Placement DSA sheet by Arsh Goyal

**Day-3(‎06-‎05-‎2022)**

**Q-1=> BFS of graph (**[**https://practice.geeksforgeeks.org/problems/bfs-traversal-of-graph/1**](https://practice.geeksforgeeks.org/problems/bfs-traversal-of-graph/1)**)**

hint\_1=> we have to move step by step means layer by layer and keep visited vector updated and use queue to store the vertexes.

**Tips - 1 =>** when we travese graph through bfs then we reach at any vertex in minimum time from root vertex. So we can use this aproach to find the shortest path b/w two vertex. We also find the shortest path using dfs but it will take more time than this aproach.

**Q-2=> DFS of graph (**[**https://practice.geeksforgeeks.org/problems/depth-first-traversal-for-a-graph/1/#**](https://practice.geeksforgeeks.org/problems/depth-first-traversal-for-a-graph/1/#)**)**

hint\_1=> in this question we have to move recursivily first we complete each of the element conected with other and then we move to other.

**Q-3=> Detect cycle in an undirected graph (**[**https://practice.geeksforgeeks.org/problems/detect-cycle-in-an-undirected-graph/1#**](https://practice.geeksforgeeks.org/problems/detect-cycle-in-an-undirected-graph/1#)**)**

hint\_1=> hame ye dhyan rakhna hai ki ye undirected graph hai to ham apne just last wale vertex pe nahi ja sakte hai iske liye hame ek pre data ko fun. ke sath hi bhejna hoga aur ham dfs ke samay pre ke alawa baki sab me jayenge. baki sab kuchh dfs hi hai.

**Q-4=> Detect cycle in an directed graph (**[**https://www.geeksforgeeks.org/detect-cycle-in-a-graph/**](https://www.geeksforgeeks.org/detect-cycle-in-a-graph/)

hint\_1=> iss wale circle detection me hame ye dhyan rakhna hai ki ham jab dfs karte hue deep ja rahe ho to agar kahi pe visited 1 milta hai to sirf tabhi 1 return karna hai jab uss node ko ham abhi tak jitne depth pe ho waha tak usko traverse kar ke aaye ho isiliye ham ek aur check vector lekar chalenge aur jab deep jayenge to usko call kar denge aur jab bahar aayenge to usko delete kar denge and time complexity ke liye hame agara kisi round ko check kar chuke hai to uss round ke pahle hi return 0 kar dena hai;

**Q-5=> Topological sort (**[**https://practice.geeksforgeeks.org/problems/topological-sort/1**](https://practice.geeksforgeeks.org/problems/topological-sort/1)**)**

hint\_1=> in this question we have to use stack to track the item in deep jab bhi ham deep jaye aur jab bhi ham sabse deepest place se nikalne lage to uss ele ko stack me push kar de phir last me stack me se top se nikal kar usko vector me dal lenge.

**kahn's algorithem =>** jab ham topological short ko bfs ka use karke karte hai to usi ko kahn's algorithem kahte hai. isme hame ek aur data structure ka use hoga jisme ham ye store karenge ki kis node pe kitne path aa rhe hai aur jis node pe koi bhi path nahi aa raha hoga wo obius si bat hai ki pahle hi rahega isiliye use ham queue me store kar lenge aur jab bhi ham isko pop kar ke iske agal bagal walo ko traverse karenge to unke incoming path ko - kar denge aur agar wo 0 hai to queue me le lenge;

**kahn's algorithem =>** isi ka use kar ke ham directed graph me cycle bhi detect kar sakte hai using bfs (mens agar koi puchhe ki bfs se cycle kaise find karenge directed graph me to ham simple kahn's algorithem ka use karke topological sort vector nikalenge agar hame valid vector mila matlab cycle nahi hai kyuki topological sort directed acyclic graph me hi hota hai cyclic me ye nahi hota ) aur validataion check karne ke liye simple vector ke length ko number of node se compare kar lenge agar = hua matlab valid hai aur nahi to invalid;

