MADE EASY
Leading Institute for IES, GATE & PSUs
Delhi | Bhopal | Hyderabad | Jaipur | Pune
Web: www.madeeasy.in | E-mail: info@madeeasy.in | Ph: 011-45124612

ALGORITHMS

COMPUTER SCIENCE & IT

Date of Test : 07/08/2026

ANSWER KEY ➤

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

DETAILED EXPLANATIONS

1. (d)

Selection sort in worst and best case take same time i.e., $O(n^2)$. So all the input take same time.

2. (b)

Before merging blindly, we have to sort both array individually which will take $O(m \log m)$ and $O(n \log n)$ time respectively. Then merging will take $O(m + n)$ in worst case.

Total number of comparisions = $m \log m + n \log n + m + n$

$= O(m \log m + n \log n)$

3. (a)

$$T_1(n) + T_2(n) = O(F(n))$$

4. (d)

$$T(n) = 2T\left(\frac{n}{\sqrt{2}}\right) + n$$

Using Master's theorem, $a = 2$, $b = \sqrt{2}$, $f(n) = n$

$$n^{\log_b a} = n^{\log_{\sqrt{2}} 2} = n^{\left(\frac{\log 2}{\log \sqrt{2}}\right)}$$

$$= \frac{1}{n^{(1/2)}} = n^2$$

$\Rightarrow \quad n^{\log_b a} = n^2$

$\Rightarrow \quad n^2 > f(n)$

$\Rightarrow T(n)$ is $O(n^2)$

5. (a)

The worst case of the sort arises, when all the elements fit in to the single bucket. Now, on applying sorting on that particular bucket, if Insertion sort or Bubble sort is used, it will result in $O(n^2)$ time.

Heap sort is not a stable algorithm.

Worst case behaviour of merge sort is O(n log n).

Hence, merge sort is the correct answer.

6. (c)

7. (b)

Consider the following array,

1	6	3	2	5	4	7
0	1	2	3	4	5	6

Here the element at array index, 0, 2, 4 and 6 are already in their correct position. Applying merge sort on the array, will take $O(n \log n)$ time.

8. **(b)**

9. **(c)**

The time complexity of the Huffman coding is O(n log n). Using heap tree to store weight of each node. Each iteration required O(log n) time to determine the small weight node and insert the new weight there are n iteration one for each item.

10. **(b)**

$$W = 8 \text{ (capacity)}$$

Feasible solutions:

(i) $\{I_1, I_3, I_4\}$,

(ii) $\{I_2, I_3\}$

Profit of $\{I_1, I_3, I_4\}$ is 23

profit of $\{I_2, I_3\}$ is 15

Optimal solution is $\{I_1, I_3, I_4\}$ with capacity of 8 and maximum profit 23 produced.

$\therefore$ I_2 is not selected in the solution.

11. **(b)**

$$T(n) = T(n-1) + 1/n \Rightarrow \Theta(\log n)$$

$$T(n) = 2T(n/2) + n / \log n \Rightarrow \Theta(n \log \log n)$$

$$T(n) = \sqrt{n} \cdot T(\sqrt{n}) + n \Rightarrow \Theta(n \log \log n)$$

$$T(n) = 4T(n/2) + n^2\sqrt{n} \Rightarrow \Theta(n^{2.5})$$

12. **(d)**

To build max-heap tree number of comparision combine every 2 element pair from n to 1.

$$\frac{n}{2} + \frac{n}{4} + \frac{n}{8} + \frac{n}{16} \cdots \frac{n}{2^{\log n}} = \text{(n) comparision.}$$

n comparision to find 1st largest element, for next two elements it take O(log n) time.

So, total comparision n + O(log n).

