CLASS TEST

S.No. : 01 SK_CS_A+B_160522

MADE EASY

India's Best Institute for IES, GATE & PSUs

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

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

ALGORITHMS

COMPUTER SCIENCE & IT

Date of Test : 16/05/2022

ANSWER KEY

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

DETAILED EXPLANATIONS

1. (b)

Apply Master Theorem:

$$T(n) = aT(n/b) + f(n)$$

$$f(n) = n^{1/2}$$

and here $a = 2, b = 4$

So, $(n^{\log_b a}) = (n^{\log_4 2})$

Will gives $(n^{1/2})$

So, $f(n) = \theta(n^{1/2})$

So, $T(n) = O(\sqrt{n}\log n)$

2. (d)

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

3. (b)

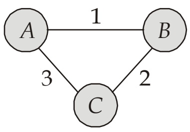
The search using linear probing stops whenever it finds an empty slot.

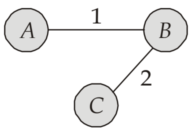
$\therefore$ $\langle e, f, g$ and then empty slot$\rangle$ and $\langle a, b, c$ and the empty slot$\rangle$ are worst case searches.

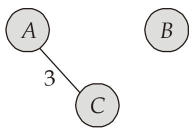
Both searches above has 4 comparisons.

4. (c)

(i) For a directed graph, the absence of back edge in DFS tree means no cycle present. **So false.**

(ii) 

Here 

and 

Two paths are possible but cost is same. **So false.**

(iii) Depth of any vertex in BFS always less than equals to depth of same vertex in DFS. **So true.**

5. (c)

$$g(n) \geq c_1 n,\ h(n) \leq c_2 n$$

$$\therefore \quad g(n) \cdot h(n) \geq c_3 n$$

$$\Rightarrow \quad g(n) \cdot h(n) = \Omega(n)$$

$$f(n) + [g(n) \cdot h(n)] = \max\ (\Omega(n), \Theta(n))$$

$$f(n) + [g(n) \cdot h(n)] = \Omega(n)$$

6. (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 $O(n^2)$

7. **(a)**

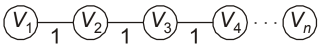
Algorithm random call (A) select last element and compare with all other elements and find largest element and swap with last element same procedure repeated for (n – 1) elements.

At final stage we get sorted array in increasing order (non-decreasing order).

8. **(b)**

Prim's algorithm will pick up the edge with least weight for a particular node, [provided it does not form a cycle] weight of edge (V_{i-1}, V_i) or (V_i, V_{i-1}) = 1

∴ MST will be



∴ Total edge weight = 2 × (n – 1) = 2n – 2.

9. **(a)**

Dijkstra's algorithm will output same as breadth first search on graph and will take O(m + n) time.

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.

∴ I_2 is not selected in the solution.

11. **(b)**

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

3 × 100, 100 × 2, 2 × 2

By **Person X applying Greedy:**

$A_1 \times (A_2 \times A_3)$

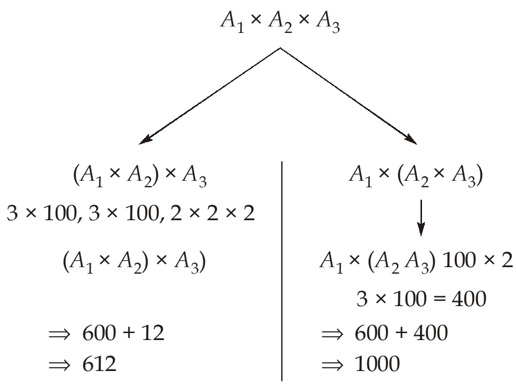
3 × 100, 100 × 2, 2 × 2

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

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

Total number of multiplication required = 600 + 400 = 1000

Person Y with Dynamic:



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

12. (a)

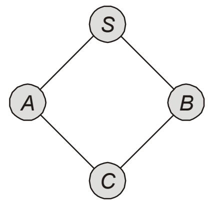
Starting vertex is D

	A	B	C	D	E	F	G	H
D	∞	∞	∞	0_{Nill}	∞	∞	∞	∞
F	16	17_D	13_D	–	∞	3_D	∞	8_D
B	16	7_E	13_D	–	∞	–	∞	8_D
H	16	–	13_D	–	∞	–	12_B	8_D
E	16	–	13_D	–	10_H	–	12_B	
G	16	–	13_D	–	–	–	12_B	
C	16	–	–	–	–	–	–	–
A	–	–	–	–	–	–	–	–

So the order of relaxed the vertices by using Dijkastra's algorithm is DFBHEGCA.

