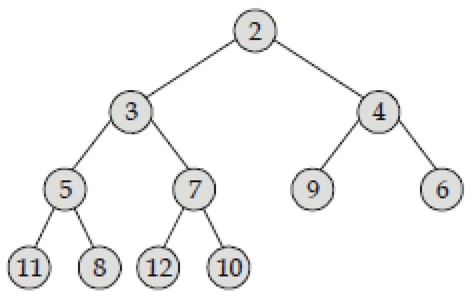
MULTIPLE SUBJECT : ALGORITHMS + PROGRAMMING AND DATA STRUCTURES (GATE - 2019) - REPORTS

Q. 1

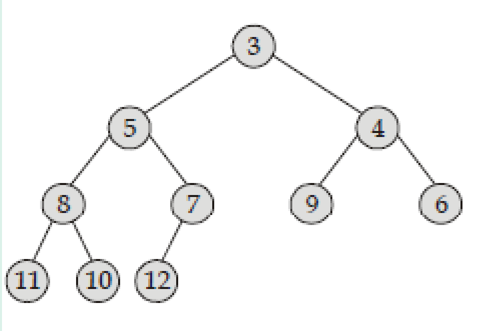
Consider the binary min heap given below:
(min heap is a binary tree where each node in a tree has a key which is less than or equal to the key of its children)



Insert the key 1 in above min heap. Which of the following is the resultant min heap after two delete operations?

Have any Doubt ?

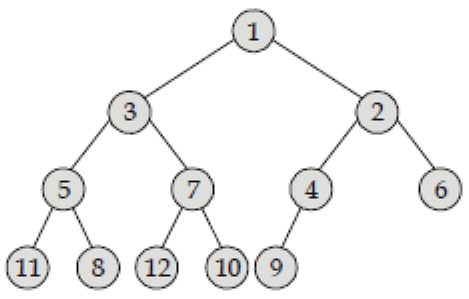
A



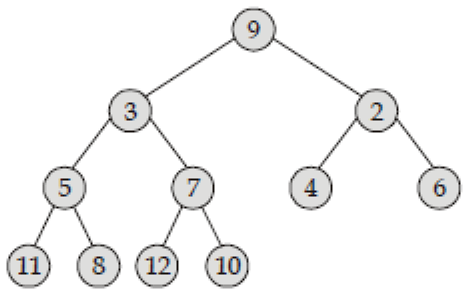
Correct Option

Solution :
(a)

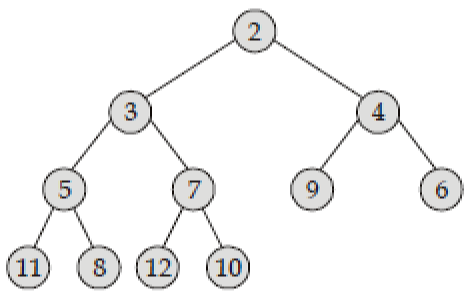
When we insert 1 in the heap



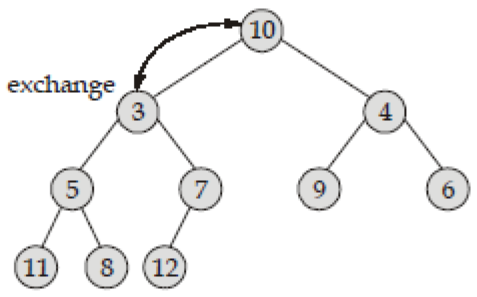
For deleting 1, 1 is exchanged with last element of min heap.

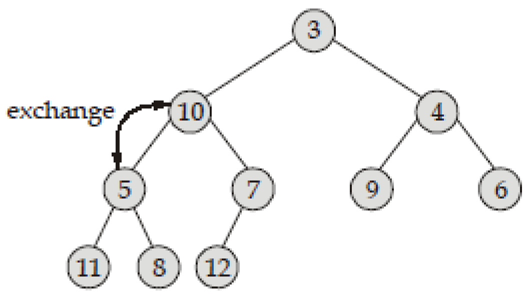


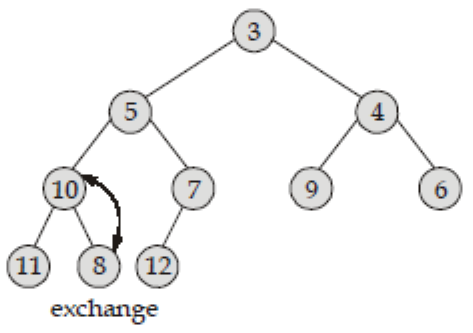
Now we have to maintain the min heap property.
After balancing the min heap it become

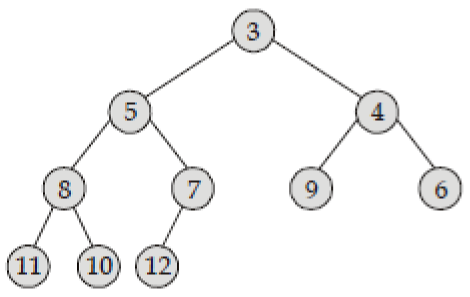


Now delete 2



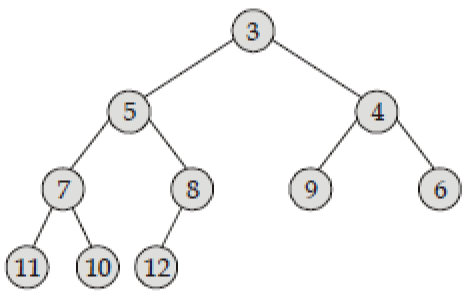




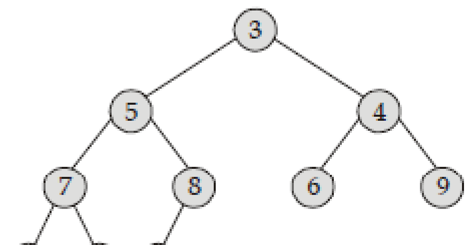


This is the resultant heap so correct answer is option (a).