**Q-6=> Shortest path in an undirected unweighted graph (**[**https://www.codingninjas.com/codestudio/problems/shortest-path-in-an-unweighted-graph\_981297?leftPanelTab=0**](https://www.codingninjas.com/codestudio/problems/shortest-path-in-an-unweighted-graph_981297?leftPanelTab=0&utm_source=youtube&utm_medium=affiliate&utm_campaign=Lovebabbar) **&**[**utm\_source=youtube HYPERLINK "https://www.codingninjas.com/codestudio/problems/shortest-path-in-an-unweighted-graph\_981297?leftPanelTab=0&utm\_source=youtube&utm\_medium=affiliate&utm\_campaign=Lovebabbar"& HYPERLINK "https://www.codingninjas.com/codestudio/problems/shortest-path-in-an-unweighted-graph\_981297?leftPanelTab=0&utm\_source=youtube&utm\_medium=affiliate&utm\_campaign=Lovebabbar"utm\_medium=affiliate HYPERLINK "https://www.codingninjas.com/codestudio/problems/shortest-path-in-an-unweighted-graph\_981297?leftPanelTab=0&utm\_source=youtube&utm\_medium=affiliate&utm\_campaign=Lovebabbar"& HYPERLINK "https://www.codingninjas.com/codestudio/problems/shortest-path-in-an-unweighted-graph\_981297?leftPanelTab=0&utm\_source=youtube&utm\_medium=affiliate&utm\_campaign=Lovebabbar"utm\_campaign=Lovebabbar" HYPERLINK "https://www.codingninjas.com/codestudio/problems/shortest-path-in-an-unweighted-graph\_981297?leftPanelTab=0&utm\_source=youtube&utm\_medium=affiliate&utm\_campaign=Lovebabbar"utm\_campaign=Lovebabbar**](https://www.codingninjas.com/codestudio/problems/shortest-path-in-an-unweighted-graph_981297?leftPanelTab=0&utm_source=youtube&utm_medium=affiliate&utm_campaign=Lovebabbar)**)**

hint\_1=> hame isme bfs ka use karna hai bas yahi change karna hai ki jab bhi ham kisi vertex pe jaye to use vertex ke parent ko store kar lena hai isse ham reverse direction me traverse kar ke pta laga sakte hai shortest path kya hai (**tips 1**);

hint\_2=> ham isme dfs ka bhi use kar sakte hai uske liye hame do array aur ek lenth ko extra bnana hai ek array me temprary path hoga jab bhi ham deep jayenge to usme vertex ko dal denge aur jab bahar ayenge to nikal lenge. Aur jab ham apne target wale vertex pe honge to uspe ham length ko compare karenge aur agar length > hoga to main array me temprary wale ko store kar ke length ko change kar lenge.(see the tle submision of above question);

**Q-7=> Shortest path in an undirected weighted graph (**[**https://www.codingninjas.com/codestudio/problems/dijkstra-s-shortest-path\_920469?leftPanelTab=0**](https://www.codingninjas.com/codestudio/problems/dijkstra-s-shortest-path_920469?leftPanelTab=0&utm_source=youtube&utm_medium=affiliate&utm_campaign=Lovebabbar) **&**[**utm\_source=youtube HYPERLINK "https://www.codingninjas.com/codestudio/problems/dijkstra-s-shortest-path\_920469?leftPanelTab=0&utm\_source=youtube&utm\_medium=affiliate&utm\_campaign=Lovebabbar"& HYPERLINK "https://www.codingninjas.com/codestudio/problems/dijkstra-s-shortest-path\_920469?leftPanelTab=0&utm\_source=youtube&utm\_medium=affiliate&utm\_campaign=Lovebabbar"utm\_medium=affiliate HYPERLINK "https://www.codingninjas.com/codestudio/problems/dijkstra-s-shortest-path\_920469?leftPanelTab=0&utm\_source=youtube&utm\_medium=affiliate&utm\_campaign=Lovebabbar"& HYPERLINK "https://www.codingninjas.com/codestudio/problems/dijkstra-s-shortest-path\_920469?leftPanelTab=0&utm\_source=youtube&utm\_medium=affiliate&utm\_campaign=Lovebabbar"utm\_campaign=Lovebabbar" HYPERLINK "https://www.codingninjas.com/codestudio/problems/dijkstra-s-shortest-path\_920469?leftPanelTab=0&utm\_source=youtube&utm\_medium=affiliate&utm\_campaign=Lovebabbar"&**](https://www.codingninjas.com/codestudio/problems/dijkstra-s-shortest-path_920469?leftPanelTab=0&utm_source=youtube&utm_medium=affiliate&utm_campaign=Lovebabbar)**)**

**Dijkstra's algorithem =>** iss algorithem ka use ham undirected wieghted graph me shortest path nikalne ke lihye karte hai isme sare weights +ve hone chahiye tabhi ye kam karega.

isme ham ek set ya phir priority queue lete hai aur source node ke distance =0 aur node ko insert kar dete hai iske baad ham apne ans vector ko define karte hai infinite ke sath aur iske bad ham set se minimum distance wala node nikalte jayenge aur phir us node ke sare neighour pe jake unpe check karenge ki wo current dist. se < hai ki nahi agar < hai to set mese agar unka pahle se koi hoga to usko nikal kar naye wale ko enter kar denge;