13. (c)

Consider a graph



During the breadth first traversal of the graph. The status of the queue will be as follows :

S	

A	B

$\therefore$ $S - A \rightarrow 1$ edge

$S - B \rightarrow 1$ edge

Difference = 0

B	C

$\therefore$ $S - B \rightarrow 1$ edge

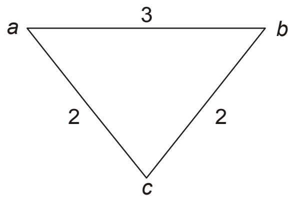
$S - C \rightarrow 2$ edge

Difference = 1

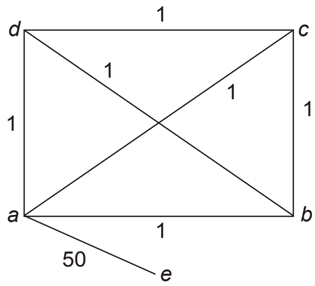
Hence, statements S_1 and S_3 are correct.

14. (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



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

$$\text{Expected MST weight} = 4 \times 8 = 32$$

$$\text{Actual MST weight} = 50 + 6 = 56$$

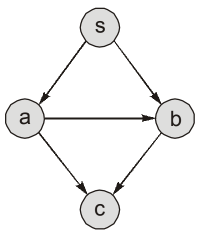
15 (b)

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

∴ Option (b) is correct.

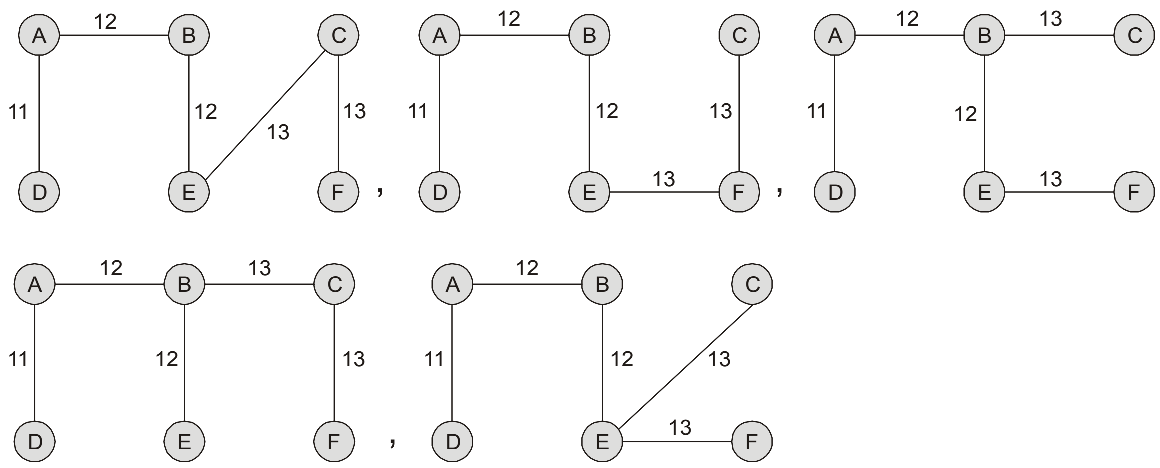
16. (c)

Consider the following graph:



- Vertices 'a' and 'b' both will be on the stack at the same time if at least one of them have an edge on the other vertex. Hence, If there is no directed path from 'a' to 'b' then there exist a directed path from 'b' to 'a'.
- Since, vertices 'a' and 'b' are traversed while performing DFS at vertex 's', so it can be said that there exist directed path from 's' to 'a' and 's' to 'b'. Hence, There exist directed path from 's' to 'a' and directed path from 's' to 'b'.

17. (b)



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

18. (b)

$$S_1 = \sum_{r=0}^{\log n-1} \frac{nr}{2^r} \text{ and } S_2 = \sum_{r=0}^{\log n-1} r2^r$$

$$S_1 = 0 + \frac{n}{2} + \frac{2n}{2^2} + \ldots\ldots + \frac{n \times (\log n - 2)}{2^{(\log n-2)}} + \frac{n \times (\log n - 1)}{2^{(\log n-1)}}$$

$\therefore$ $S_1 \cong \Theta(n)$

$S_2 = 0 + 2^1 + 2 \,.\, 2^2 + \ldots + (\log n - 2)\, 2^{(\log n-2)} + (\log n - 1)\, 2^{(\log n-1)}$

$\therefore$ $S_2 \cong \Theta(n \log n)$

19. (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 $g(n) = \Omega(h(n))$

20. (c)

