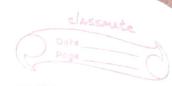
```
Exp 5
  SSSP Problem
1] Dijkstra's Algo.
* Graph
   Vertices: {0,1,2,3}
   Edges: 0 -> 1 (weight 4)
          0 \rightarrow 2 (1)
         2 -> 1 (2)
           1 \rightarrow 3 (1)
          2 \rightarrow 3 (5)
* Execution
  1) Initializath
     dist = [0, \infty, \infty, \infty] .... src 0
      pg=(0,0)] -- distance O to vertex O
  2) First iterath
    process vertex 0 (smallest dist. in pg)
     relax edges from O
     Edge 0 \rightarrow 1: update dist[1] = 0 + 4 = 4
          0 \to 2: | dist [2] = 0+1 = 1
    pq = \{ (1,2), (4,1) \}
  3.) Second it.
     process vertex 2 (relax edges from 2). smallest dist in pg
         2 \rightarrow 1: update dist[1] = 1 + 2 = 3 (shorter than 4)
         2 -> 3: update dist[3] = 1 + 5 = 6
     pq = \{ (3, 1), (6, 3) \}
  4) Third it.
      process vertex 1 (smallest dist in pg)
      relax edges from 1
          1 \rightarrow 3: update dist[3] = 3+1=4 (shorter than 6)
      pg = { (4,3) }
```



```
5) Fourth it.
    process vertex 3 (smallest dist. in pg)
    no edges to relax.
    pq = { }
```

27 Bellman-Ford Algo.

\* Graph

$$0 \rightarrow 2 (1)$$

$$2\rightarrow 1(-2)$$

$$1 \rightarrow 3(1)$$

$$2\rightarrow 3(5)$$

\* Executà

$$dist = [0, \infty, \infty, \infty]$$
 ---- src. 0

$$2 \rightarrow 1$$
: " dist[1] = 1+(-2) = -1  
1  $\rightarrow$  3: " dist[3] = -1 + 1 = 0

$$1 \rightarrow 3$$
: " dist  $[3] = -1 + 1 = 0$ 

3']	All Pairs Shortest Path
*	Graph
	Vertices: {0,1,2}
	Edges: $0 \rightarrow 1 (wt. 2)$
	0->2 (4)
	$1\rightarrow 2(1)$
	$2 \rightarrow 0 (1)$
*	Execut? - For each vertex as src, run Dijkstra's algo. & return
	2D arr. with shortest distances.
	1.) Src. 0
	$0 \rightarrow 0$ : dist $[0] = 0$ src $0$
	$0 \rightarrow 1$ : dist $[1] = 2$
	$0 \rightarrow 2$ : path $0 \rightarrow 1 \rightarrow 2$ , dist[2] = 2+1 = 3
	2) Src. 1  1 to 0: no path; dist[0] = INF
	1 to 0: no path; dist[0] = INF
	1 to 1; dist[1] = 0 src 1
	1 to 2: dist [2] = 1
	3.) Src. 2
	2 to 0 : dist[0] = 1
	2 to 1 : Path 2→0→1, dist[1] = 1+2=3
	2 to 2 : dist[2] = 0 src 2
	[all Distances' arr = $\begin{bmatrix} 0 & 2 & 3 \\ \infty & 0 & 1 \\ 1 & 3 & 0 \end{bmatrix}$
	∞ 0 1
<b>9</b> 0	This example depicts limitath of Dijkstra's algo it's a greed algo. that assumes that once a vertex is processed, its
2 2 3	algo. that assumes that once a vertex is processed, its
	shortest dist. from the src. is finalized.

\* Execut! - For each vertex run Bellman-Ford algo.

1:) Src. 0

0 to 0: dist[0] = 0

O to 1: dist[1] = 2O to 2:  $0 \rightarrow 1 \rightarrow 2$ , dist[2] = 2+1 = 3

2.) Src. 1

1 to 0:  $1 \rightarrow 2 \rightarrow 0$ , dist[0] = 1 + 1 = 2

1 to 2: dist[2] = 1

1 to 1: dist[1] = 0

3.) Src. 2

2 to 0: dist[0] = 1

2 to 1: 2 - 0 -> 1, dist[1] = 1+2 = 3

2 to 2: dist[2] = 0

[all Distances' arr = 
$$\begin{bmatrix} 0 & 2 & 3 \\ 2 & 0 & 1 \\ 1 & 3 & 0 \end{bmatrix}$$

· Correctly computes shortest paths in graphs with cycles

• Detects -ve <del>edge</del> weight cycles if they exist.