13. **(d)**

Number of leaf nodes = n

Let internal nodes be K

$\therefore$ Total nodes = K + n

For m-ary tree, number of leaf nodes with K internal nodes = (m – 1) K + 1

$\therefore$ (m – 1) K + 1 = n

$$\therefore \quad K = \frac{n-1}{m-1}$$

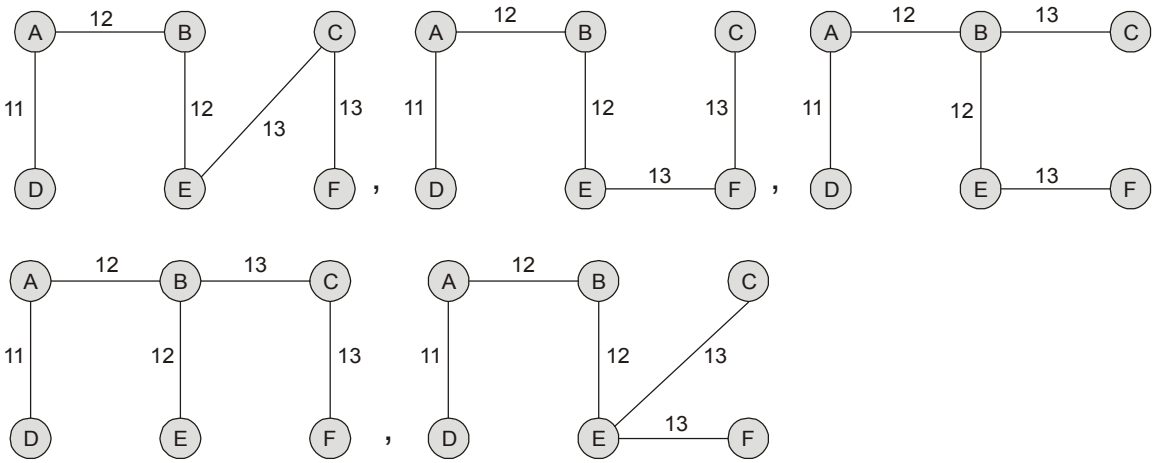
$\therefore$ Total number of nodes

$$= \frac{n-1}{m-1} + n = \frac{n-1+n(m-1)}{m-1}$$

$$= \frac{n-1+nm-n}{m-1} = \frac{nm-1}{m-1}$$

© Copyright: MADE EASY

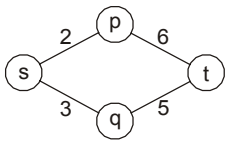
14. (b)



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

15. (b)

Even if no two edges have the same weight, there could be two paths having the same weight.



To find the shortest path between s and t.

$s \to p \to t \Rightarrow 8$

$s \to q \to t \Rightarrow 8$

So, there are two distinct paths.

16. (b)

17. (a)

$$f(n) = O(n),\ g(n) = \Omega(n),\ h(n) = \theta(n)$$

$$g(n) + f(n).h(n) = \Omega(n) + \underbrace{O(n).\theta(n)}$$

$$= \Omega(n) + [\theta(n) \leftrightarrow \theta(n^2)]$$

$$= \Omega(n)$$

18. (a)

19. (c)

$$R^{(k)}[i, j] = R^{(k-1)}[i, j] \ ||\ (R^{(k-1)}[i, k]\ \&\&\ R^{(k-1)}[k, j])$$

Transitive closure of directed graph computed using floyd warshall's algorithm. Which computes using all pairs shortest path algorithm.

$$R = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \end{array},$$

$$R^{(1)} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad R^{(2)} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} (1, 2) \text{ \& } (2, 3) \Rightarrow (1, 3)$$

$$R^{(3)} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, (1, 3) \text{ \& } (3, 4) \Rightarrow (1, 4)$$

$$R^{(4)} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} = \text{Transitive closure of } R$$

20. (c)

Finding minimum edge each time will take (v) time, this will done for ($V - 1$) time. Will take worst case time

$$= O(|V|) \times (V - 1)$$
$$= O(|V|^2)$$

21. (c)

Here Master's theorem is not applicable directly.

$$T(n) = 2.T\left(\sqrt{n}\right) + \log\left(\sqrt{n}\right)$$

put $n = 2^k$

$$T(2^k) = 2.T\left(2^{\frac{k}{2}}\right) + \log_2\left(2^{\frac{k}{2}}\right)$$

$$T(2^k) = 2.T\left(2^{\frac{k}{2}}\right) + \frac{k}{2}$$

put $S(k) = T(2^k)$

$$S(k) = 2.S\left(\frac{k}{2}\right) + \frac{k}{2}$$

Now apply Master's theorem: $S(k) = \Theta(k \log k)$

$$T(2^k) = \Theta(\log n \,.\, \log \log_2 n)$$

© Copyright: MADE EASY

22. **(a)**

Function MHY (A, i, key) represent increase key or decrease key operation which will take log n time to replace the value of key and maintaining the Max heap property.