**Q-8=> Shortest path in a directed acyclic graph(weighted ) (**[**https://ideone.com/nA6Np4**](https://ideone.com/nA6Np4)**)**

hint\_1=> iss question ke liye apne pass bhut se alborithm hai jaise bellmon ford o(ve), dijkstra's o(e+vlogv) but ye sab efficient aproach nahi hai agar ek vertex se sabhi vertex ka shortest distance puchha hai to iske liye ham pahle **topological short** wale vector ko nikalna hoga iske baad ham vector ke sabse bahar wale vertex se chalu karenge aur andar tak jayenge aur jab bhi koi int\_max ke == nahi hoga uske liye phir adj me ghus kar uske sare adjacent pe ja ke check karna hai aur jo bhi uska weight hai usme current wale se compare karke usko add kar dena hai agar wo pahle wale se kam hai to;

**Q-9=> Shortest path in a graph(weighted -ve and -ve cycle ) (**[**https://www.codingninjas.com/codestudio/problems/bellmon-ford\_2041977?leftPanelTab=0&utm\_source=youtube&utm\_medium=affiliate&utm\_campaign=Lovebabbar**](https://www.codingninjas.com/codestudio/problems/bellmon-ford_2041977?leftPanelTab=0&utm_source=youtube&utm_medium=affiliate&utm_campaign=Lovebabbar)**)**

**Bellmon ford algorithem =>** iss algorithem me ham ek distance ka array banate hai aur usko INT\_MAX ke sath initialige aur dt[src] ko 0 karne ke bad n-1 time har edge pe jake ye wala formula run karte hai dist[u]!= int\_max && dist[u] +wt < dt[v] aur agar sahi hai to dt[v] ko update kar dete hai; phir nth wala alag se run karte hai same formula ke sath aur agar iss wale me bhi dt update hua to iska matlab ki graph me -ve cycle hai nahi to dt wala array hi apna shortest distance hai;

**Q-10=> Minimum Spaning Tree (**[**https://practice.geeksforgeeks.org/problems/minimum-spanning-tree/1/**](https://practice.geeksforgeeks.org/problems/minimum-spanning-tree/1/)**)**

**Minimum spaning Tree =>** agar ham kisi graph ko tree me convert kar sake jisme n node aur n-1 edges ho to use st kahte hai aur agar ye ham wheighted graph me karte hai to phir jo tree bna hai uske wt ka sum min hai to use mst kahte hai;

**Prims Algorithem =>** isme ham 3 vector lenge 1st key ka jisme distance store hoga 2nd mst ka jisme ye store hoga ki kaun sa node mst me chala gaya hai aur 3rd wala parent ko track karne ke liye; ham pahle kise bhi node ke distance yani key ko 0 kar denge iske baad har bar ham key me uss minimum distance wale node ko find karenge jo mst me nahi hai aur use mst me dalne ke bad ham iss node ke sare nbr jo mst me na ho uspe check karenge ki unka key unke wt se kam hai ki nahi agar nahi hai to key ko update kar denge;

**Kruskal's Algorithem =>** isme ham disjoint set aur union by rank aur path optimisation ka use karenge find\_prt aur union wala disjoint set ka part hai iska use ham cycle find karne me bhi karte hai isme hame sare edges pe loop chalana hota hai (mst ke liye edges ko wt ke basis pe sort karna hota hai) eske bad har bar u aur v ka prt find karenge agar dono ka prt same hai means cycle hai (mst ke liye iss case me kuchh nahi karna hai) but agar == nahi hai to dono ka union kar denge ; agar rank == hai to kisi ko bhi kisi me add kar denge bas prt wale ka rank increase karenge aur == nahi hai to chhote wale ko bade wale ka child bna denge;

**Q-11=> Bridges in graph (**[**https://www.codingninjas.com/codestudio/problems/bridges-in-graph\_893026?leftPanelTab=0&utm\_source=youtube&utm\_medium=affiliate&utm\_campaign=Lovebabbar**](https://www.codingninjas.com/codestudio/problems/bridges-in-graph_893026?leftPanelTab=0&utm_source=youtube&utm_medium=affiliate&utm_campaign=Lovebabbar)**)**