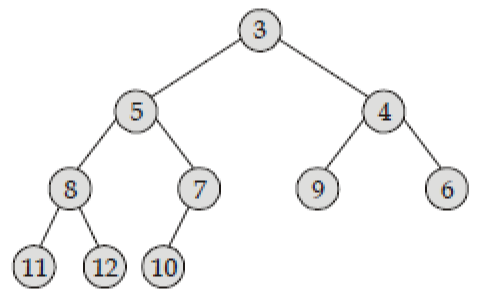
B



C



D



QUESTION ANALYTICS +

Q. 2

Match **List-I** with **List-II** and select the correct answer using the codes given below the lists:

List-I	List-II
A. $T(n) = T(n - 1) + n$	1. $O(n\log n)$
B. $T(n) = T\left(\frac{n}{2}\right) + 1$	2. $O(n)$
C. $T(n) = 2T\left(\frac{n}{2}\right) + n$	3. $O(\log n)$
D. $T(n) = 2T\left(\frac{n}{2} + 8\right) + n$	4. $O(n^2)$

Codes:

	A	B	C	D
(a)	4	3	1	3
(b)	4	2	3	1
(c)	4	1	2	3
(d)	4	3	1	1

Have any Doubt ?

A a

B b

C c

D d — Correct Option

Solution :
(d)

A. $T(n) = T(n - 1) + n \Rightarrow T(n) = O(n^2)$

B. $T(n) = T\left(\frac{n}{2}\right) + 1 \Rightarrow T(n) = O(\log n)$

C. $T(n) = 2T\left(\frac{n}{2}\right) + n \Rightarrow T(n) = O(n\log n)$

D. $T(n) = 2T\left(\frac{n}{2} + 8\right) + n \Rightarrow T(n) = O(n\log n)$

Therefore, (d) is correct.

QUESTION ANALYTICS +

Q. 3

Consider the following program:

```
# include <stdio.h>
int * ptr[12];
int main ( )
{
    if (* (ptr + 5) = = * (ptr + 3))
    {
        printf("Equal");
    }
    else
    {
        printf("Not Equal");
    }
    return 0;
}
```

What will be the output of the above program?

Have any Doubt ?

A It will always print Equal — Correct Option

Solution :
(a)
Here ptr is a global array of pointer to int, global variable are initialized to 0. All the element of ptr are initialized implicity to 0.
So correct option is (a).

B It will always print Not Equal

C Compiler error

D Run time error

QUESTION ANALYTICS +

Q. 4

What is the time complexity to find the maximum number of edge to be added to a tree so that it stays a bipartite graph?
(a) O(V + E) (b) O(VE)
(c) O(V) (d) O(V^2)
(Where V is the number of nodes in a tree and E is the number of edges)

FAQ | Have any Doubt ?

A a

B b

C c — Correct Option

Solution :
(c)

To find the maximum number of edge to be added to a tree so that it stays a bipartite graph we do the following:

1. Do a simple DFS (as BFS) traversal of tree and color is with two colors.
2. While coloring keep track of counts of nodes colored with the two color. Let counts be col_1 and col_2.
3. Now we know maximum edges a bipartite graph can have are $\text{col}_1 * \text{col}_2$ and tree with V nodes has V – 1 edge.
4. So answer is $\text{col}_1 * \text{col}_2 - (V - 1)$

   A tree is always a bipartite graph as we can always break into two disjoint sets with alternate levels.

   For DFS as BFS tree traversal time complexity is O(V).

   Time complexity of this algorithm is O(V).

So correct option is (c).

D d

QUESTION ANALYTICS +

Q. 5

Consider the following statements:

$S_1 : n^{2019} = O(n^{2020})$

$S_2 : O(n^{2019}) = O(n^{2020})$

Which of the above statements is/are correct?

Have any Doubt ?

A Both S_1 and S_2

B Only S_1 — Correct Option

Solution :
(b)

It's easy to see why S_1 is true.

However S_2 is false. Note that S_2 is actually comparing two sets.

$O(n^{2019})$ is actually a set containing all functions asymptotically smaller than n^{2019}.

$O(n^{2020})$ contains all functions asymptotically smaller than n^{2020}.

Now take $n^{2019.5} \Rightarrow n^{2019.5} \notin O(n^{2019})$

but $n^{2019.5} \in O(n^{2020})$

Hence $O(n^{2019}) \neq O(n^{2020})$

Only S_1 is correct.

C Only S_2

D None of these

QUESTION ANALYTICS +

Q. 6

A function is given below:

```
void printN (int n) {
    printf ("%d", n);
    if (n = = 0)
        return n;
    else if (n% 2 = = 1)
        printN (2 * n);
    else
        printN (n/2 – 1);
```

printN function is called with printN(5) what is the 5th value printed by this function?

Have any Doubt ?

A 3

B 4

C 2 — Correct Option

Solution :
(c)

1. printN (5) is called
   first it print the value 5

```
   void printN (int n) {
       printf ("%d", n);
```

```
    if (n = = 0)                          ... condition (1)
        return n;
    else if (n% 2 = = 1)                  ... condition (2)
        printN (2 × n);
    else
        printN (n/2 – 1);
}
```

5 is not 0 so it go to else if part condition is satisfied printN (10) is called.

2. printN (10)
   10 is printed
   condition (1) and (2) is not satisfied it go to else part and printN(4) is called.
3. printN (4)
   4 is printed
   condition (1) and (2) is not satisfied it go to else part and printN (1) is called.
4. printN(1)
   1 is printed
   condition (2) is satisfied printN (2) is called.
5. printN (2)
   2 is printed this is the 5th element.

So correct option is (c).

D 7

QUESTION ANALYTICS +

Q. 7

Consider the following functions, foo() and bar():

```
int foo(int n)
{
    if (n <= 1) return 2^n;
    return 8*foo(n – 1);
}
int bar(int n)
{
    if (n <= 1) return 3^n;
    return 3*bar(n – 1) + 2*bar(n –1);
}
```

The time complexity of the functions foo() and bar() is

Have any Doubt ?

