CLASS TEST

S.No. : 01SK_CS_ALL_221021

MADE EASY
MADE EASY
India's Best Institute for IES, GATE & PSUs

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

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

ALGORITHMS

COMPUTER SCIENCE & IT

Date of Test : 22/10/2021

ANSWER KEY

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

DETAILED EXPLANATIONS

1. (a)

(I) $-1 \le \sin n \le 1 \Rightarrow \sin n = O(1)$

$\Rightarrow \sin n \le C \,.\, 1$ for atleast some positive value of C. Which is true.

(II) $-1 \le \cos n \le 1 \quad \cos n = \Omega(1)$

$\cos n > C \,.\, 1$ and C is positive.

This fails if cos n < O.

Hence (I) is correct and (II) is false.

2. (a)

$$\text{Suppose} \qquad n = \left.2^{2^{2^{2}}}\right]\text{500 times}$$

$$\log^* n = 500$$

$$\log n = \left.2^{2^{2^{\cdot^{\cdot^{\cdot}}}}}\right]\text{499 times}$$

$$\log(\log^* n) = \log 500 = g(n)$$

$$\log^*(\log n) = 499 = f(n)$$

$$f(n) > g(n)$$

$$\Rightarrow \qquad g(n) = O(f(n))$$

3. (b)

(a) Conquer step is recursively call the subproblems and obtain their respective solutions.

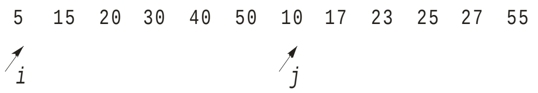
(b) Merge sort has combine step (merge algorithm)

(c) Quicksort has no combine step.

4. (b)

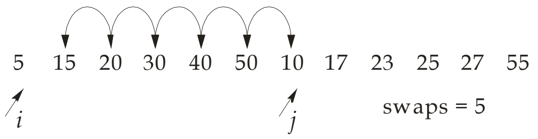
	Stable	Inplace
Selection sort	No	Yes
Insertion sort	Yes	Yes
Bubble sort	Yes	Yes
Heap sort	No	Yes

5. (c)



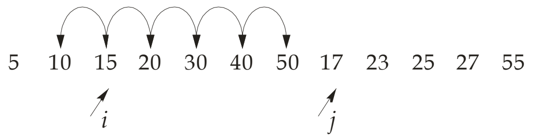
```
5   15   20   30   40   50   10   17   23   25   27   55
i                              j
```

After 1st comparison,



```
5   15   20   30   40   50   10   17   23   25   27   55
i                              j          swaps = 5
```

After 2nd position



____ by proceed

Total swaps after

$$5 \text{ comparisons} = 0 + 5 + 0 + 4 + 0 = 9$$

6. **(c)**

Strassen's reduced divide and conquer approach to 7 multiplications and 18 additions.

Hence, time complexity becomes

$$T(n) = 7T\left(\frac{n}{2}\right) + 18\left(\frac{n}{2}\right)^2$$

7. **(c)**

1. Calculate medians of both the arrays. Let the medians be m_1 and m_2 [array 1 → m_1, array 2 → m_2]

(b) If $m_1 < m_2$ then recursively find median of first half of array 2 and second half of array 1.

(c) If $m_1 > m_2$ then recursively first median of first half of array 1 and second half of array 2.

Since every time half of the total elements are discarded, time complexity is $\theta(\log n)$

8. **(a)**

Items	I_1	I_2	I_3	I_4	I_5
W_i	20	40	5	10	30
V_i	100	160	30	20	90
$P_i = V_i/W_i$	5.0	4.0	6.0	2.0	3.0

Arranging items with decreasing value of P_i.

Items	I_3	I_1	I_2	I_5	I_4
W_i	5	20	40	30	10
V_i	30	100	160	90	20
$P_i = V_i/W_i$	6.0	5.0	4.0	3.0	2.0

Items chosen : I_3, I_1, a fraction of I_2

Weight of I_2 to be added = 60 – 5 – 20 = 35

40 units of I_2 has 160 value

$$35 \text{ units of } I_2 = \frac{35 \times 160}{40} = 140 \text{ value}$$

$$\text{Total profit} = 30 + 100 + 140 = 270$$

9. **(c)**

Since, array is sorted, hence binary search can be used to find the location for the newly inserted element. Then all the elements from that location till n will be shifted.