**bridge in graph =>** agar kisi graph me kisi edge ko hatane pe graph do component me divide ho jata hai to use edge ko bridge kahte hai;

hint\_1=> isme ham 3 vector banayenge 1st wala discovery ke liye 2nd wala low value ke liye and 3 rd wala vsited aur ek timer bhi chalega parent ke sath; aab ham har bar dis aur low ko initiali time ke = kar ke time ko ++ kar denge iske baad ham us node ke nbr me jayeng aur agar back edge hai to low me node aur nbr ke dis ka min hoga aur jab ham wapas aayenge call se to phir low me node aur nbr ke low ka min hoga phir edge wale condition ka matlab hai ki agar nbr ka low apne node ke discovery se bada hai means nbr pe ham node se hokar hi gaye honge ya phir iske bad tabhi uska value isse jayada hai nahi to low ka value update ho gaya hota;

**Q-12=> Articulation point in graph (**[**https://www.codingninjas.com/codestudio/library/articulation-points-in-a-graph**](https://www.codingninjas.com/codestudio/library/articulation-points-in-a-graph)**)**

**AP. in graph =>** agar kisi graph me kisi node ko hatane pe graph do component me divide ho jata hai to use node ko AP. kahte hai;

hint\_1=> iska algorithem bhi same hoga jaise bridge ka hai bas yahi change hoga ki agar bridge wale condition me bas = bhi add ho jayega aur prt !=-1(ye iss liye add hoga kyuki pahle wale condition se ye to pta chal jayega ki ham apne nbr pe iss node se hi hokar jayenge bur iss node ke uper bhi to node hona chahiye tabhi to graph do component me divide hoga) bas; last me isme ye bhi chek karna hoga ki agara kisi node ka prt -1 hai aur uska child > 1 hai to wo node bhi AP hoga;

**Q-13=> Minimum Spaning Tree (**[**https://www.codingninjas.com/codestudio/problems/count-strongly-connected-components-kosaraju-s-algorithm\_1171151?leftPanelTab=0&utm\_source=youtube&utm\_medium=affiliate&utm\_campaign=Lovebabbar**](https://www.codingninjas.com/codestudio/problems/count-strongly-connected-components-kosaraju-s-algorithm_1171151?leftPanelTab=0&utm_source=youtube&utm_medium=affiliate&utm_campaign=Lovebabbar)**)**

**Strongly conected component =>** kisi graph me koi aisa segment hai jisme hame kisi bhi node se uss segment ke sare node pe ja sakte hai us segment ko ham SCC kahte hai;

**Kosaraju's Algorithem =>** isme ham pahle tp sort vectore nilkalenge phir graph ka transpose nikal kar ham tp sort vector ke basis pe graph me dfs karenge aur jitne bar bhi hame disconected graph milega wahi SCC count hoga;

**Q-14=> Number of Island (**[**https://leetcode.com/problems/number-of-islands/**](https://leetcode.com/problems/number-of-islands/)**)**

**Q-15=> Flood Fill (**[**https://leetcode.com/problems/flood-fill/**](https://leetcode.com/problems/flood-fill/)**)**

**Q-16=> Rat in Maze (**[**https://practice.geeksforgeeks.org/problems/rat-in-a-maze-problem/1**](https://practice.geeksforgeeks.org/problems/rat-in-a-maze-problem/1)**)**

**Q-17=> Steps By knight (**[**https://practice.geeksforgeeks.org/problems/steps-by-knight5927/1**](https://practice.geeksforgeeks.org/problems/steps-by-knight5927/1)**)**

**Q-18=> Number of Operations to Make Network Connected (**[**https://leetcode.com/problems/number-of-operations-to-make-network-connected/**](https://leetcode.com/problems/number-of-operations-to-make-network-connected/)**)**

**Q-19=>Find Eventual Safe States (**[**https://leetcode.com/problems/find-eventual-safe-states/**](https://leetcode.com/problems/find-eventual-safe-states/)**)**

hint\_1=> isme dp ke momorisation aur dfs ka use karna hai nahi to tle aa jayega;

**Q-20=>Time Needed to Inform All Employees (**[**https://leetcode.com/problems/time-needed-to-inform-all-employees/**](https://leetcode.com/problems/time-needed-to-inform-all-employees/)**)**

hint\_1=> isme hame jo manager wala array hai wo ek tarah se prt ka array hai to ham simple prt pe hi har ek node ke liye time nikal lenge aur usme se jo maximum time hoga wahi ans hoga bas isme ham memorisation lga ke time complexity ko ghata sakte hai;

**Q-21=>Negative weight cycle (**[**https://practice.geeksforgeeks.org/problems/negative-weight-cycle3504/1#**](https://practice.geeksforgeeks.org/problems/negative-weight-cycle3504/1#)**)**

hint\_1=> isme ham belmond ford wale algorithem ka use karke -ve cycle find karenge (iske liye wahi last wale itration me agar dt[v] ki value change hoti hai to phir -ve cycle hai);