CLASS TEST

S.No. : 01 **BS_CS_B_290919**

Programming and Data Structures

MADE EASY

India's Best Institute for IES, GATE & PSUs

Delhi | Noida | Bhopal | Hyderabad | Jaipur | Lucknow | Indore | Pune | Bhubaneswar | Kolkata | Patna

Web: www.madeeasy.in | **E-mail:** info@madeeasy.in | **Ph:** 011-45124612

CLASS TEST 2019-2020

COMPUTER SCIENCE & IT

Date of Test : 29/09/2019

ANSWER KEY ➤ Programming and Data structures

1. (c)
2. (c)
3. (d)
4. (a)
5. (d)
6. (c)
7. (b)
8. (a)
9. (a)
10. (a)
11. (b)
12. (d)
13. (a)
14. (c)
15. (c)
16. (b)
17. (d)
18. (b)
19. (a)
20. (c)
21. (a)
22. (b)
23. (d)
24. (c)
25. (b)
26. (b)
27. (d)
28. (b)
29. (b)
30. (b)

DETAILED EXPLANATIONS

1. (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

$$u = \max(2, 4, 8) = 8$$

$$\therefore \quad t = 20 + 8 = 28$$

2. (c)

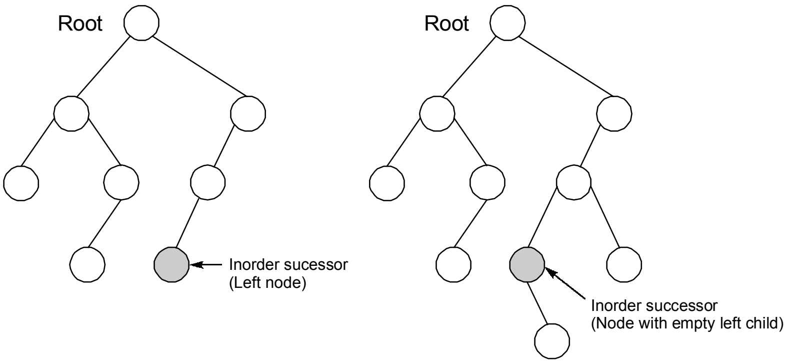
P1 traverses node by node

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.

3. (d)

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



4. (a)

Iterations of for statement

	$i = 0$		$i = 1$		$i = 2$		$i = 3$		$i = 4$	
for:	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

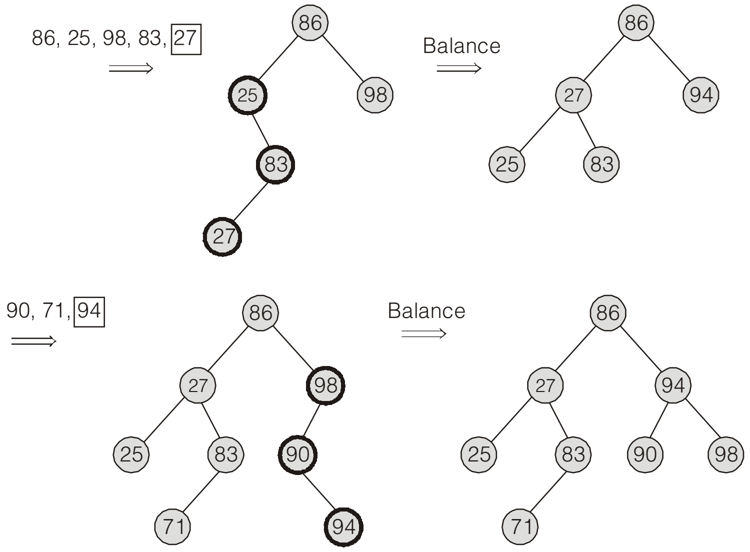
$x = 1, \; y = 5$

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

5. (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.

6. (c)



The order: 86, 25, 98, 83, 27, 90, 71, 94 will result the given AVL

[**Note:** Option (a) and obtion (b) will generate different AVL trees]

7. (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_2 (int ∗)

The correct declaration for (d) is int ∗(∗f) fun (int ∗)

8. (a)

Linker 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.

9. (a)

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

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

$$\therefore \qquad \text{Minimum size } = 2^{\lceil \log_2(n+1) \rceil} - 1$$

10. (a)

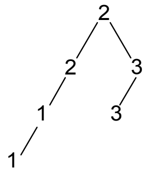
For n = 3

The number of function calls are $(2^{n+1} - 1) = 15$

The number of moves are $(2^{n} - 1) = 7$

11. (b)

Tree after execution of above code



Level order traversal is 2 2 3 1 3 1.

