

# PROGRAMMING AND DATA STRUCTURES

## SOLUTIONS

1. Consider the following declaration.

```
struct
{
    short P[10];
    union
    {
        short a;
        float b;
        long z;
    } u;
} t;
```

Assume that objects of the type short, float and long occupy 2 bytes, 4 bytes and 8 bytes respectively. The memory requirements for variable 't' and 'u' ignoring alignment considerations respectively are \_\_\_\_\_.

- |            |            |
|------------|------------|
| (a) 34, 14 | (b) 28, 14 |
| (c) 28, 8  | (d) 20, 14 |

**Solution:** Option (c)

**Explanation:**

(c)

Here structure creates the memory for array and union, but union creates the memory for only 'long z' which is maximum among all data types in union.

$$u = \max(2, 4, 8) = 8$$
$$\therefore t = 20 + 8 = 28$$

2. The function delete (head, element) is used to delete a node from the linked list by finding the node value with a given element. The parameter head is the first node of the list. Find the missing statements A and B in the following "delete" function to delete the node? (Assume all

elements are distinct in the list and the function returns pointers that point to the first node of the list).

Node delete (Node head, int element)

```
{
    Node x = head;
    if (x.data == element) return head.next;
    while (x.next != NULL)
    {
        if (____ A ____ )
        {
            ____ B ____;
            return head;
        }
        x = x.next;
    }
}
```

(a) ? : ?. ????? == ????????? ? : ?. ????? == ?. ?????. ????? (b)

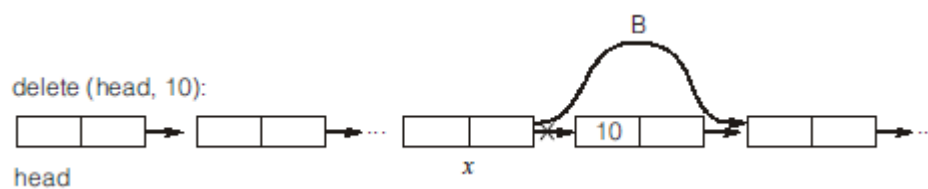
?: ?. ?????. ????? == ????????? ? : ?. ????? == ?. ?????. ?????

(c) ? : ?. ????? == ????????? ? : ?. ?????. ????? == ?. ????? (d)

?: ?. ?????. ????? == ????????? ? : ?. ?????. ????? == ?. ?????

**Solution:** Option (b)

**Explanation:**



```
?? ( (?. ????? == 10)
    {
        ????? = ?. ?????. ?????;
        return head;
    }
```

3. Consider the following code

```
Node *find (Node * head)
{
    Node * P1 = head, *P2 = head;
    while (P2)
    {
        P1 = P1 → next;
        P2 = (P2 → next)? P2 → next → next : NULL;
    }
    printf ("%d", P1 → value);
}
```

Assume Node is the structure type with two members: 'value' and 'next'. Identify the node value printed by the above code if non-empty linked list header is passed to the function find?

- (a) First element of the list [i.e., value of the first node]
- (b) Second element of the list
- (c) Middle element of the list
- (d) Last element of the list

**Solution:** Option (c)

**Explanation:**

(c)

P1 traverses node by node

P2 traverses by skipping one node. The node pointed by P1 is the middle node in the linked list when P2 reaches to NULL.

∴ Middle element of the list is printed by P1 → value.

4. In delete operation of binary search tree, we need inorder successor (or predecessor) of a node when a node to be deleted where it has both left and right child. Which of the following is true about inorder successor needed in delete operation?

