

GRAPHS...

video-39

"let's make it easy too"



If you have tried my
"Graph Concepts & Qns" playlist,
these Qns, will seem very easy.
Do try it once ;)



Facebook
Instagram } → codestorywithMIK

(Twitter) → CSwithMIK

codestorywithMIK →

Design Graph with "Shortest Path Calculator", ...

Company Tag :- will update soon...

2642. Design Graph With Shortest Path Calculator

Hint

Hard 243 25

Companies

There is a **directed weighted** graph that consists of n nodes numbered from 0 to $n - 1$. The edges of the graph are initially represented by the given array `edges` where `edges[i] = [fromi, toi, edgeCosti]` meaning that there is an edge from `fromi` to `toi` with the cost `edgeCosti`.

Implement the `Graph` class:

- `Graph(int n, int[][] edges)` initializes the object with n nodes and the given edges.
- `addEdge(int[] edge)` adds an edge to the list of edges where `edge = [from, to, edgeCost]`. It is guaranteed that there is no edge between the two nodes before adding this one.
- `int shortestPath(int node1, int node2)` returns the **minimum** cost of a path from `node1` to `node2`. If no path exists, return `-1`. The cost of a path is the sum of the costs of the edges in the path.

(Dijkstra)

Pre-requisite :-

- (i) Dijkstra's Algorithm. ✓✓
- (ii) Floyd Warshall Algorithm. ✓✓



GRAPH CONCEPTS
& QNS

Graph Concepts & Qns :
Convert Story To Code

7

codestorywithMIK
41 videos Public

Now, YOU WILL ALSO BE ABLE TO SOLVE GRAPH QNS IT'S MY GAURANTEE
TOPIC: INDRODUCTION & SHAPATH GRAHAN

The main motive of this playlist is to understand each Topic of Graph and solve qns on eac...MORE

Play all Shuffle

VIDEO-24 DIJKSTRA'S PART-1 30:51
Dijkstra's Algorithm | PART-1 | (Microsoft) | Graph Concepts & Qns - 24 | Explanation+Coding
codestorywithMIK ✓

VIDEO-25 DIJKSTRA'S PART-2 17:32
Dijkstra's Algorithm | PART-2 | (Microsoft) | Graph Concepts & Qns - 25 | Explanation+Coding
codestorywithMIK ✓

VIDEO-26 DIJKSTRA'S PART-3 19:46
Dijkstra's Algorithm | PART-3 | Why not Queue ? | Microsoft | Graph Concepts & Qns - 26 | Explanation
codestorywithMIK ✓

VIDEO-32 FLOYD WARRSHALL ALGORITHM 22:40
Floyd Warshall Algorithm | Full Detail | Samsung | Graph Concepts & Qns - 32 | Explanation + Coding
codestorywithMIK ✓

Dijkstra's Algo

$\text{unordered_map} <\text{int}, \text{vector} <\text{pair} <\text{int}, \text{int}>>> \text{adj};$

$\text{Graph}(\text{int } n, \text{vector} <\text{vector} <\text{int}>> \& \text{edges}) \{$

$O(V+E)$

u
v
cost

$\text{adj}[u].\text{push_back}(\{v, \text{cost}\});$

}

M

void addEdge (vector<int> edge) {
 edge = {u, v, cost}

O(1)
 }

adj[u].push_back({v, cost})

int ShortestPath (int node1, int node2) {

Disj's Code ←

}

$O(M \times \log V)$

Floyd Warshall

$O(V^3)$

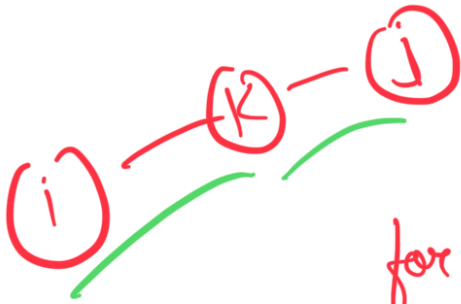
	v1	v2	v3	v4
v1	✓	✓	✓	✓
v2	✓	✓	✓	✓
v3	✓	✓	✓	✓
v4	✓	✓	✓	✓

Graph (int n, vector<vector<int>> & edges) {

$[n] [n] \rightarrow \{10^9\}$

edges $\rightarrow u, v, cost$

$\hookrightarrow [u][v] = cost \leftarrow$



for (k = 0; k < n; k++) {

for (int i = 0; i < n; i++) {

for (int j = 0; j < n; j++) {

adj[i][j] = max(adj[i][j],
adj[i][k] + adj[k][j])

~~$O(N^3)$~~

Floyd
warshall

}

void addEdge (vector<int> edge) {

$u, v, cost$
 \uparrow

for (i = 0; i < n)

for (j = 0; j < n) {

$adj[i][j] = \min(adj[i][j],$

$adj[i][u] + cost + adj[v][j]$

int ShortestPath (int node1, int node2) {

~~$O(N^2)$~~

return Adj[node] [node] == 10⁹ ? -1

: adj[u][v].

}