A O(2^n) for both foo() and bar()

B O(2^n) for foo() and O(3^n) for bar()

C O(n) for foo() and O(2^n) for bar() Correct Option

Solution :
(c)

The recurrence relation for foo(n):

$$T(n) = T(n-1) + c; \text{ where } c \text{ is a constant}$$

$$\Rightarrow \quad T(n) = O(n)$$

The time recurrence for bar(n):

$$T(n) = 2T(n-1) + c; \; c \text{ is a constant}$$

$$\Rightarrow \quad T(n) = O(2^n)$$

D O(n) for both foo() and bar()

QUESTION ANALYTICS +

Q. 8

Through an experiment, it is found that selection sort performs 5000 comparisons when sorting an array of size k. If the size of the array is doubled, what will be the number of comparisons?

Have any Doubt ?

A 5000

B 10000

C 20000 Correct Option

Solution :
(c)

Selection Sort algorithm makes comparisons proportional to n^2, irrespective of the input. Hence, if the array size is doubled, the comparisons will become $2^2 = 4$ times.
Therefore, number of comparsions: 5000 × 4 = 20000.

D None of these

QUESTION ANALYTICS +

Q. 9

Let A be a three-dimensional array declared as follows:
A: array [2 ..... 10] [15 ..... 25] [10 ..... 21]
Assuming that each integer takes four byte. The array in stored in row major order and the first element of the array is stored at location 100, what is the address of the element A[5] [17] [16] ________.

Have any Doubt ?

● 1804 Correct Option

Solution :
1804

Location of the element

A[5] [17] [16] = Base address + [((5 – 2) × 11 × 12 + (17 – 15) × 12 + (16 – 10)) × Size of element]

= 100 + [(3 × 11 × 12 + 2 × 12 + 6) × 4]

= 100 + [396 + 24 + 6] × 4

= 100 + [426] × 4

= 100 + 1704 = 1804

QUESTION ANALYTICS +

Q. 10

Consider the following function written in a C like language:

```
int Bar (int n)
{
     if (n < 2) return;
     else
     {
          int sum = 0;
          int i, j;
          for (i = 1; i < = 4; i++) Bar (n/2);
          for (i = 1; i < = n; i++)
          {
               for (j = 1; j < = i; j++)
               {
                    sum = sum + 1;
               }
          }
     }
}
```

Now consider the following statements:

S_1 : The time complexity of Bar(n) is $\theta(n^2 \log(n))$.

S_2 : The time complexity of Bar(n) is $\Omega(n^2 \log(n^2))$.

S_3 : The time complexity of Bar(n) is $O(n^3 \log(n^2))$.

The number of correct assertions are ________.

Have any Doubt ?

● 3 Correct Option

Solution :
3

Recurrence relation for Bar(n):

$$T(n) = 4T\left(\frac{n}{2}\right) + \theta(n^2)$$

$$\Rightarrow \quad T(n) = \theta(n^2 \log n)$$

S_1 is correct.

Now $\quad n^2 \log(n^2) = n^2 \cdot 2 \log n = \Omega(n^2 \log n)$

Hence $T(n)$ can also be written as, $T(n) = \Omega(n^2 \log n)$

$\therefore$ S_2 is correct.

Similarly S_3 is also correct.

QUESTION ANALYTICS +

Q. 11

There is given a infix expression:

$$A + B * C/((D + E) + F * \overset{1}{G})$$

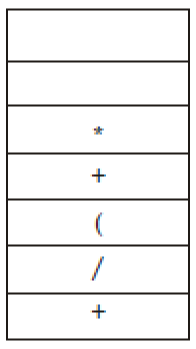
While converting the infix expression to postfix expression number of symbol in the stack at the indicated point 1 in the infix expression (assume stack is initially empty) ________.

Have any Doubt ?

● 5 Correct Option

Solution :
5

While converting the infix expression to postfix symbol in the stack at point 1 the status of stack is



Stack

Postfix expression is ABC * DE + FG * + / +.

Total there is 5 symbol at the point 1.

QUESTION ANALYTICS +

Q. 12

Consider the following message:

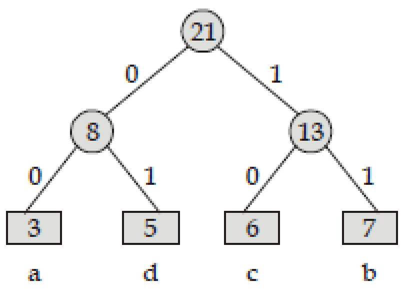
aabbbbabccdddccccbbdd

The number of bits required for Huffman coding of the above message is _______.

Have any Doubt ?

● 42 Correct Option

Solution :
42

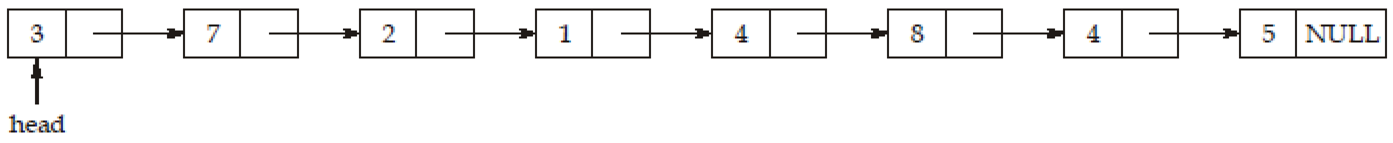


Number of bits = 3 × 2 + 7 × 2 + 5 × 2 + 6 × 2
= 42 bits

QUESTION ANALYTICS +

Q. 13

A singly linked list is given below:



```
        struct node
        {
            struct node * next;
            int data;
        }
   void predict (struct node * head)
        {
            struct node * current;
            current = head;
   while (current → next → next! = NULL)
            current = current → next;
   current → next → next = head;
            head = current → next;
            current → next = NULL;
            current = NULL;
        }
```

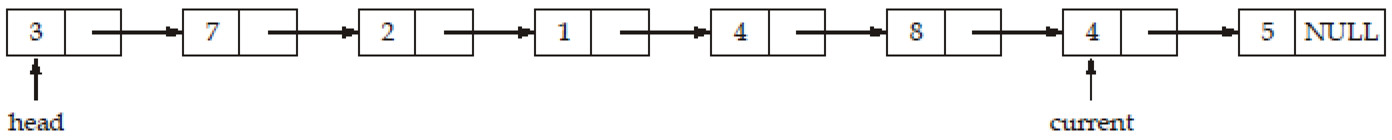
When head of the given linked list is pass to the function predict after executing this code head is pointing to the node which contain the value ________.

