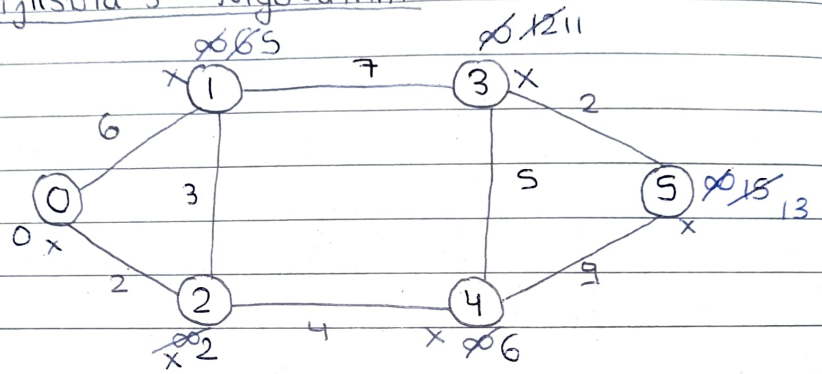
Date Page

Day - 208

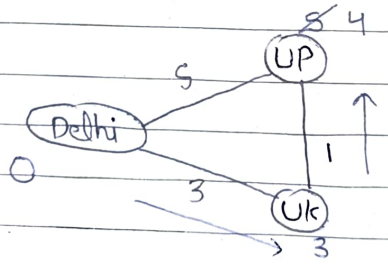
Graph - 12

* Dijkstra's Algorithm:



⇒ This algo is used to find single source shortest path.

⇒ Let's take an another example to solve the problem —



⇒ We will find shortest distance from delhi to all other places.

Delhi	D	UK	UP
	0	3	4

⇒ We know that delhi to delhi distance will be 0.

Date _____ Page _____

⇒ But we can't say that delhi - UK and delhi - UP distance will be 3 & 5 respec..

⇒ Also, the delhi - UK distance will be 3 - confirmed because it is less than Dlelhi - UP distance.

⇒ Also, it is possible that — (Delhi + UK) ⟶ UP distance will be less than Delhi → UP distance.

⇒ So, $3 + x < 5$ (possible)

Delhi $\xrightarrow{3}$ UK $\xrightarrow{1}$ UP

⇒ $3 + 1 < 5$

⇒ We don't go back and also we don't change or update any confirmed answers.

⇒ At the time of implementation, we will use two vectors — explore & distance.

⇒ Steps:

⟶ (V) { Select the unexplored vertex and its ~~digit is~~ distance is min. among all the unexp. vertexes.

⟶ (V-1) { Relax the edges:
↳ Look at your all the unexplored neighbours.

Date _____ Page _____

↳ if(dist[node] + weight < dist[neighbour])
dist[neighbour] = dist[node] + weight;

⇒ T.C. :

V + (V-1)
(V)

⇒ (V) * V

⇒ $O(V^2)$

⇒ S.C. : O(V).

⇒ This is not our most optimized solution.

⇒ As, we have to find min dis. every time. This is increasing our T.C.

⇒ So, to solve this problem, we can use priority queue.

⇒ Priority queue of type pair <int, int>

⇒ So, when we are updating the distance, we will push it into priority queue.

S.C.

⇒ O(V+E)

T.C.

⇒ Elog E + ElogE

⇒ O(Elog E) or O(ElogV)

Date _____ Page _____

⇒ So,

Method 1		Method 2
① V^2	≥	② $E \log V$

For dense graph $\cong V^2$ (Edges),

$$V^2 > V^2 \log V$$

For sparse graph, Edges $\cong V$,

$$V < V \log V$$

⇒ we mostly time uses sparse graph. in real life.

⇒ This algo is not work for -ve weights.