Hence, it takes $O(n)$ time.

11. **(a)**

Big Oh and Big omega are reflexive.
Big Oh and Big omega are not symmetric.
Big Oh, Big omega and Big theta are transitive.
Big theta is symmetric but big Oh and Big omega are not symmetric.

12. **(c)**

Taking log for all functions
$n^{\log n} \log n$, $n^2 \log n$, $n \log n \log \log n$, $(\log n)^3$, $n(\log n)^2$

13. **(b)**

For $n > 10000$ $\quad f(n) = n^{2+\cos n}$
$\quad -1 < \cos n < 1$
$\quad n < f(n) < n^3$
$g(n) = n^{|\sin n|} \Rightarrow O < |\sin n| < 1$
$\Rightarrow \quad n^{\circ} < g(n) < n^1$

14. **(d)**

When the code enters the third for loop i is initialized to 1 and goes till n^2. Then check for 2nd for loop is performed $i \le n^3$ and i is incremented to $n^2 + 1$. Again it enters the third loop and i is initialized to 1. Hence this leads to an infinite loop.

15. **(b)**

i independent run $\rightarrow O(n^2)$
$j = n^3 \rightarrow k \rightarrow \log_3 n^3$ times
$j = (n^3)^{1/3} \rightarrow k \rightarrow \log n$ times and soon
$\log n^3 + \log n + \log n^{1/3} + \log n^{1/3^2}$ ....
$= \log [n^3 \times n \times n^{1/3} \times n^{1/3^2} .....] = O(\log_3 n)$

$$\text{So total time complexity} = O(n^2 \log n) \times O(n)$$
$$= \theta(n^3 \log n)$$

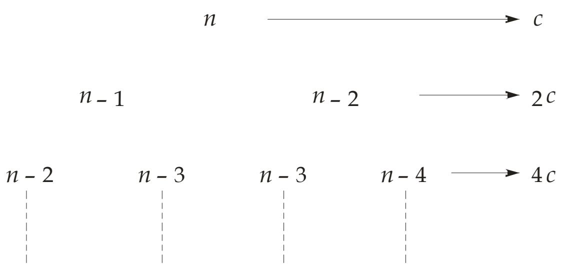
So, (b) is correct.

16. **(d)**

For worst case, consider the following recurrence relation,

$$T(n) = T(n-1) + T(n-2) + C$$

where C is a constant.
Using recursive tree method,



worst case height = n
So worst case time complexity = $c + 2c + 2^2c + \ldots\ldots\ldots\ldots\ldots\ldots 2^n c$

$$= c[\,1 + 2 + 2^2 + \ldots\ldots\ldots\ldots\ldots\ldots 2^n] = c\,\frac{2^{n+1}-1}{2-1} = \theta(2^n)$$

Note : $n * T(n-1) \rightarrow$ [multiplication takes constant time]

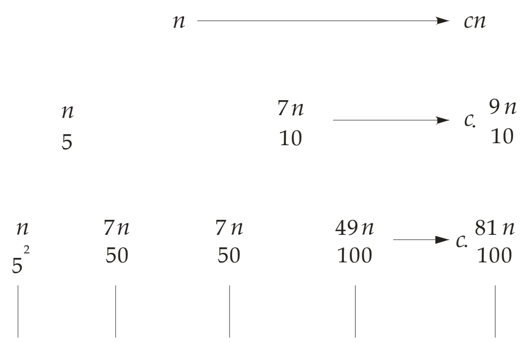
17. (c)

$$T(n) = 4T(n/3) + n^2 \log n$$
$$n^{\log_3 9} = n^{\log_3 4} < n^2$$

Now, $n^2 \log n$ if approximated to a lesser value say n^2, then also it is bigger than $n^{\log_3 4}$

So, $T(n) = \theta(n^2 \log n)$ [by using Master's theorem]