Have any Doubt ?

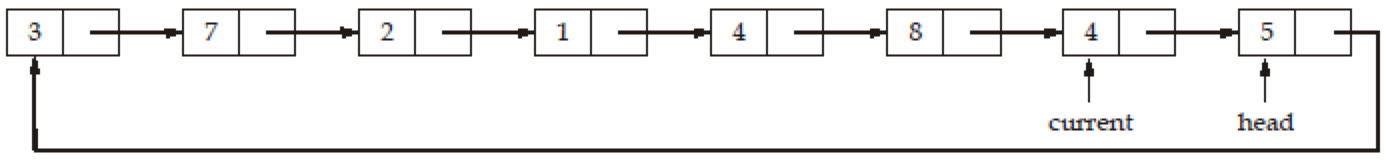
● 5 Correct Option

Solution :
5

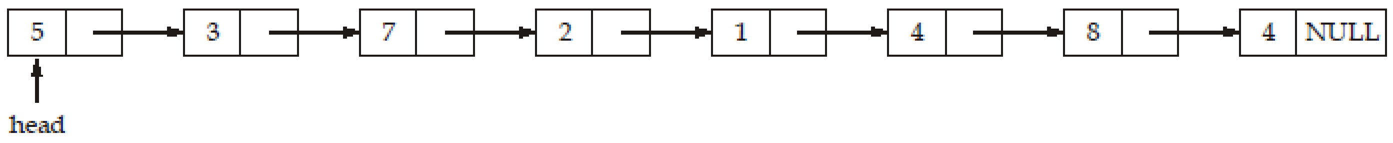
We pass the head pointer the function predict.
current = head, current is pointing to head
while (current → next → next! = NULL)
current = current → next;
After executing this loop list is



current → next → next = head
current → next → next, contain the address of head node
head = current → next
After this line head is pointing to node contain value 5.



current → next = NULL;
current = NULL
After this the linked list look like this



So head is pointing to node contain value 5.

QUESTION ANALYTICS +

Q. 14

Let *q* be a queue and S be a stack. The function dequeue and pop are the conventional operation that they return whatever they remove. Assume that *q* and S are initially empty and *i* has been declared as an int.

```
        enqueue (q, 5);
        enqueue (q, 2);
        push (S, 4);
        push (S, 1);
   for (i = 0; i < 5; i++) {
        printf ("%d", dequeue(q));
        printf ("%d", pop(S));
            enqueue (q, i + 2);
            push (S, i + 6);
   }
```

What is the sum of all value printed by this code fragment ________.

Have any Doubt ?

● 47 Correct Option

Solution :
47

q be a queue and S be a stack

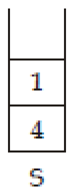
enqueue (q, 5)   5

enqueue (q, 2)   5   2

↑   ↑

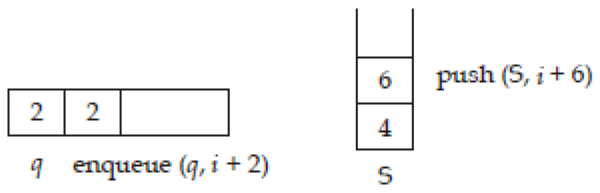
front   rear

push (S, 7)   push (S, 9)



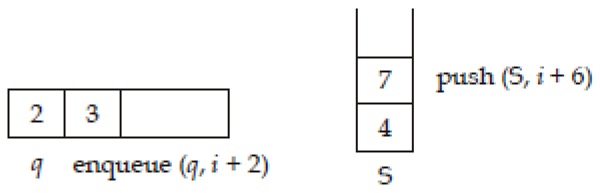
in the for loop for $i = 0$

dequeue (q) gives 5 and 5 is printed pop (S) return 1 and 1 is printed.



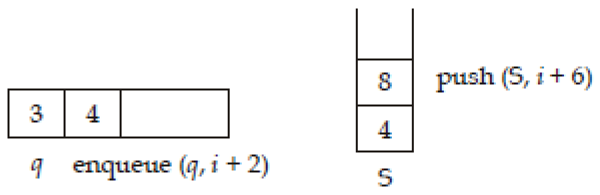
for $i = 1$

dequeue (q) print 2 and pop (S) print 6



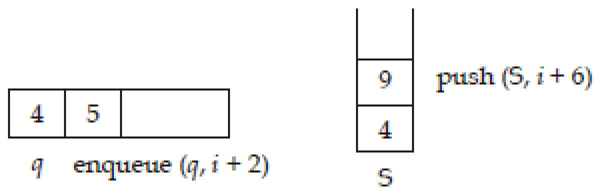
for $i = 2$

dequeue (q) print 2 and pop (S) print 7



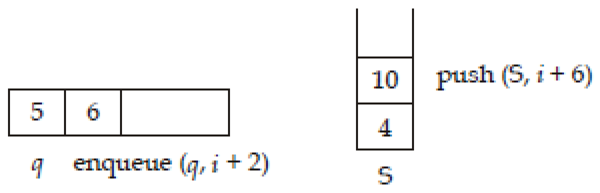
for $i = 3$

dequeue (q) print 3 and pop (S) print 8



for $i = 4$

dequeue (q) print 4 and pop (S) print 9



for $i = 5$ condition become false and it come out from the for loop so value printed it

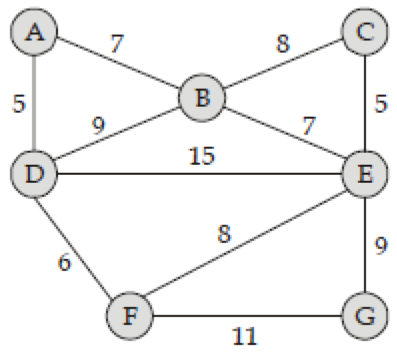
5 1 2 6 2 7 3 8 4 9

So answer is 47.

QUESTION ANALYTICS +

Q. 15

Consider the following graph G:



The sum of edge weights of the spanning tree formed using Kruskal's algorithm on graph G is ________.

Have any Doubt ?