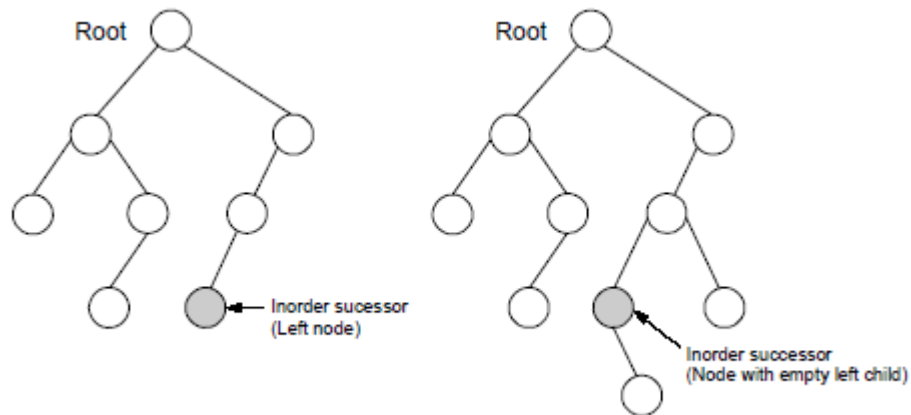
- (a) Inorder successor is always either leaf node or a node with empty right child.
- (b) Inorder successor maybe an ancestor of the node.
- (c) Inorder successor is always a leaf node.
- (d) Inorder successor is always either a leaf node or a node with empty left child.

**Solution:** Option (d)

**Explanation:**

(d)

Successor of Root element is always the smallest element of the Right subtree. Because it will be the next largest element after the element to be deleted.



5. What will be the output printed by the following C program?

```
void main ( )  
{  
    int x = 1, i, y = 2;  
    for (i = 0; i < 5; i++)  
    {  
        x << 1;  
        y = x + i;  
    }  
    printf("%d, %d", x, y);  
}
```

(a) 1, 5

(b) 32, 5

(c) 1, 72

(d) 32, 72

**Solution:** Option (a)

**Explanation:**

(a)

Iterations of for statement										
for:	i = 0		i = 1		i = 2		i = 3		i = 4	
	x	y	x	y	x	y	x	y	x	y
x << 1;	1	2	1	1	1	2	1	3	1	4
y = x + i;	1	1	1	2	1	3	1	4	1	5

x = 1, y = 5

[**Note:** x << 1 will not change the value of x, but x = x << 1 will change the value of x]

6. Consider the following function declaration

int\* f(int \*);

Which of the following is correct about the declaration?

- (a) f is a function which takes integer pointer as argument and returns integer.
- (b) f is a function which takes integer pointer as an argument and returns address of an integer.
- (c) f is a pointer to a function which takes integer pointer as an argument and returns integer.
- (d) f is a pointer to a function which takes integer pointer as an argument and returns address of an integer.

**Solution:** Option (b)

**Explanation:**

(b)

The correct declaration for (a) is int f (int \*)

The correct declaration for (b) is int\* f(int \*);

The correct declaration for (c) is int (\*f) f<sub>2</sub>(int \*)

The correct declaration for (d) is int \*(\*f) fun(int \*)

7. Which of the following is NOT true about linked list implementation of queue?

- (a) In enqueue operation, if new nodes are inserted at the beginning of linked list, then in dequeue operation nodes must be removed from end.
- (b) In enqueue operation, if new nodes are inserted at the end, then in dequeue nodes must be removed from the beginning.

- (c) Both (a) and (b)
- (d) None of the above

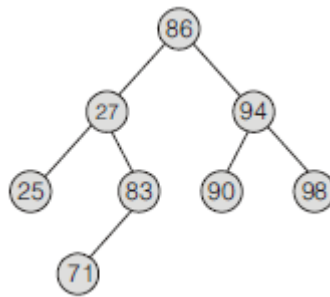
**Solution:** Option (d)

**Explanation:**

(d)

Both (a) and (b) are true to keep the first in first out order, a queue can be implemented using linked list in any of the given two ways.