12. (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 it execution $i = 2$ is retained for it’s next call as “i” is static variable.

$\therefore$ $65 + 1 = 66$ (B)

$65 + 2 = 67$(C)

$65 + 3 = 68$(D)

$\therefore$ BCD is the correct output.

13. (a)

<< and >> are bitwise operators used to multiply and divide by power of 2 respectively (shift operators)

$\therefore$ $i << 3 \Rightarrow i * 8$

$j >> 2 \Rightarrow j / 4$

14. (c)

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

15. (c)

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

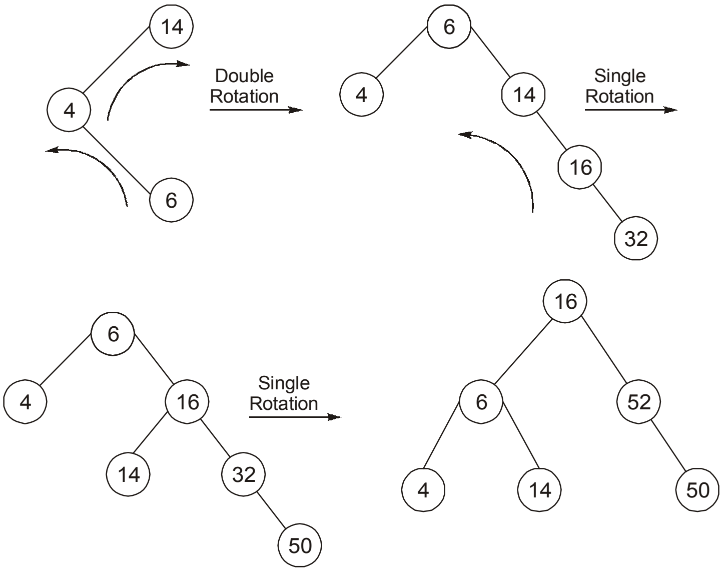
16. (b)

This program finds a repeated number (duplicate) in the list. This program adds all the values in the variable “value” and subtract “k(k–1)/2” from it. Array has 0 to (k – 1) locations and numbers are from 1 to (k – 1). So one number will be repeated in the array.

17. (d)

$$f1(3) = 2$$
$$f1(4) = 3$$
$$f1(5) = 6$$
$$f1(6) = 9$$
$$2 + 3 + 6 + 9 = 20$$

18. (b)



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

19. (a)

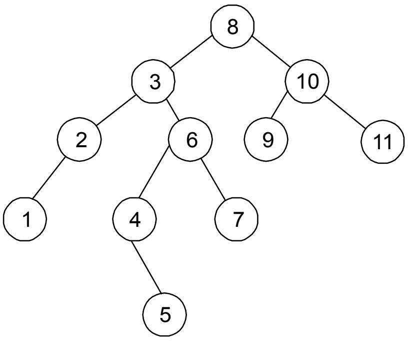
g(15) = 2 + g(10) + g(13)

$$\left.\begin{array}{l} g(10) = 13 \\ g(13) = 25 \end{array}\right\} \text{Compute recursively}$$

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

20. (c)

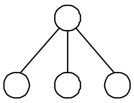
Final tree:

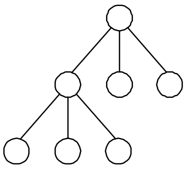


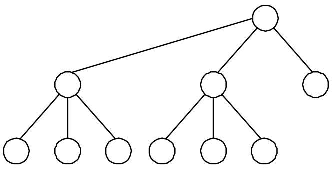
1, 4, 5 and 7 have height greater then node 6.

21. (a)

n → 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

22. (b)

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.

23. (d)

The given lower triangular matrix can be represented as

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

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

We must cross upto $(i - 1)^{\text{th}}$ row.