39   Correct Option

Solution :
39
MST of G will have AD, CE, DF, AB, BE, GE as edges. Hence sum will be,
(5 + 5 + 6 + 7 + 7 + 9) = 39

QUESTION ANALYTICS +

Q. 16

Consider an array A given below:
A = 30, 15, 48, 34, 26, 29
Let X be the number of inversions in the given array A. Now, another array B is constructed by making all the numbers in A negative and keeping the order between each number same. Let the number of inversions in the modified array so obtained be Y. Then X + 2Y = ________.

Have any Doubt ?

22   Correct Option

Solution :
22

Inversion pairs in A = (1, 2) (1, 5) (1, 6) (3, 4) (3, 5) (3, 6) (4, 5) (4, 6)

Hence   X = 8

Now the new array after modifications

$$A' = \begin{bmatrix} \underset{1}{-30}, \underset{2}{-15}, \underset{3}{-48}, \underset{4}{-34}, \underset{5}{-26}, \underset{6}{-29} \end{bmatrix}$$

Inversion pairs in A′ = (1, 3) (1, 4) (2, 3) (2, 4) (2, 5) (2, 6) (5, 6)

Therefore   Y = 7

$$\begin{aligned} X + 2Y &= 8 + 2(7) \\ &= 8 + 14 = 22 \end{aligned}$$

QUESTION ANALYTICS +

Q. 17

Consider the following statements about a binary tree:
S_1 : The number of leaves (nodes with no children) in each left subtree is differ by atmost one to the number of leaves in the corresponding right subtree. This tree has worst case height O(logn) (where n is the number of nodes in the binary tree).
S_2 : The number of node in each left subtree is within a factor of 2 of the number of nodes in the corresponding right subtree. Also, a node is allowed to have only one child if that child has no children. This tree has worst case height O(logN). (where N is the number of nodes in the binary tree).

Have any Doubt ?

A Both statements are true

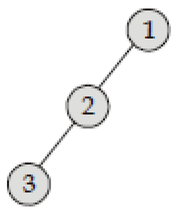
B Only S_1 is true

C Only S_2 is true Correct Option

Solution :
(c)

S_1 : Consider a tree of n nodes, where node 1 is the root and node $i > 1$ is the child of node $i - 1$. For each node, the left subtree has one leaf, whereas the right subtree has zero. This meets the "balancing" condition. The height of the tree is O(n).

$i = 3$



For node 1 number of leaves in left subtree is 1 and in right it is 0 same for the node 2.
For node 3 number of leaves in left subtree and in right subtree is 0 so balancing condition is satisfied.
So S_1 is false.

S_2 : The proof is very similar to the AVL proof.
Let $N(h)$ be the minimum number of nodes contained in a tree with height h.
The base cases are $N(0) = 0$

$$N(1) = 1 \text{ and } N(2) = 2$$

Recurrence for N

$$\begin{aligned} N(h) &= 1 + N(h-1) + \frac{1}{2}N(h-1) \\ &= 1 + \frac{3}{2}N(h-1) \end{aligned}$$

Because a tree with height h must have one subtree of height $h - 1$ and the other subtree has at least half the number of nodes in that subtree.
The solution to this recurrence is

$$N(h) = \theta\left(\frac{3}{2}\right)^h \text{ which gives } h = \theta(\log N) \text{ as desired}$$

So correct option is (c).

D Both statements are false

QUESTION ANALYTICS +

Q. 18

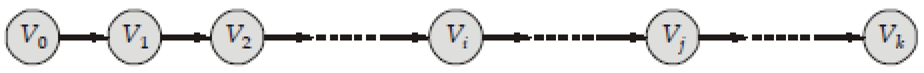
Consider the following statements:
S_1 : If all edge weights of a graph are positive, then any subset of edges that connects all vertices and has minimum total weight is a tree.
S_2 : Let $p = <V_0, V_1, V_2 \ldots\ldots, V_k>$ be the shortest path from vertex V_0 to V_k and for all i, j such that $0 \le i \le j \le k$, let p_{ij} be the subpath of p from vertex V_i to V_j.
Then p_{ij} is a shortest path from V_i to V_j.
Which of the above statements is/are correct?

Have any Doubt ?

A Both S_1 and S_2 Correct Option

Solution :
(a)

It's quite easy to understand S_1. So let's see why S_2 is correct. We'll prove S_2 by contradiction.



If we assume that $V_i \rightarrow V_j$ is not shortest, then it means there's a better path and we can use this path, thus reducing the shortest distance from V_0 to V_k. However this contradicts the fact that p is the shortest path from V_0 to V_k and hence $V_i \rightarrow V_j$ has to be shortest, $\forall 0 \le i \le j \le k$. (i.e. i and j must lie between 0 and k, and j must be greater than or equal to i).

B Only S_1

C Only S_2

D None of these

QUESTION ANALYTICS +

Q. 19

Which of the following functions exhibits the fastest growth rate as n grows closer to infinity?

I. $(\log \log n)!$

II. $(\log \log n)^{\log n}$

III. $2^{\sqrt{\log \log n}}$

IV. $(\log \log n)^{\log \log \log n}$

Have any Doubt ?

A I only

B II only Correct Option

Solution :
(b)

Let, $n = 2^{2^k}$

$$\left.\begin{array}{ll} \text{I.} & k! \\ \text{II.} & k^{2^k} \\ \text{III.} & 2^{\sqrt{k}} \\ \text{IV.} & k^{\log k} \end{array}\right\} \Rightarrow \text{Its quite clear to see that II is the greatest among all the 4 competitors.}$$

Hence (b) is the correct answer.

C III only

D IV only

QUESTION ANALYTICS +

Q. 20

Consider the following statements:
S_1 : Running a DFS on an undirected graph G = (V, E) (where V is set of vertices and E is set of edges) always produces the same number of cross edges no matter what order the vertex list V is in and no matter what order the adjacency lists for each vertex are in.
S_2 : In a breadth first search of an undirected graph there is no forward edges but there can be back edges.

Have any Doubt ?

A Both statements are true

B Only S_1 is true Correct Option

Solution :
(b)

S_1 : DFS in an undirected graph never produces cross edges. So S_1 is true.