18. (a)



$$T(n) = cn + \frac{9cn}{10} + \frac{81c}{100} + \ldots\ldots\ldots\ldots$$

$$= cn\left[1 + \frac{9}{10} + \left(\frac{9}{10}\right)^2 + \left(\frac{9}{10}\right)^3 + \ldots\ldots\ldots\left(\frac{9}{10}\right)^k\right]$$

$$\left(\frac{7}{10}\right)^k n = 1 \Rightarrow \left(\frac{10}{7}\right)^k = n \Rightarrow k = \log_{10/7} n$$

$$T(n) = cn\left[\frac{\left(\frac{9}{10}\right)^{k+1} - 1}{\frac{9}{10} - 1}\right] = O(n) \qquad \text{(decreasing GP series)}$$

© Copyright: MADE EASY

19. **(a)**

```
power (a, n)
    { if (n = = 1)
        return (1 * a);
    else
    {
    mid = n/2;
    value = power (a, mid);
    value = value * value; // 1 multiplication
    }
    return value;
}
```

Recurrence relation for multiplication

$$T(n) = T(n/2) + 1; T(1) = 1; T(2) = 1$$

So, $n/2k$, or $k = \log_2 n$

$$T(n) = \log_2 n + 1$$

Since return value is $(1 * a)$ so 1 more multiplication involved.

OR

Compute by putting some value of n like 2^{10} → 11 multiplications.

20. **(a)**

For least comparisons arrays can be

1	2	3	4	5	6	7

8	9	10	11	12	13

I - compare the last position of the first array.

II - and the first position of the second array.

if I < II then copy first array and then second array.

1 comparison done.

21. **(c)**

First 2 4's normal sequence.

Next 2 4's reversed sequence.

So after every 4 4's, original sequence is restored.

So, till 312 4's original sequence.

0 - 9 → 2 4's

10 - 19 → 2 4's

and soon.

1st 4 → 5 index

5th 4 → 25 index

9th 4 → 45 index

13th 4 → 65 index

⋮

313th 4 → 1565 index

314th 4 → 1569 index

315th 4 → 1570 index [reversed]

22. **(b)**

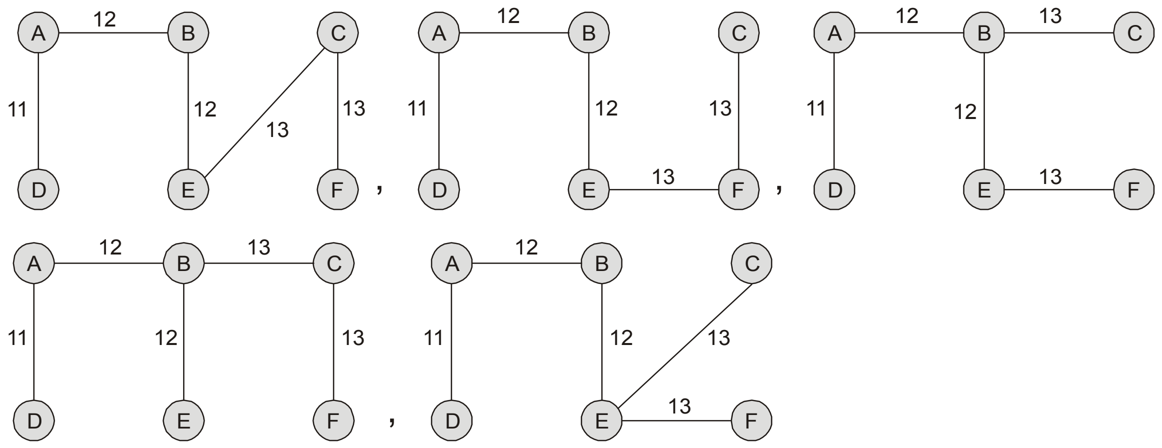
			2	6			10					
	6	9	20	15	8	10	20	15	18	17		20
	20	15	6	9	18	17	2	6	8	10	12	10
i = –1	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	pivot
	j = 0	*j*	*j*	*j*	*j*	*j*	*j*	*j*	*j*	*j*	*j*	
	i	*i*	*i*	*i*	*i*	*i*						

(6, 9, 2, 6, 8, 10), 10, (15, 18, 17, 12, 20)

23. **(b)**

Options (b) is false. Best case swaps for bubble sort is O when the array is already sorted in increasing order.