Time complexity to sort n elements using merge sort = $\Theta(n\log n)$

$$\Theta(n) = \Theta\left(\frac{n}{\log n}\log\frac{n}{\log n}\right)$$

$$\Theta(n) = \Theta\left(\frac{n}{\log n}[\log n - \log\log n]\right)$$

$$\Theta(n) = \Theta\left(\frac{n}{\log n}\log n\right) \quad [\log n - \log\log n = O(\log n)]$$

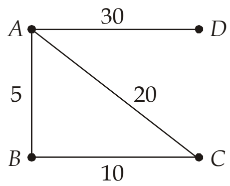
$$\Theta(n) = \Theta(n)$$

21. (a)

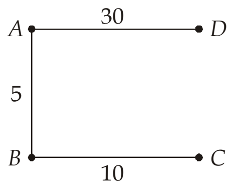
Since edges costs are distinct, so cheapest edge must be present in every minimum spanning tree while expensive edge is may not excluded from every minimum spanning tree.

Statement S_2 is false

Example:



MST will be:



So, most expensive edge is not excluded.

22. (b)

S_1 : Fully sorted array in case of quick sort becomes worst case.
Hence, time taken = $O(n^2)$.

S_2 : Insertion sort on almost sorted array takes $O(n)$ time.

S_3 : Selection sort always takes $O(n^2)$ time.

S_4 : Merge sort always takes $O(n\log n)$ time.

23. (c)

The time complexity is $O(\log n)$ using binary search.

The idea is to go to the middle element at index $n/2$ calculate $a_{n/2} = a + [n/2 - 1] \times d$ and check $a[n/2] = a_{n/2}$ or not if equal check on RHS only otherwise LHS.

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 ....

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

$$= O(n)$$

25. **(a)**

Since by looking through options, we get to know 'b' will be the start vertex.

	a	b	c	d	e	f	g	h
	∞	0	∞	∞	∞	∞	∞	∞
b	5	–	∞	1	3	∞	∞	∞
(b – d) d	5	–	∞	–	3	6	∞	∞
(d – e) e	5	–	4	–	–	6	9	∞
(e – c) c	4	–	–	–	–	6	9	7
(c – a) a	–	–	–	–	–	6	9	7
(d – f) f	–	–	–	–	–	–	9	7
(c – h) h	–	–	–	–	–	–	9	–
(e – g) g	–	–	–	–	–	–	–	–

So, correct sequence will be (b – d), (b – e), (e – c), (c – a), (d – f), (c – h), (e – g).

26. **(a)**

We can find the middle element of the heap, by simply accessing the $n/2^{th}$ element of the array which can be done in O(1) time. Further that element is replaced by the last element of the array in O(1) time. Now, perform the heapify operation which will take $O(\log n)$ time.

$$\text{Time taken} = O(1) + O(1) + O(\log n)$$
$$= O(\log n)$$

27. **(d)**

Prim's algorithm always gives connected whenever a spanning tree is constructed.

(c, e), (e, b), (b, a), (b, d), (b, f)

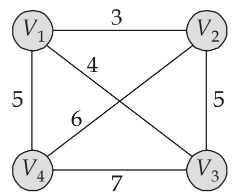
29. **(a)**

If m1 is greater than m2, then median is present in between first element of arr1 to m1 or from m2 to last element of arr2.

If m2 is greater than m1, then median is present in between m1 to last element of arr1 or from first element of arr2 to m2.

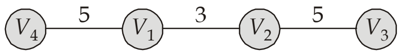
30. (b)

S_1 : Consider the following graph:

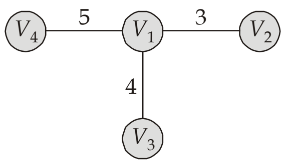


Minimum bottleneck spanning tree:

(A):



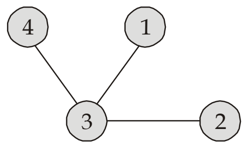
(B):



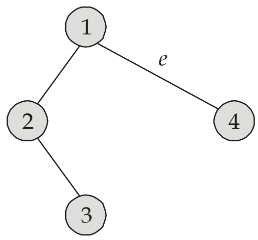
However only (b) is MST therefore S_1 is false.

S_2 : Consider 2 MST's:

(A):



(B):



Where A has a lighter bottleneck edge. This means that B has an edge '*e*' that is heavier than every edge of A. If we include this edge in A, it will form a cycle in which *e* would be heaviest. This contradicts the definition of MST.

Thus, '*e*' cant be present in B, meaning that both A and B have same bottleneck edge.

Thus, S_2 is true.

■■■■