S_2 : There is no forward and no back edges in breadth first search of an undirected graph. Suppose (u, v) is a back edge or a forward edge in a BFS of an undirected graph. Then one of u, say u is proper ancestor of the other (v) in the breadth-first tree. Since we explore all edge of u before exploring any edge of any of u's descendants, we must explore the edge (u, v) at the time we explore u. But then (u, v) must be a tree edge. So S_2 is false

So option (b) is correct.

C Only S_2 is true

D Both statements are false

QUESTION ANALYTICS +

Q. 21

What is the time complexity of best known algorithm for reversing a singly linked list in a group of given size k for example if the given linked list is 1 → 2 → 3 → 4 → 5 → 6 → 7 → 8 → NULL and k is 3 then output should be 3 → 2 → 1 → 6 → 5 → 4 → 8 → 7 → NULL?
(Where n is the number of node in the linked list.)

Have any Doubt ?

A $O(n)$ Correct Option

Solution :
(a)

To reverse the singly linked list in a group of given size k.

1. Reverse the first sub-list of size k while reversing keep track of the next node and previous node. Let the pointer to the next node be next and pointer to the previous node be previous.
2. Head → next = reverse (next, k) (recursively call for rest of the list and link the two sublists).
3. return prev (prev becomes the new head of the list)

```
while (current! = NULL && count < k)
{
    next = current → next;
    current → next = prev;
        prev = current;
        current = next;
        count ++;
}   /* reverse first k nodes */
    if (next! = NULL)
    head → next = reverse (next, k);
    return prev;
}
```

Here every node is processed just once, time complexity of the function is $O(n)$.

So correct option is (a).

B O($k.n$)

C O(n^2)

D O($n\log k$)

QUESTION ANALYTICS +

Q. 22

Consider the following C code:

```
int getNextGap(int gap)
{
    gap = (gap*10)/13;
    if (gap < 1)return 1;
    return gap;
}
void mystery(int a[ ], int n)
{
    int gap = n;
    bool red = true;
    while (gap! = 1||red == true)
  {
         gap = getNextGap(gap);
         red = false;
         for (int i = 0; i < n-gap; i++)
         {
             if (a[i] > a[i + gap])
             {
                 swap(a[i], a[i + gap]);
                 red = true;
             }
         }
    }
}
```

The array A = {9, 4, –1, 3, 5, 7, 99, –33, 104} is passed to the above function mystery, with n equal to the number of integers in A. The final contents of the array after the function has been executed is

Have any Doubt ?

A –33, –1, 3, 4, 5, 7, 9, 99, 104 Correct Option

Solution :
(a)
The algorithm sorts the array in ascending order, hence (a) is the answer.

B –33, –1, 3, 4, 5, 104, 99, 9, 7

C 104, 99, 9, 7, –33, –1, 3, 4, 5

D –33, 104, 3, 4, 5, 7, 9, 99, –1

QUESTION ANALYTICS +

Q. 23

The function shown in the pseudocode below is used to remove duplicates from a sorted singly linked list.

```
     struct node
     {
         int data;
         struct node * next;
     }
 void remove duplicates (struct node * head)
 {
     struct node * current = head;
     struct node * next_node;
     if (current == NULL)
         return;
     while (current → next! = NULL)
 {
     if (current → data = = current → next → data)
     {
             A;
         free (current → next);
             B;
     }
     else
     current = current → next; }
 }
```

If pointer to the head of linked list is pass to given function, the appropriate expression for the two boxes A and B are?

Have any Doubt ?

A A : next_node = current → next
B : current → next = next_node

B A : next_node = current → next → next
B : current → next = next_node
Correct Option

Solution :
(b)
In this algorithm we traverse the list from the head node while traversing, compare each node with its next node, if data of next node is same as current node then delete the next node. Before

we delete a node, we need to store next pointer of the node

```
    struct node * current = head;
    struct node * next_node;
    /* pointer which store the next pointer of a node to be deleted */
    if (current = = NULL)
        return;
    while (current → next! = = NULL)
    if (current → data = = current → next → data)
```

It compare current node with next node if it is same then

```
    next_node = current → next → next                              ...A
    (store the pointer of the node after duplicate node)
        free (current → next)                                      ...B
    (it delete the duplicate node and free the space)
    current → next = next_node
```

Now current → next point to the after the duplicate node which we deleted, so the correct sequence is

```
 A  next_node = current → next → next;
    free (current → next);
 B  current → next = next_node;
```

So correct option is (b)

C A : next_node = current → next → next
B : current → next = next_node → next

D A : next_node = current → next
B : current → next = next_node → next

QUESTION ANALYTICS +

Q. 24

Consider the Insertion Sort procedure given below, which sorts an array L of size n(≥ 2) in ascending order:

```
begin
    for xindex: = 2 to n do
        x : = L [xindex];
        j : = xindex – 1;
        while j > 0 and L[j] > x do
            L[j + 1] : = L[j];
            j : = j – 1;
        end {while}
        L [j + 1] : = X;
    end{for}
end
```

It is known that insertion sort makes at most n(n – 1)/2 comparisons. Which of the following is true?

Have any Doubt ?

A There is no input on which insertion sort makes n(n – 1)/2 comparisons.

B Insertion sort makes n(n – 1)/2 comparisons when the input is already sorted in ascending order.

C Insertion sort makes n(n – 1)/2 comparisons only when the input is sorted in descending order. Correct Option

Solution :
(c)

In worst case Insertion sort will have $n(n - 1)/2$ comparisons i.e. when input is sorted in descending order.

```
50 40 30 20 10 ..... n
pass 1: 50 40 30 20 10 ..... n           0 comparison
pass 2: 40 50 30 20 10 ..... n           1 comparison
      ⋮
pass n: n ..... 10 20 30 40 50           n – 1 comparisons
```

Total $1 + 2 + 3 + ... + (n - 1) = n(n - 1)/2$.

D There are more than one input orderings where insertion sort makes n(n – 1)/2 comparisons.