**8.** Consider the following AVL tree.



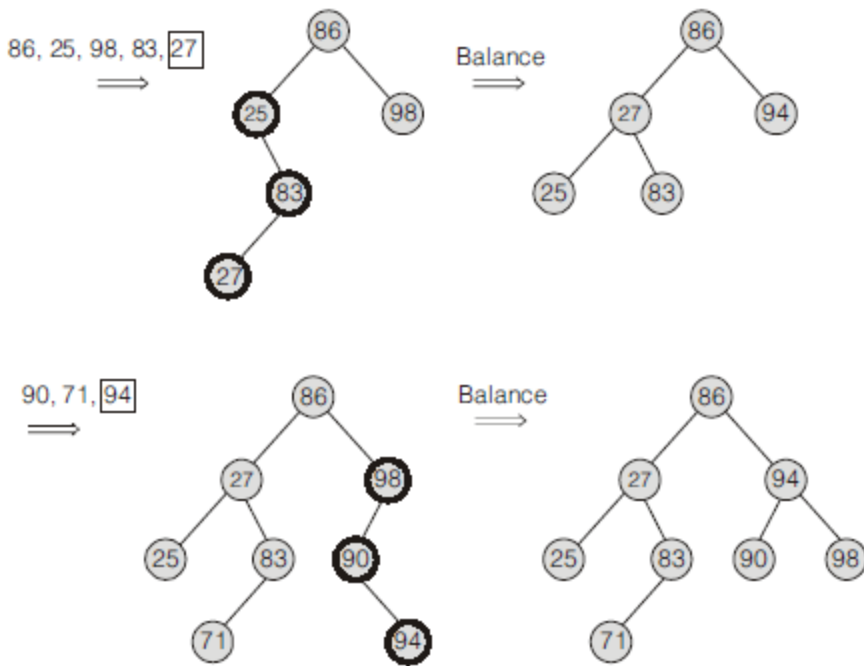
Which of the following order of elements are inserted into an empty AVL tree, so that it is possible to get the above AVL tree?

- (a) 94, 71, 86, 25, 98, 83, 27, 90
- (b) 98, 94, 90, 83, 86, 25, 71, 94
- (c) 86, 25, 98, 83, 27, 90, 71, 94
- (d) None of these

**Solution:** Option (c)

**Explanation:**

(c)



The order: 86, 25, 98, 83, 27, 90, 71, 94 will result the given AVL  
[Note: Option (a) and Option (b) will generate different AVL trees]

9. Find the output of the following program.

```
main ()  
{  
    extern int i ;  
    i = 20;  
    printf ("%d", i);  
}
```

- (a) Linked error  
(c) Compiler error

- (b) 20  
(d) None of these

**Solution:** Option (a)

**Explanation:**

(a)

Linked error: Undefined symbol-i Extern int i; Specifies to the compiler that the memory for i is allocated in some other program and that address will be given to the current program at the time of linking. But linker finds that no other variable of name 'i' is available in any other program with memory space allocated for it. Hence linker error occurred.

**10.** Consider the following procedure struct.

```
gate (root)
{
    if (root == null) return 0;
    if (root → leftchild == null && root → rightchild == null) return 1;
    else return (maximum (gate(root → leftchild), gate(root → rightchild)) + 1);
}
```

What is the functionality of above function?

- (a) Returns the height of binary tree
- (b) Returns number of leaf nodes in the binary tree
- (c) Returns number of levels in the binary tree
- (d) None of these

**Solution:** Option (c)

**Explanation:**

(c)

Given function returns the number of levels of input binary tree.

**11.** A 3-ary tree is a tree in which every internal node has exactly 3 children. The number of leaf nodes in such a tree with 19 internal nodes will be \_\_\_\_\_.


**Solution:** 39

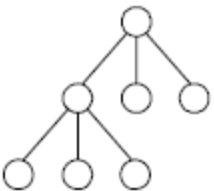
**Explanation:**


39



$n \rightarrow$  number of internal nodes :

Let  $n = 1$    $\Rightarrow 3 \Rightarrow 2(1 - 1) + 3$

Let  $n = 2$    $\Rightarrow 5 \Rightarrow 2(2 - 1) + 3$

Let  $n = 3$    $\Rightarrow 7 \Rightarrow 2(3 - 1) + 3$

Number of internal nodes  $= 2(n - 1) + 3 = 2(19 - 1) + 3 = 39$

**12.** Consider the following code.

```
void main( )
{
    static int i = 5;
    if ( --i )
    {
        main ( );
        printf( "%d", i );
    }
}
```

The number of zero's printed in the output are \_\_\_\_\_.

**Solution:** 4

**Explanation:**

4

The variable 'i' is declared as static, hence memory for 'i' will be allocated for only once, as it encounters the statement. The function main ( ) will be called recursively unless i becomes equal

to zero and since main ( ) is recursively called, so the value of static i, i.e. 0 will be printed every time the control is returned. So total 4 times zero is printed.

