)||(head->next==NULL))return head;
q=NULL;p=head;
```

```
while(p->next!=NULL){
          q=p;
          p=q->next;
}
________________________
return head;
}
```

Choose the correct alternative to replace the blank line.

(A) `q=NULL;p->next=head;head=p;`

(B) `q->next=NULL;head=p;p->next=head;`

(C) `head=p;p->next=q;q->next=NULL;`

(D) `q->next=NULL;p-next=head;head=p;`

SOLUTION

Here the program wants to make the last node of the list, the first node.

Here q is the second last node and p is the last node

→ The second last node's next should be now NULL so q->next=NULL.

→ p->next should below head node.
so p->next=head

→ Now the head node is p.
So head = p.

Hence (D) is correct option.

Question. 62

The following program is to be tested for statement coverage :

```
begin
     if(a==b){S1;exit}
     else if(c==d){S2;}
     else {S3;exit;}
S4;
end
```

The test cases T1, T2, T3, and T4 given below are expressed in terms of the properties satisfied by the values of variables a b c and d. The exact values are not given.

T1 : a b c and d are all equal

T2 : a, b, c and d are all distinct

T3 : a = b and c! = d

T4 : a! = b and c = d

Which of the test suites given below ensures coverage of statements S1, S2, S3 and S4 ?

(A) T1, T2, T3 (B) T2, T4

(C) T3, T4 (D) T1, T2, T4

SOLUTION

The following test cases covers statements.

T_1: all are equal
S_1 executed and then so no other execution.

T_2: all are distinct
Only S_3 executed

T_3: a=b & c!=d
Only S_1 executed

T_4: !=b & c=d
Only S_2, S_4 only

So to have all statements the option should be either T_1, T_2, T_4 *or* T_2, T_3, T_4

Option (D) is T_1, T_2, T_4

Hence (D) is correct option.

Statement for Linked Answer Questions 63 & 64

A has table of length 10 uses open addressing with hash function h(k)=k mod 10, and linear probing. After inserting 6 values into an empty has table, the table is as shown below.

0	
1	
2	42
3	23
4	34
5	52
6	46
7	33

8	
9	

Question. 63

Which one oft he following choices gives a possible order in which the key values could have been inserted in the table ?

(A) 46, 42, 34, 52, 23, 33 (B) 34, 42, 23, 52, 33, 46

(C) 46, 34, 42, 23, 52, 33 (D) 42, 46, 33, 23, 34, 52

SOLUTION

Here for hashing Linear probing is used, i.e. it finds the hash key value through hash function and maps the key on particular position In Hash table. In case of key has same hash address then it will find the next address then it will find the next empty position in the Has Table.

Here we check all options:

(A) Here 42 will be inserted at the 2nd position in the array next 52, also has same hash address 2. But it already occupied so it will search for the next free place which is 3rd position. So here 52 is misplaced and it is not possible key values.

Table 3 table

(B) Here 46 is misplaced so it is not possible value.

(C) This is same as given hash table.

So correct order is 46, 34, 42, 23, 52, 33

Hence (C) is correct option.

Question. 64

How many different insertion sequences of the key values using hte same hash function and linear probing will result in the hash table shown above ?

(A) 10 (B) 20

(C) 30 (D) 40

SOLUTION

Here the given order of insertion is 46, 34, 42, 23, 52, 33

Figure

→ Here 42, 23, 34, 46 are inserted direct using hash function

→ But to insert 52 we have 6 vacant places.

→ After insertion of 52 at any of the 6 places, we have 5 places remaining for 33.

So total combination.

$6 \times 5 = 30$ possible ways

Hence (C) is correct option.

****