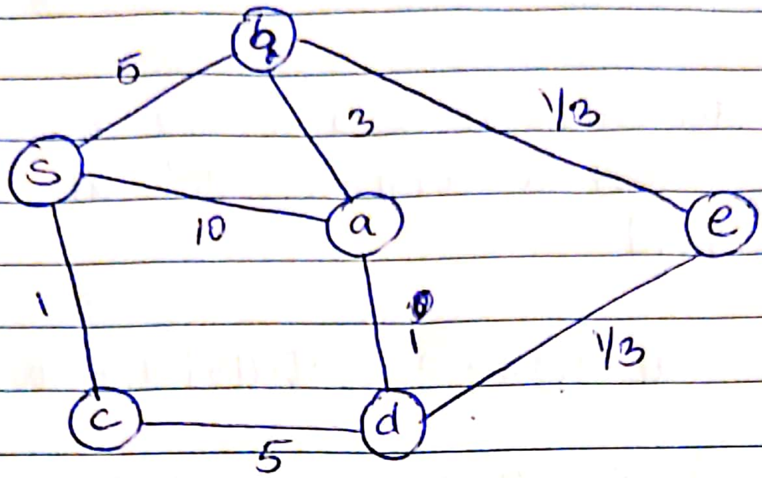
Q1)



For s

$d(a) = 10$
$d(b) = 5$
$d(c) = 1$
$d(d) = \infty$
$d(e) = \infty$

Smallest weight = 1 (c).

: For c Visited(s, c)

$d(a) = 10$
$d(b) = 5$
$d(c) = 1$
$d(d) = 1 + 5 = 6$
$d(e) = \infty$

Smallest weight = 5 (b)

For B Visited (S, C, B).

$d(a) = \min(10, 5+3)$
$= 8.$
$d(b) = 5$
$d(c) = 1$
$d(d) = 6$
$d(e) = 5 + 1/3 = 5.33$

For E visited (S, C, B, E).

$d(a) = 8$
$d(b) = 5$
$d(c) = 1$
$d(d) = \min(6, 5.33 + 0.33)$
$= 5.66.$
$d(e) = 5.33$

For d Visited (S, C, B, E, D)

$d(a) = \min(8, 5.66 + 1)$
$= 6.66.$

$d(b) = 5$
$d(c) = 1$
$d(d) = 5.66$
$d(e) = 5.33$

$\therefore$ dist(a) = 6.66 and dist(d) = 5.66.

CS CamScanner

Current Node	A	B	C	D	E
S	10	5	1	∞	∞
C	10	5	1	6	∞
B	8	5	1	6	16/3
E	8	5	1	17/3	16/3
D	20/3	5	1	17/3	16/3
A	20/3	5	1	17/3	16/3

∴ dist(A) = 10, 8, 20/3 = 6.66

dist(D) = 6, 17/3 = 5.66

CS CamScanner

Q2) Given a 2-D array, we need to arrange the elements such that

$$\sum_{i=1}^{n}\sum_{j=1}^{n}(i+j)\,A[i][j] \quad \text{is maximum.}$$

if we construct a table of 4×4 with elements as $(i+j)$ at $[i][j]$ position

$$\begin{bmatrix} 0 & 1 & 2 & 3 \\ 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 5 \\ 3 & 4 & 5 & 6 \end{bmatrix}$$

Since $a[3][3] = 3+3=6$; $a[2][1] = 2+1=3$.

Thus we need to sort the elements in an increasing order and place them diagonally.

Eg if the elements present are 4, 8, 5, 2 ; the positions must look like the following

$$\begin{bmatrix} 2 & 5 \\ 4 & 8 \end{bmatrix}.$$

Proof:-

Consider the elements in 2-D array are placed diagonally in an increasing sorted order till $a[i][j]$ but $a[i][j+1] < a[i][j]$.