**13.** Consider a two-dimensional array with elements stored in the form of lower triangular matrix. The elements must be crossed to read A[4, 2] from the array A[-6, ..., +8, -6, ..., +8] whose base address 1000 is \_\_\_\_\_. (Assume elements are stored in row major order).

**Solution:** 1063

**Explanation:**

1063

The given lower triangular matrix can be represented as

$$\begin{array}{c}
 \begin{array}{cccccccc}
 & -6 & -5 & -4 & \dots & \dots & \dots & +8 \\
 -6 & a_{11} & & & & & & \\
 -5 & a_{21} & a_{22} & & & & & \\
 -4 & a_{31} & a_{32} & a_{33} & & & & \\
 . & . & . & & & & & \\
 . & . & . & & & & & \\
 . & . & . & & & & & \\
 . & . & . & & & & & \\
 +8 & a_{81} & a_{82} & \dots & \dots & \dots & \dots & a_{88}
 \end{array}
 \end{array}$$

Let (i, j) be the element to be accessed.

We must cross upto (i - 1)<sup>th</sup> row.

Number of elements upto (i - 1)<sup>th</sup> row or 10<sup>th</sup> row

$$\begin{aligned}
 &= 1 + 2 + 3 + \dots + [(i - 1) - (l_{bi}) + 1] \quad [l_{bi} \rightarrow \text{lower bound of } i] \\
 &= 1 + 2 + 3 + \dots (3 - (-6) + 1) \\
 &= 1 + 2 + 3 + \dots + (10) \\
 &= \frac{10 \times 11}{2} = 55
 \end{aligned}$$

In i<sup>th</sup> row we must cross (j - l<sub>bj</sub>) elements.

[l<sub>bj</sub> → lower bound of j]

$$= 2 - (-6) = 8$$

∴ In total = 55 + 8 = 63 elements need to be crossed

Resulted address = Base address + Number of element crossed

$$1000 + 63 = 1063$$

**14.** Consider the following program

```
int main ( )
{
    char *str = "Gate2015"
    printf ("%d", ravindra (str));
    return 0;
}

int ravindra (char *P1)
{
    char *P2 = P1;
    while (*++P1);
    return (P1 - P2);
}
```

The output of the above program will be \_\_\_\_\_.

**Solution:** 8

**Explanation:**

8

The function counts number of characters in the input string. Gate2015 has 8 characters. P1 points to null character and P2 points to first character, at the end of while loop.

**15.** The sum of the minimum and maximum number of nodes in the AVL tree of height 5 is \_\_\_\_\_. (Assume root node is present at height zero)

**Solution:** 83

**Explanation:**

83

Number of (minimum) nodes =  $S(h - 1) + S(h - 2) + 1$

Number of (maximum) nodes =  $2^h + 1 - 1$

**16.** Consider the following program.

```
int main ( )
{
    int x = 016;
    printf ("%d", x);
    return 0;
}
```

The output of the above program will be \_\_\_\_\_.

**Solution:** 14

**Explanation:**

14

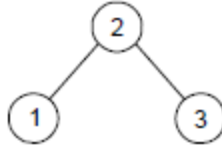
When a constant value starts with 0, it is considered as octal number.

$$(16)_8 = (14)_{10}$$

**17.** Consider the following program.

```
void find (struct Node *node)
{
    struct Node *ptr, *q;
    if (node == NULL) return;
    find (node → left);
    find (node → right);
    ptr = node → left;
    node → left = newNode (node → data);
    node → left → left = ptr;
}
```

If the root of following tree is passed to the above function, what is the level order traversal of output tree produced by above function? (newNode is a function which creates new node)



(a) 2 2 3 3 1 1

(b) 2 2 3 1 3 1

(c) 2 3 2 3 1 1

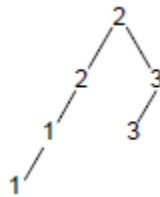
(d) 2 3 2 3 2 1

**Solution:** Option (b)

**Explanation:**

(b)

Tree after execution of above code



Level order traversal is 2 2 3 1 3 1.

