30.03.2020

5. Single Source Shortest Path – Dijkstra's Algorithm

(a) Pseudocode

```
procedure dijkstra (G, l, s)
Input:  Graph G = (V, E), directed or undirected;
        positive edge lengths {l_e : e ∈ E} ; vertex s ∈ V.
Output: For all vertices u reachable from s, dist(u) is set to the
        distance from s to u.

for all u ∈ V:
    dist (u) = ∞
    prev (u) = nil
dist(s) = 0

H = makequeue(V)   (using dist-values as keys)
while H is not empty:
    u = deletemin(H)
    for all edges (u,v) ∈ E:
        if dist (v) > dist (u) + l(u,v):
            dist (v) = dist (u) + l(u,v)      ⟶ Relaxation
            prev (v) = u
            decreasekey (H, v)
```

(b) Compute the shortest dist from vertex 1 to all other vertices using Dijkstra's algorithm.

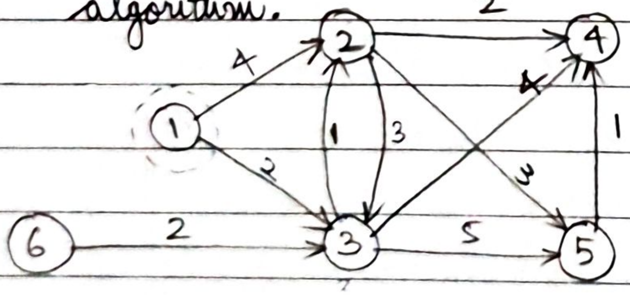
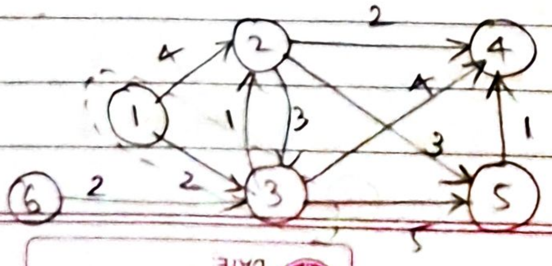
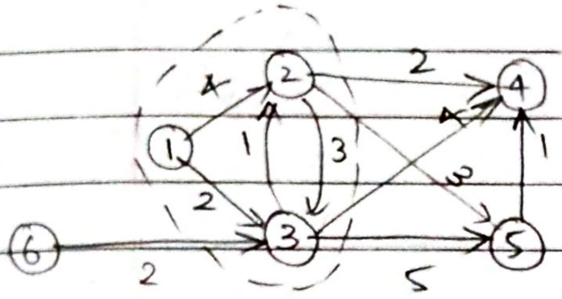


table with node : dist values

1 : 0	4 : ∞
2 : 4	5 : ∞
3 : 2	6 : ∞

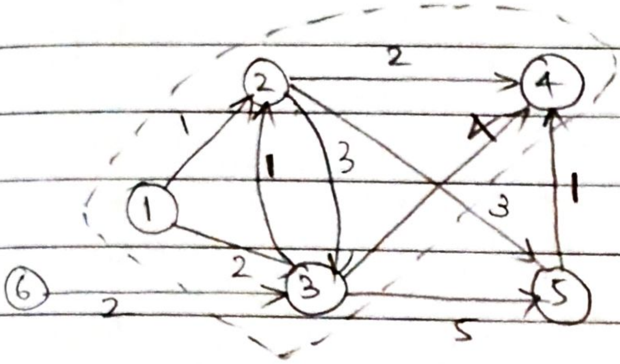


1 : 0	4 : ~~∞~~ 6 ($\infty > 2+4$)
2 : 3 ($4 > 2+1$)	5 : ~~∞~~ 7 ($\infty > 2+5$)
3 : 2	6 : ∞

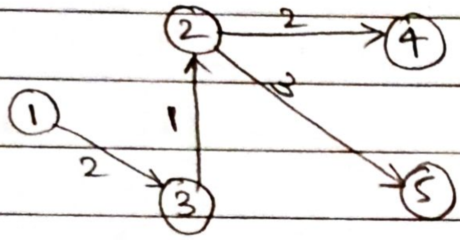


$6 > 3+2$

1: 0	4: 5
2: 3	5: 6 ($7 > 3+3$)
3: 2	6: ∞



1: 0	4: 5
2: 3	5: 6
3: 2	6: ∞



shortest distances from 1 to the other vertices:

1: 0	4:5
2 : 3	5:6
3 : 2	6 : ∞ (no path)