24. **(b)**



All above trees MST with cost: 11 + 12 + 12 + 13 + 13 = 61

25. **(c)**

Consider the following execution of Dijkstra's algorithm :

	P	Q	R	S	T	U	V	W
P	0	1	3	∞	∞	∞	∞	∞
Q	0	1	3	9	4	18	∞	∞
R	0	1	3	9	4	18	∞	∞
T	0	1	3	9	4	8	∞	∞
U	0	1	3	9	4	8	∞	13
S	0	1	3	9	4	8	11	13
V	0	1	3	9	4	8	11	13
W	0	1	3	9	4	8	11	13

The path calculated using Dijkstra's algorithm from S to V is 11.
But as per the given figure path 'S' to 'V' has '2' as the minimum weight,

$$P\xrightarrow{1}Q\xrightarrow{3}T\xrightarrow{4}U\xrightarrow{5}W\xrightarrow{-11}V\ .$$

This path will be correctly calculated if edge [W-V] is relaxed before edge [Q-S]. For that, the path from S to P should be greater than 13. Hence [13 to 20] all weights are possible, giving '8' possible values.

The second way which is possible is, we reach vertex 'V' from 'P' using path [Q-S], but for that, the path [P → Q → S → V] should be less than or equal to 2. Let the path of Q to S be '*z*'.

It can be written as, $1 + z + 2 \leq 2$

$\Rightarrow \quad z + 3 \leq 2$

$\Rightarrow \quad z \leq -1$

Hence the values ranging from [–1 to –20] are also possible where both (–1) and (–20) are inclusive giving 20 different values

So, Total possible values $= 8 + 20$

$= 28$

26. **(b)**

```
(1) for (i = 1; i ≤ n; ++ i)
    {
(2) for (j = 1; j ≤ i * i; ++ j)
    {

    If ((j % i) = = 0)

(3) for (k = 1; k ≤ j; ++ k)
    {
    c = c + 1;
    }
    }
}
```

(1) $\Rightarrow$ Run for n times

(2) $\Rightarrow$ Run for n^3 times

i.e. $1^2 + 2^2 + 3^2 + 4^2 \ldots\ldots\ldots\ n^2$

$\Rightarrow \dfrac{n(n+1)(2n+1)}{6} = O(n^3)$

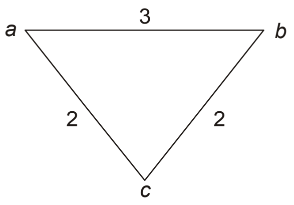
(3) $\Rightarrow$ Run for $O(n)$ times

$\therefore$ Overall Time $= n[O(n^3) + O(n)]$

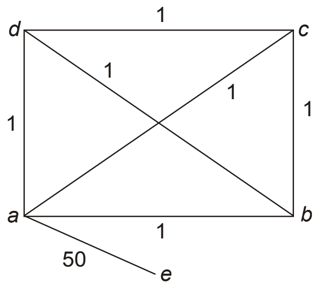
$= O(n^4)$

28. **(d)**

I. MST contain *ac* and *bc* but not contain *ab*, which is the shortest path between *a* and *b*



II. We may be forced to select the edges with weight much higher than average



$$\text{Average weight} = \frac{50+6}{7} = 8$$

Expected MST weight $= 4 \times 8 = 32$

Actual MST weight $= 50 + 6 = 56$

29. (c)

The Huffman tree for this distribution is :

Codeword

1	x_1	0.49	0.49	0.49	0.49	0.49	0.51	1
00	x_2	0.26	0.26	0.26	0.26	0.26	0.49	
011	x_3	0.12	0.12	0.12	0.13	0.25		
01000	x_4	0.04	0.05	0.08	0.12			
01001	x_5	0.04	0.04	0.05				
01010	x_6	0.03	0.04					
01011	x_7	0.02						

30. (c)

$5 \times 0.02 + 5 \times 0.04 + 5 \times 0.04 + 5 \times 0.04 + 3 \times 0.12 + 2 \times 0.26 + 1 \times 0.49 = 0.10 + 0.20 + 0.20 + 0.20 + 0.36 + 0.52 + 0.49 = 2.07$

© Copyright: MADE EASY