**18.** Which of the following is correct output for the program code given below?

```

main ( )
{
    void pr( );
    pr ( );
    pr ( );
    pr ( );
}

void pr ( )
{
    static int i = 1;
    printf ("%c", (65 + i++));
}
  
```

(a) 66, 67, 68

(b) 66, 66, 66

(c) 67, 68, 69

(d) None of these

**Solution:** Option (d)

**Explanation:**

(d)

The correct output is “BCD” when the function pr ( ) is first called the value of i is initialized to 1. After the pr ( ) completes its execution i = 2 is retained for it’s next call as “i” is static variable.

∴  $65 + 1 = 66$  (B)

$65 + 2 = 67$  (C)

$65 + 3 = 68$  (D)

∴ BCD is the correct output.

**19.** Which of the following are equivalent to the statement?

$\text{int } k = (i \ll 3) + (j \gg 2)$

(a)  $\text{int } k = i * 8 + j / 4;$

(b)  $\text{int } k = i * 3 + j * 2;$

(c)  $\text{int } k = i * 3 + j / 2;$

(d)  $\text{int } k = i / 8 + j * 4;$

**Solution:** Option (a)

**Explanation:**

(a)

$\ll$  and  $\gg$  are bitwise operators used to multiply and divide by power of 2 respectively (shift operators)

∴  $i \ll 3 \Rightarrow i * 8$

$j \gg 2 \Rightarrow j / 4$

**20.** Consider the following foo function and identify the return value of foo function.

```
int foo (unsigned int n)
{
```

```

int c, x = 0;
while (n != 0)
{
    if (n & 01) x++;
    n >>= 1;
}
return c;
}

```

- (a) It counts the total number of bits set in an unsigned integer.
- (b) It counts the number of bits which are zero.
- (c) It counts the number of occurrences of 01.
- (d) It returns the same value as 'n'.

**Solution:** Option (a)

**Explanation:**

(a)

It counts the number of bits set in an unsigned integer.

```

while (n != 0)
{
    if (n & 01) x++;          /* performs bitwise AND operator and if condition is
                              satisfied if result contains atleast one 1.
    n >>= 1
}

```

x++; Maintains the count for number of 1's

n >>= 1 Shift the 'n' bit number by 1 bit to right.

**21.** Which of the following are the number of assignments, number of additions and number of subtractions respectively required for swapping 2 variables without the help of 3<sup>rd</sup> variable?

- (a) 3, 2, 2
- (b) 3, 1, 2
- (c) 3, 3, 2
- (d) 2, 2, 2

**Solution:** Option (b)

**Explanation:**

(b)

The following is the process for swapping two variables i and j without 3<sup>rd</sup> variable.

i = i + j  
j = i - j  
i = i - j;

∴ 3 assignments, 2 subtractions, 1 addition.

**22.** Consider the AVL tree T in which left subtree contains half of the maximum number of nodes possible in the balanced AVL tree of height h and right subtree consists of one 3<sup>rd</sup> of the maximum number of nodes possible in AVL tree of height 'h'.

Assume that tree T may or may not be height balanced at present. What is the total maximum possible number of nodes in T?

(a)  $\frac{5}{6}(2^{h+1} - 1) - 1$

(b)  $\frac{5}{6}(2^{h+1} - 1) + 1$

(c)  $\frac{3}{2}(2^{h+1} + 1)$

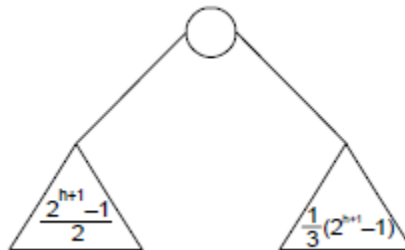
(d)  $\frac{3}{2}(2^{h+1} + 1) - 1$

**Solution:** Option (b)

**Explanation:**

(b)

Maximum nodes in AVL tree of height h =  $2^{h+1} - 1$



$$\begin{aligned}\therefore \text{Total nodes} &= \left(\frac{2^{h+1} - 1}{2}\right) + \frac{1}{3}(2^{h+1} - 1) + 1 \\ &= \frac{5}{6}(2^{h+1} - 1) + 1\end{aligned}$$