$\therefore$ Total sum $= a[0][0] \times 0 + a[1][0] \times 1 + a[0][1] \times 1 \ldots a[n][n] \times 2n.$

Let totalsum $- \left(a[i][j] \times (i+j) + a[i][j+1] \times (i+j+1)\right) = C$

$\therefore$ Total sum $= C + a[i][j](i+j) + a[i][j+1].(i+j+1)$

Swapping the values

Total sum $= C + a[i][j+1](i+j) + a[i][j].(i+j+1)$

$\therefore$ Difference $= a[i][j](i+j+1-i-j) +$

$a[i][j+1](i+j-i-j-1)$

$= a[i][j] - a[i][j+1]$.

Since $a[i][j] > a[i][j+1]$;

Difference > 0

$\therefore$ Total sum after swapping is greater.
Hence elements must be placed in ascending order diogonlly.

Time Complexity

For sorting the elements, $T(n) = n \log n$

For placing the elements in the matrix, the program will have 3 for loops

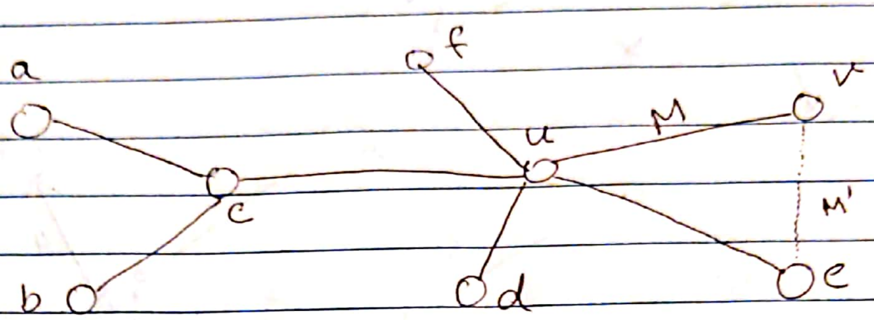
Hence $T(n) = n^3$.

$$\therefore T(n) = n^3 + n\log n$$

$$= n^3$$

CamScanner

Q3) For a graph 'G', let the edge with maximum weight be 'e' with weight 'M'. This node connects vertices 'u' and 'v'. For a minimum spanning tree, it should connect to all vertices.



Let there be a minimum spanning Tree 'T', such that 'M' is a part of this Spanning Tree.

$\therefore$ Total Weight (T_1) = sum of other weights + M

If the edge 'e' is removed from the spanning Tree, there must be another edge e' connecting to vertex 'v' with weight M'. Since connecting e' to the other spanning tree and removing e from the spanning tree still remains a spanning tree.

Total Weight (T_2) = sum of other weights + $M' - M$

$\because$ $M > M'$,

$T_2 < T_1$

But 'T' was considered as the minimum spanning Tree. Hence it is a contradiction.

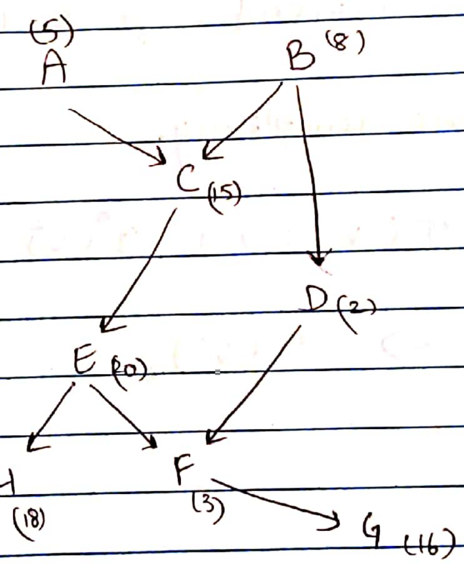
Hence the maximum weight edge will always be excluded from a Minimum spanning Tree.

CS CamScanner

Q4) For this problem we are given that the graph's vertices have weights and we need to find the longest path with increasing weights.