Number of elements upto $(i - 1)^{\text{th}}$ row or 10^{th} row

$$= 1 + 2 + 3 + \ldots\ldots.. + [(i - 1) - (l_{bi}) + 1][l_{bi} \rightarrow \text{lower bound of } i]$$

$$= 1 + 2 + 3 + \ldots\ldots.. (3 - (-6) + 1)$$

$$= 1 + 2 + 3 + \ldots\ldots. + (10)$$

$$= \frac{10 \times 11}{2} = 55$$

In i^{th} row we must cross $(j - l_{bj})$ elements. $[l_{bj} \rightarrow \text{lower bound of j}]$

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

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

Resulted address = Base address + Number of element crossed

1000 + 63 = 1063

24. (c)

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

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

25. (b)

The above code returns the sum of the nodes storing even values in the tree. It traverses the tree recursively, and add even values while returning.

52 + 76 + 60 + 70 + 14 = 272

26. (b)

The while loop may some time leads to segmentation error. Segmentation error occurs when we access data from an address that does not exist. This can happen when we reach last node and assign the NULL value to temp in while loop.

So again when we access temp → data. This creates segmentation error.

```
While(temp→data!=y && temp → next! = NULL).
{
        q=temp;
          temp=temp →next;
}
```

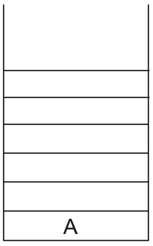
The above code is correct implementation of while Loop.

27. (d)

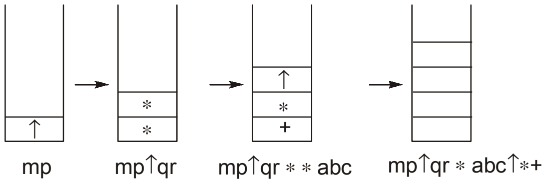
The minimum set of operation are

$$\left.\begin{array}{l}\text{Push A}\\ \text{emit}\\ \text{Push } B\\ \text{emit}\\ \text{Push C}\\ \text{emit}\\ \text{Push A}\\ \text{emit}\\ \text{POP}\\ \text{emit}\\ \text{POP}\\ \text{emit}\\ \text{POP}\\ \text{emit}\end{array}\right\}\text{14 operations}$$

After the above operations final content of stack is



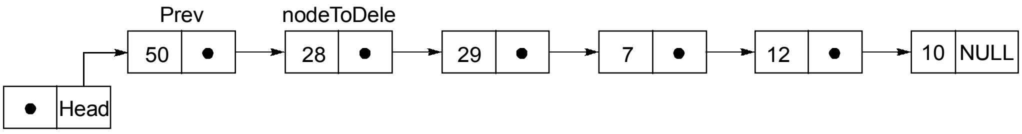
28. (b)



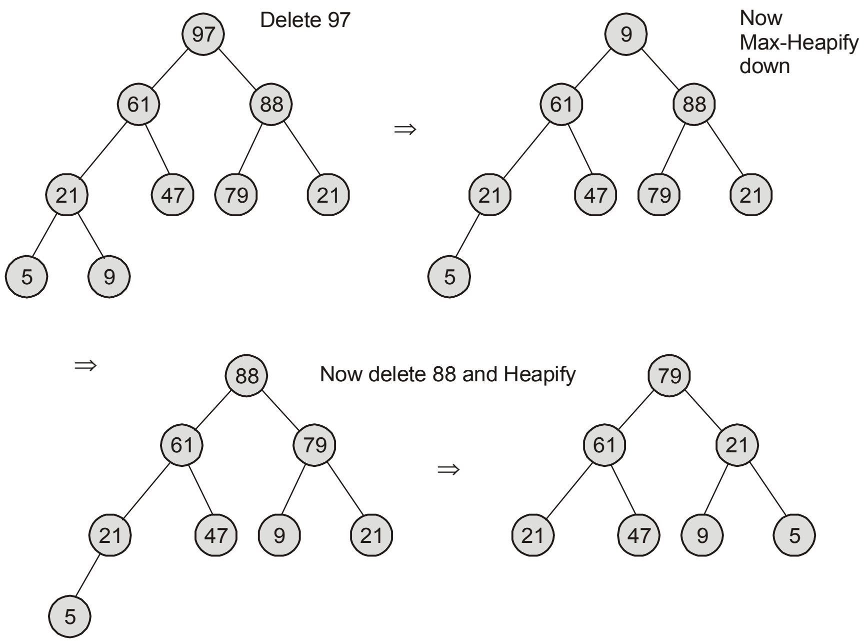
max size of stack = 3

29. (b)

The elements 28 is inserted between 50 and 29. The new list after the code is executed is shown below.



30. (b)



∴ 79, 61, 21, 21, 47, 9, 5