**23.** The minimum size that an array may require to store a binary tree with 'n' nodes is \_\_\_\_\_.

(a)  $2^{\lceil \log_2 2(n+1) \rceil} - 1$

(b)  $2^n - 1$

(c)  $2^n - n + 1$

(d)  $n + 1$

**Solution:** Option (a)

**Explanation:**

(a)

In case of full or complete binary tree minimum height  $\Rightarrow h_{\min} = \lceil \log_2 (n+1) \rceil$

Hence, last element will be stored at  $2^{h_{\min}} - 1$

$$\therefore 2^{\lceil \log_2 2(n+1) \rceil} - 1$$

**24.** Consider the following program.

```
variable l;
procedure My (K: integer)
begin
    K = K + 1;
    Print (K);
end

procedure R ( )
var l;
begin
    l = 5;
    My (l);
    print (l);
end;
begin
    l = 3;
    My (l);
    R ( );
    print (l);
```

end

Find the output produced by above program using dynamic scoping, and all functions uses call by value.

(a) 4, 6, 6, 4

(b) 4, 6, 5, 3

(c) 4, 5, 6, 4

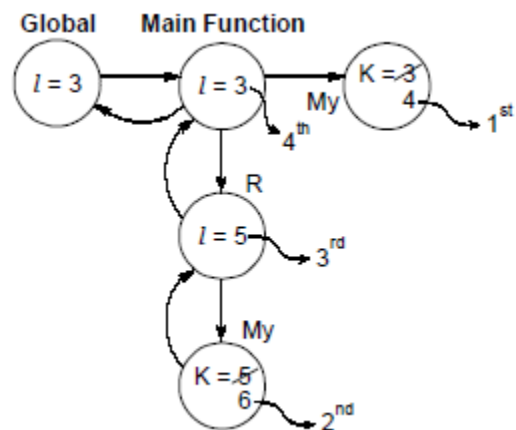
(d) 3, 6, 6, 3

**Solution:** Option (b)

**Explanation:**

(b)

The program uses dynamic scoping. So if the variable is not available in the present function, then it searches back in the previous function scope from where it was called.



Hence the output is 4, 6, 5 and 3 printed.

**25.** Consider the following C program.

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

void fun (struct listnode *head)
{
```

```

    if (head == NULL || head → next == NULL) return;
    struct listnode *tmp = head → next;
    head → next = tmp → next;
    free (tmp);
    fun (head → next);
}

```

What is the functionality of the above function?

- (a) It reverses the linked list
- (b) It deletes the linked list
- (c) Alternate nodes will be deleted
- (d) It reverses the linked list and delete alternate nodes

**Solution:** Option (c)

**Explanation:**

(c)

The above program deletes every alternate node in the linked list (In particular second, fourth, sixth... soon nodes will be deleted)

**26.** Consider the following function.

```

void f (int n)
{
    if (n ≤ 0) return;
    else
    {
        print (n);
        f (n - 2);
        print (n);
        f (n - 1);
    }
}

```

Let  $f(n)$  be the number of values printed. What is the number of values printed by above function?

(a)  $f(n-1) + f(n-2)$

(c)  $f(n-1) + f(n-2) + 2$

(b)  $f(n-1) + f(n-2) + 1$

(d)  $f(n-2) + f(n-3)$

**Solution:** Option (c)

**Explanation:**

(c)

$f(n-1) + f(n-2) + 2$  values printed by  $f(n)$ , where 2 indicate number of print statements.

**27.** What is the number of additions in the fibonacci series of  $n$  which uses following recursive function.

$$\text{fib}(n) = \text{fib}(n-1) + \text{fib}(n-2)$$

(a)  $f(n-1) + f(n-2)$

(c)  $f(n-1) + f(n-2) + 1$

(b)  $f(n-1) + f(n-2) + n$

(d)  $f(n-1) + f(n-2) + 2n$

**Solution:** Option (c)

**Explanation:**

(c)

Number of additions:  $\text{fib}(n) = \text{fib}(n-1) + \text{fib}(n-2) + 1$

**28.** Consider the following code.