For a graph with V vertices, to place them in topological order, $T(n) = O(V+E)$

For Graph 'G' :-



After Topologically sorting.

:- A B D C E F G H

$\Rightarrow$ [5, 8, 2, 15, 20, 3, 16, 18]

$\therefore$ Longest Increasing routes are

~~5, 8, 15, 20~~
~~5, 8, 16, 18~~
~~2, 3, 16, 18~~
~~2, 15, 16, 18~~

with length 4.

8, 15, 20 (B, C, E)
2, 3, 16 (D, F, G).

Thus we create a table with $T[i][j]$ ~~...~~ = '1' if they are connected from $i \rightarrow j$ and '0' if not.

We then check if ~~...~~ the last element of the computed value is less than the current. If yes we add ~~...~~ 1. ~~...~~ we also ~~...~~ check if the value is not '0' $\because$ Not connected nodes.

$\therefore$ $T[0][n]$ gives the result.

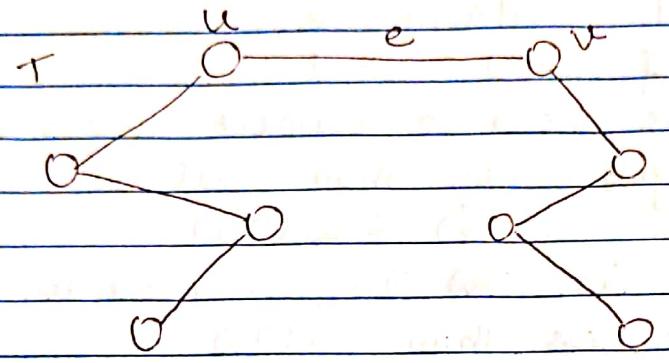
Time Complexity.

$T(V+E) + T(V^2)$

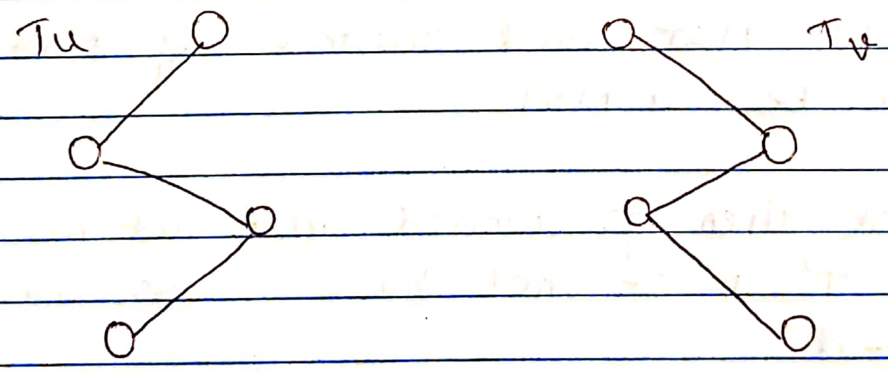
$\Rightarrow T(V^2)$

Q5) Let T be the Minimum Spanning Tree with weight W and let 'e' be the edge whose weight will decrease by d.

$\therefore$ New weight of edge $e = W - d$.



If we remove edge 'e' from T.
Let T_u and T_v be 2 new subtrees formed



Now T is not a MST, hence there exists a T' which is a MST.
If new MST doesn't have e, then its weight becomes $W(T')$

$\therefore W(T') < W - d < W(T)$

Thus $w(T')$ is new MST which has lower weight than original $w(T)$ MST.

Hence it contradicts the fact that T was an MST.

but if T' contains e,
if $w(T') < w-d$
then we need to connect nodes of Tu with weights less than $w(T_u)$
i.e. $w(T_u') < w(T_u)$
Similarily for T_v, we need to connect it with weight less than $w(T_v)$

i.e $w(T_v') < w(T_v)$

But Tu and Tv are both MST because T is MST and subtree of T should also be a MST.

if not then T would also not be an MST
∴ T' must not have less weight than $w-d$