QUESTION ANALYTICS +

Q. 25

Consider the following function:

```
int fun (int a[ ], int l, int target)
{
        int i = 0, j = 0, sum = 0, count = 0;
    while (j < l) {
        if (sum < target) {
            sum = sum + a[j];
            j++;
    }
    else if (sum > target) {
            sum = sum – a[i];
            i++;
    }
        else {
            count ++;
            sum = sum – a[i];
            i++;
        }
    }
    if (sum = = target)
    if (sum == target)
        count ++;
    return count;
}
```

If a[ ] states the elements

a[ ] = {2, 3, 3, 2, 5, 4, 1, 3, 6, 8, 2, 3, 4, 4, 2, 2}

What would be the return value of the function call fun (a, 16, 8) ________.

Have any Doubt ?

● 6 Correct Option

Solution :
6

function is called with fun(a, 16, 8)

This function return value of count, count is incremented when sum = target.

For the following value of i and j sum is equal to target and value of count is incremented.

1. for $i = 0$ and $j = 2$
   (sum = = target) and value of sum is 8 count is incremented
2. for $i = 1$ and $j = 3$
   count is incremented
3. for $i = 5$ and $j = 7$
   count is incremented
4. for $i = 9$ and $j = 9$
   count is incremented
5. for $i = 12$ and $j = 13$
   count is incremented
6. for $i = 13$ and $j = 15$
   count is incremented

and value of $j = 16$ and condition in while loop is not satisfied it come out from the while loop.

Value of count is 6 it incremented 6 times so value return by this function is 6.

QUESTION ANALYTICS +

Q. 26

The minimum number of comparisons required to find the 65th smallest element in a minheap is equal to ________.

FAQ Have any Doubt ?

● 2080 Correct Option

Solution :
2080

For 1st smallest (root) → 0 comparisons

2nd smallest element → 1 comparisons

3rd smallest smallest → 2 comparisons

4th smallest element → 3 comparisons

⋮

65th smallest element → 64 comparisons

Total number of comparisons = [1 + 2 + 3 + ... + 64]

$$= \frac{(64)(65)}{2} = (32)(65)$$

$$= 2080 \text{ comparisons}$$

QUESTION ANALYTICS +

Q. 27

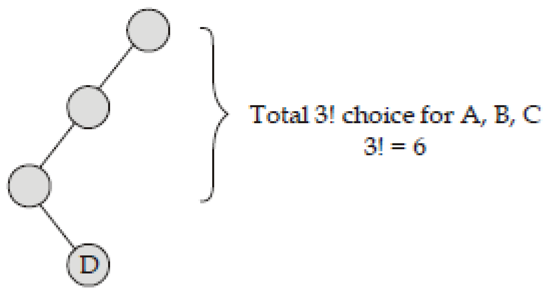
Four vertices {A, B, C, D} is given which have only vertex D as a leaf total number of binary trees possible when every binary tree has four node ________.

Have any Doubt ?

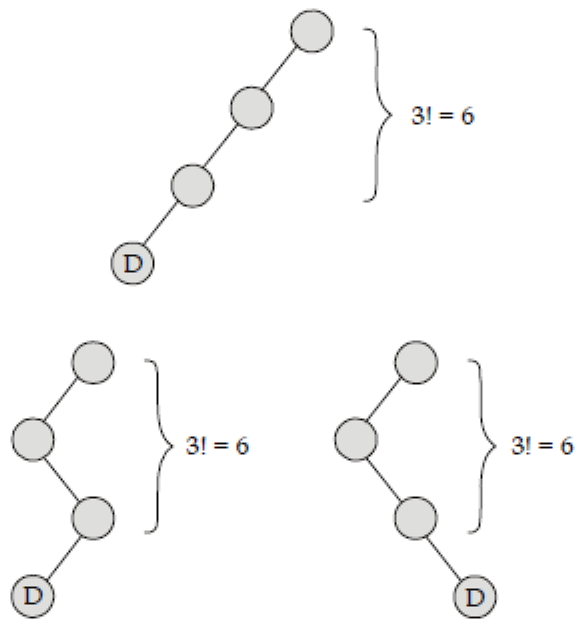
● 48 Correct Option

Solution :
48

When we fix vertex D as leaf



Similarly,



Total 24 choice when D as only leaf in left subtree and no leaf in right subtree, similarly 24 choice when vertex D as only leaf in right subtree.

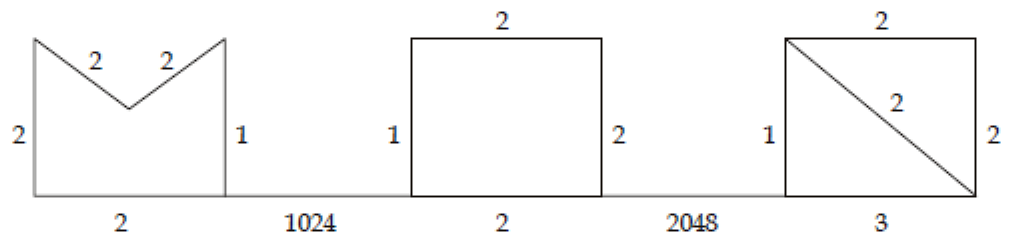
Total 24 + 24 = 48 labelled binary trees are possible.

QUESTION ANALYTICS +

QUESTION ANALYTICS

Q. 28

The number of MSTs for the graph given below are ________.

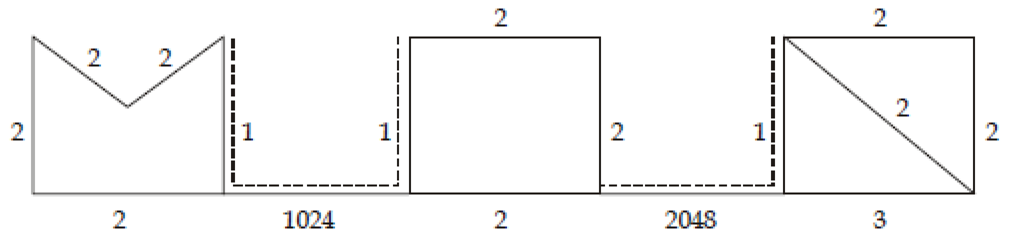


Graph 'H'

Have any Doubt ?