```
int x = 0, i;  
for (i = 0; i < 10, i++)  
    if (i%2 && x++)  
        x += 2;
```

What will be the value of  $x$ ?

(a) 11

(c) 15

(b) 13

(d) 17

**Solution:** Option (c)

**Explanation:**

(c)

(15)

$$i = 1 \Rightarrow x = 3$$

$$i = 3 \Rightarrow x = 6$$

$$i = 5 \Rightarrow x = 9$$

$$i = 7 \Rightarrow x = 12$$

$$i = 9 \Rightarrow x = 15$$

**29.** Consider the following infix expression which is to be converted to postfix expression using stack.

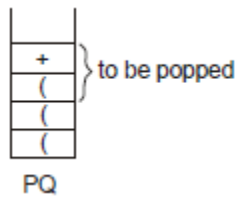
$$(((P + Q) * (R + S)) / T) + (A * (B + C))$$

The sum of all unique possible heights of stack when converting from infix to postfix is \_\_\_\_\_.

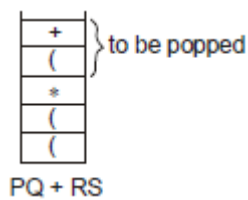
**Solution:** 15

**Explanation:**

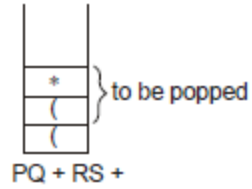
15



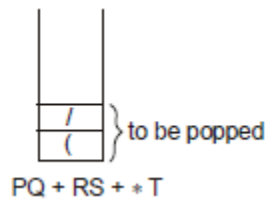
Now ')' comes in Infix exp, pop up to first occurrence of '('



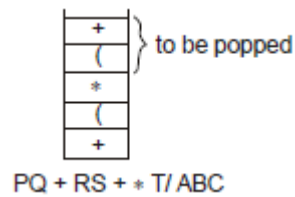
Now ')' comes in Infix expression pop upto occurrence of '('.



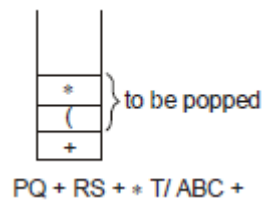
Again ')' comes pop till first '(' comes



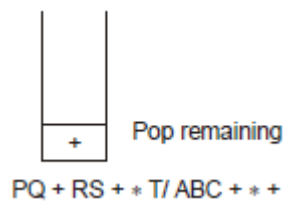
Now ')' comes pop all elements



Now, '(' comes, pop upto '('



Now ')' comes pop until '(' comes maximum possible stack height is 5



$PQ + RS + * T/ABC + * \Rightarrow PQ + RS + *T/ABC + * +$  in the postfix expression

$\therefore$  The sum of all unique possible heights =  $1 + 2 + 3 + 4 + 5 = 15$

**30.** The sum of outputs printed by running the following program is \_\_\_\_\_.

```
int main ( )
{
    int i;
    for (i = 3; i <= 6; i++)
        printf(“%d”, f1(i));
}

int f1 (int n)
{
    if (n < 2) return n;
    else return (f1(n - 1) + f2(n - 2));
}