So, it will take $\Theta(\log n)$ time.

23. **(c)**

$$f(n) = 2^{\log_2 n} = n^{\log_2 2} = n$$
$$g(n) = n^{\log n}$$
$$h(n) = n^{1/\log n} = \sqrt[\log n]{n} \left[n > \sqrt[\log n]{n} \text{ for all large value of } n \right]$$

[It is less than n since max power of n is always less than 1 for large value of n]

So, $g(n) \geq f(n \geq h(n)$

So, $f(n) = O(g(n))$ and $g(n) = \Omega(h(n))$

24. **(b)**

For n time, inner loop will execute for n times.

For $\frac{n}{2}$ time, inner loop will execute for $\frac{n}{2}$ times.

For $\frac{n}{4}$ time, inner loop will execute for $\frac{n}{4}$ times.

and do on .....

So time complexity: $T(n) = O\left(n + \frac{n}{2} + \frac{n}{4} + ... + 1\right)$

$= O(n)$

25. **(b)**

X and Y have LCS's of length 4. The following are longest common subsequences of X and Y.

⟨B, C, B, A⟩

⟨B, D, A, B⟩

Hence (b) is true.

26. **(d)**

A. Matrix chain multiplication : (n^3)

B. Travelling salesman problem : (n^n)

C. 0/1 knapsack : (mn)

D. Fibonacci series : $O(n)$

27 **(b)**

$$\begin{cases} l[i, j], & \text{if } k = 0 \\ \min\left\{d_{i,j}^{k-1},\ d_{i,k}^{k-1} + d_{k,j}^{k-1}\right\}, & \text{if } 1 \leq k \leq n \end{cases}$$

$\therefore$ Option (b) is correct.

28. (b)

$$A_1 A_2 A_3 = A_1 \times (A_2 \times A_3)$$

$3 \times 100, 100 \times 2, 2 \times 2$

By **Person X applying Greedy:**

$A_1 \times (A_2 \times A_3)$

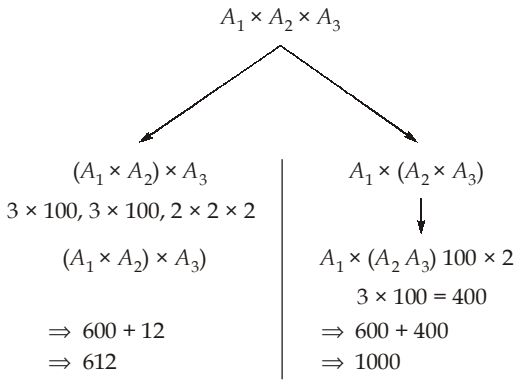
$3 \times 100, 100 \times 2, 2 \times 2$

$(A_2A_3) \rightarrow 100 \times 2, 2 \times 2 = 200 \times 2 = 400$

$A_1 \times (A_2A_3) \rightarrow 3 \times 100, 100 \times 2 = 300 \times 2 = 600$

Total number of multiplication required = 600 + 400 = 1000

Person Y with Dynamic:



Number of multiplication saved by Person Y = 1000 – 612 = 388

29. (c)

In Bellman ford algorithm for every round 'E' edges are relaxed.

Each edge relaxation takes constant time. In general there are (V – 1) rounds excluding the verification round.

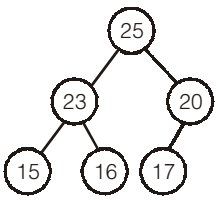
$$\text{Total time} = (V - 1) \times O(E) \times O(1) = O(VE)$$

But for complete graph $E = O(V^2)$

$\therefore$ $\text{Total time} = O(V^3)$

30. (a)

Max heap will be



■■■■

© Copyright: MADE EASY