● 36 Correct Option

Solution :
36



[Dotted edge are those included in every MST]

$$\begin{aligned}\text{Number of MSTs} &= ({}^4C_3 \times {}^3C_2 \times {}^3C_2)\\ &= (4 \times 3 \times 3)\\ &= 36\end{aligned}$$

QUESTION ANALYTICS +

Q. 29

Consider the given C functions:

```
int g(int x)
{
    if (x = = 0)
        return 1;
    return g(x – 1) + f(x – 1);
}
    int f(int x)
{
    if (x = = 0)
    return 2;
    return f(x – 1) + g(x – 1);
}
```

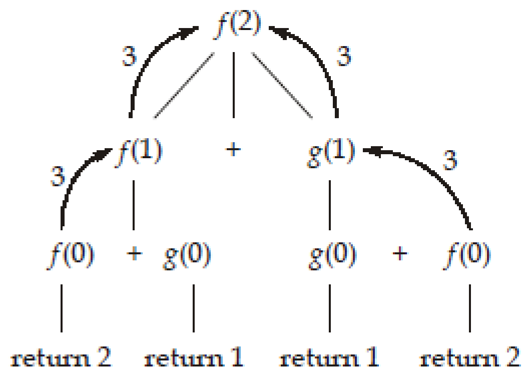
What is value return by $g(f(2))$ ________.

Have any Doubt ?

● 96 Correct Option

Solution :
96

When $g(f(2))$ is called
First $f(2)$ is calculated



$f(2)$ return 6 now $g(6)$ is called
Similarly table of $f(N)$ and $g(N)$ values

N	0	1	2	3	4	5	6
g	1	3	6	12	24	48	96
f	2	3	6	12	24	48	96

$$\begin{aligned}\text{In general for } x \ge 1 \quad g(x) &= f(x)\\ &= 3 \times 2^{x-1}\\ \text{We conclude } g(f(2)) &= g(6) = 96\end{aligned}$$

QUESTION ANALYTICS +

Q. 30

Consider the following array A. Quicksort is run on the array A and assume that the algorithm picks the first element as pivot. In how many ways can the elements present in the array be arranged so that the effect of first pass of quicksort algorithm is preserved?
A = (60, 50, 100, 35, 33, 72, 85)

Have any Doubt ?

● 36 Correct Option

Solution :
36

After running first pass of quicksort, the elements will get divided into partitions.

$$\overset{3!\text{ arrangements}}{\underbrace{35, 33, 50,}_{\downarrow\ \le 60}} \;\Big|\; 60 \;\Big|\; \overset{3!\text{ arrangements}}{\underbrace{100, 72, 85}_{\downarrow\ > 60}}$$

(Number of arrangements preserving first pass of quicksort)

$$= 3!\,3!$$
$$= 36$$

QUESTION ANALYTICS +

Q. 31

Consider a hash table with 'n' slots that uses chaining for collision resolution, table is initially empty. What is the probability that after 4 keys are inserted then atleast a chain of size 3 is created when the value of n = 9 ________. (Upto 3 decimal places)

Have any Doubt ?

● 0.012 (0.012 - 0.013) Correct Option

Solution :
0.012 (0.012 - 0.013)

Probability of inserting first key in any of the n slots of empty table = 1

$$\text{Probability of chain size 3} = 1\times\frac{1}{n}\times\frac{1}{n}\times\frac{n-1}{n}=\frac{n-1}{n^3}$$

$$\text{Probability of chain size 4} = 1\times\frac{1}{n}\times\frac{1}{n}\times\frac{1}{n}=\frac{1}{n^3}$$

$$\text{Probability of atleast a chain size 3} = \frac{n-1}{n^3}+\frac{1}{n^3}=\frac{1}{n^2}$$

For $n = 9$

$$= \frac{1}{81}=0.0123$$

QUESTION ANALYTICS +

Q. 32

Consider the Knapsack instance below:

Capacity of Knapsack = 15
Number of objects = $7(x_1, x_2, x_3 \,.....\, x_7)$
Profits $(p_1, p_2 \,.....\, p_3) = (10, 5, 15, 7, 6, 18, 3)$
Weights $(w_1, w_2 \,.....\, w_7) = (2, 3, 5, 7, 1, 4, 1)$

If Knapsack problem is solved using maximum profit per unit weight then the object number which is partially placed in the Knapsack (for ex. if x_2 is the answer, then fill 2 as the answer, if x_3 is the answer then fill 3) is ________.

Have any Doubt ?

● 2 Correct Option

Solution :
2

$\frac{p_i}{w_i}$ ratio ⇒ (5, 1.67, 3, 1, 6, 4.5, 3)

In decreasing order of $\frac{p_i}{w_i} \Rightarrow (x_5, x_1, x_6, x_3, x_7, x_2, x_4)$

Now if we go on including the objects in the above order, we will see that x_2 is partially placed in the Knapsack.
Hence 2 is the answer.

QUESTION ANALYTICS +

Q. 33

Consider the following program:

```
# include <stdio.h>
void S(int * x, int * y)
{
     static int * temp;
         temp = x;
         x = y;
         y = temp;
}
void find ( )
{
     static int k = 0, a = – 8, b = 0;
         while (k < = 6)
     {
         if ((k++)% 2 = = 1)
             continue;
         a = a + k + 2;
         b = b + k – 3;
     }
     S(&a, &b);
}
int main ( )
{
     find ();
     return 0;
}
```

When the above program is executed value of (a + 8) – b is ________.

Have any Doubt ?

● 20 Correct Option

Solution :
20

function S( ) does not actually swap two variables, rather just swap their address in variable x and y.
Find ( ) function when executed

for $k = 0$

$a = -8 + 1 + 2 = -5$

$b = 0 + 1 - 3 = -2$

for $k = 1$

No effect on a and b

for $k = 2$

$a = -5 + 3 + 2 = 0$

$b = -2 + 3 - 3 = -2$

for $k = 3$

No effect on a and b

for $k = 4$

$a = 0 + 5 + 2 = 7$

$b = -2 + 5 - 3 = 0$

for $k = 5$

No effect on a and b

for $k = 6$

$a = 7 + 7 + 2 = 16$

$b = 0 + 7 - 3 = 4$

for $k = 7$ come out from the while loop

So value of $(a + 8) - b$ is $16 + 8 - 4 = 20$

Answer is 20.

QUESTION ANALYTICS +