int f2 (int n)
{
    if (n <= 1) return n;
    else return (2 * f1 (n - 2) + 1);
}
```

**Solution:** 20

**Explanation:**

20

$$f1(3) = 2$$

$$f1(4) = 3$$

$$f1(5) = 6$$

$$f1(6) = 9$$

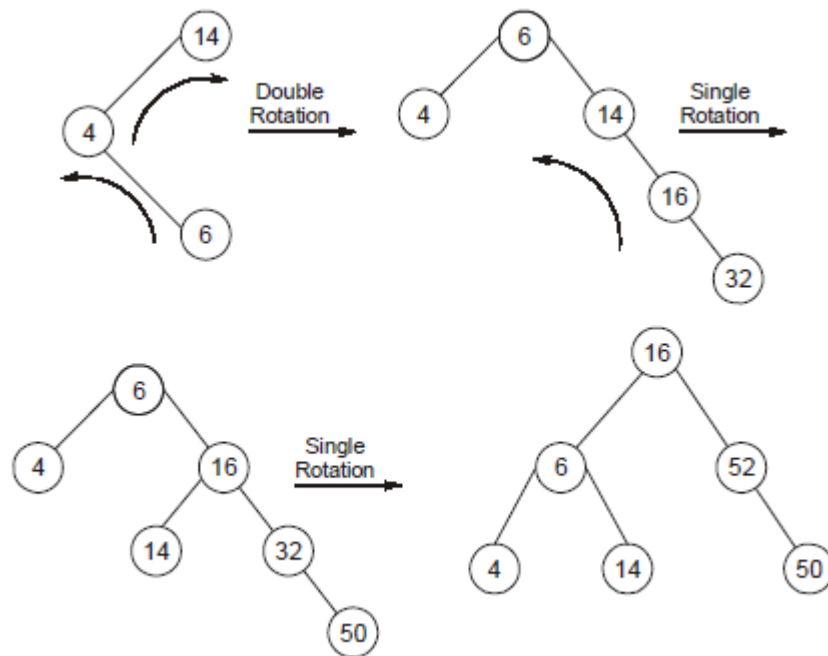
$$2 + 3 + 6 + 9 = 20$$

**31.** The key 14, 4, 6, 16, 32, 50 in the order are inserted into an initially empty AVL tree. The total numbers of rotations to make AVL with the given keys are \_\_\_\_\_. Assume “single rotation = 1 rotation” and “double rotation = 1 rotation”.

**Solution:** 3

**Explanation:**

3



One double and two single rotations are required. So total 3 rotations.

**32.** Consider the following postorder and Inorder traversals of binary tree.

Post order: 1, 2, 5, 4, 7, 6, 3, 9, 11, 10, 8

In order: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11

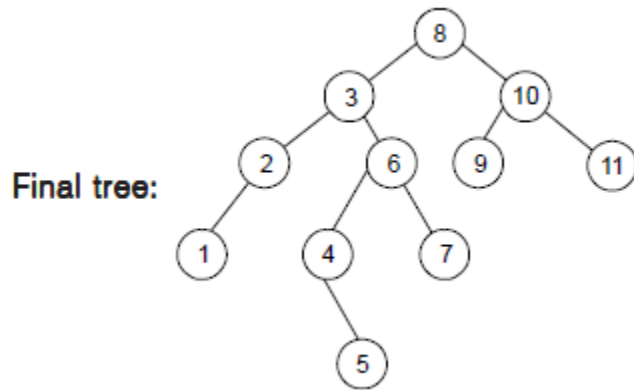
The total number of nodes that have height greater than the height of node 6 are \_\_\_\_\_.  
Assume root is at lowest height.

**Solution:** 4

**Explanation:**

4





1, 4, 5 and 7 have height greater than node 6.

**33.** Consider the following recursive function.

```

int g(int e)
{
    if (e > 4)
        return (2 + g(e - 5) + g(e - 2));
    return 1;
}
  
```

The value returned from the  $g(15)$  is \_\_\_\_\_.

**Solution:** 40

**Explanation:**

40

$$g(15) = 2 + g(10) + g(13)$$

$$g(10) = 13 \quad g(13) = 25$$

$$\therefore g(15) = 2 + 13 + 25 = 40.$$

Also for this kind of question, instead of computing recursively, by seeing the recurrence we can go bottom up, like in this case  $g(0) = g(1) = g(2) = g(3) = g(4) = 0$ .

Now  $g(5) = 2 + g(0) + g(3) = 2$ ;  $g(6) = 2 + g(1) + g(4) = 2$ ... and so on we can build till  $g(15)$

**34.** A priority queue can efficiently implemented using which of the following data structures?  
Assume that the number of insert and peek (operation to see the current highest priority item)

and extraction (remove the highest priority item) operations are almost same.

- (a) Array
- (b) Linked List
- (c) Heap Data Structures like Binary Heap, Fibonacci Heap
- (d) None of the above

**Solution:** Option (c )

**35.** A Priority-Queue is implemented as a Max-Heap. Initially, it has 5 elements. The level-order traversal of the heap is given below:

10, 8, 5, 3, 2

Two new elements '1' and '7' are inserted in the heap in that order. The level-order traversal of the heap after the insertion of the elements is:

- (a) 10, 8, 7, 5, 3, 2, 1
- (b) 10, 8, 7, 2, 3, 1, 5
- (c) 10, 8, 7, 1, 2, 3, 5
- (d) 10, 8, 7, 3, 2, 1, 5

**Solution:** Option (d)