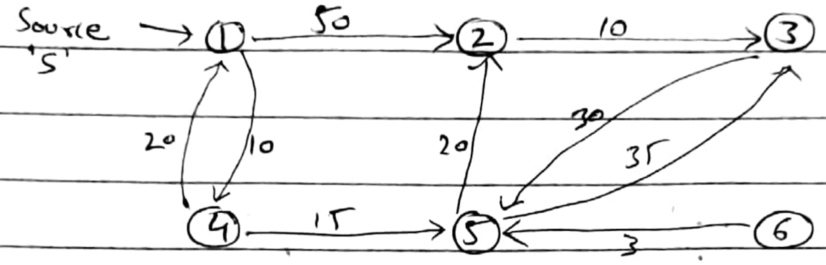
Single Source Shortest Path

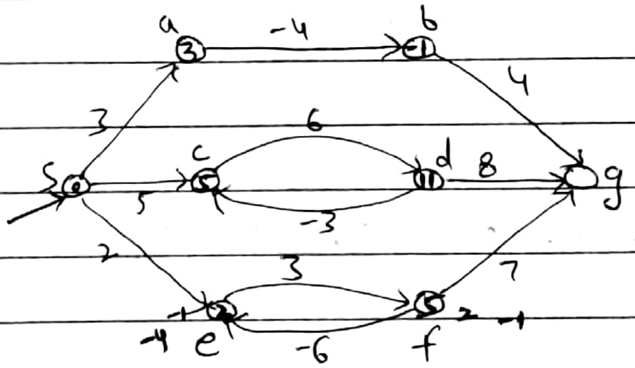
Given a graph $G = (V, E)$ we want to find a shortest path from a given source vertex $s \in V$ to each vertex $v \in V$



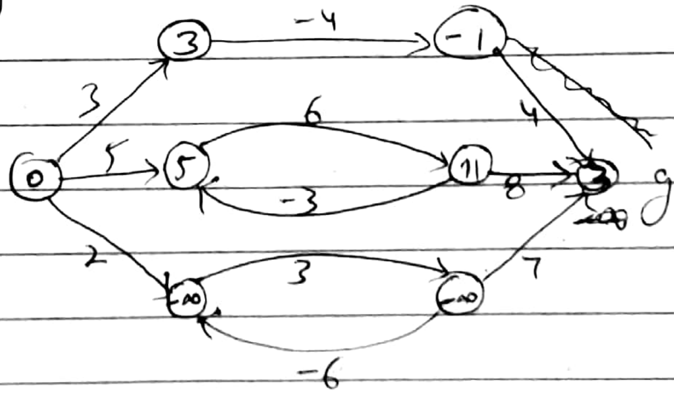
Path	Length
(1,4) : 1,4	10
(1,2) : 1,4,5,2	45
(1,3) : 1,4,5,2,3	55
(1,5) 1,4,5	25
(1,6)	∞

Terminology

1) vertex
2) Edge
3) weighted edge
   1) Positive
   2) Negative
4) Cycle
   1) Positive weighted
   2) Negative weighted



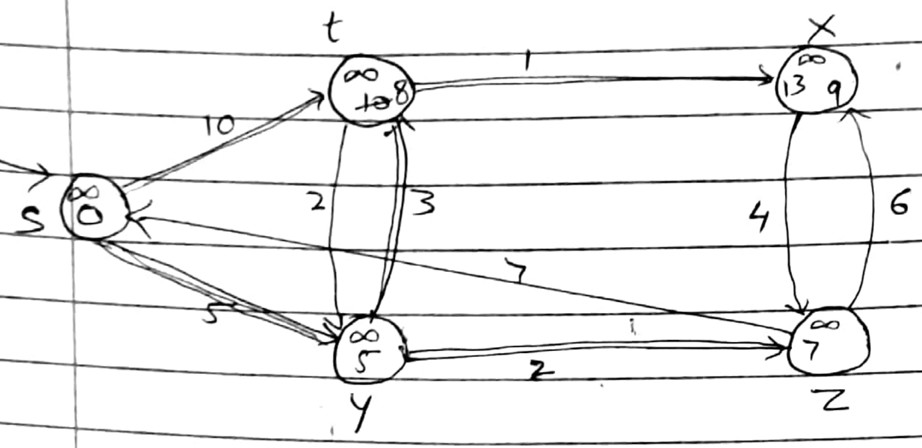
$\Downarrow$



__/__/__

1) Dijkstras Algorithm (Negative weight but not negative cycle)
2) Bellman ford Algorithm (Positive weight)

DIJKSTRAS Algorithm
for the problem in which all egdes weight are non negative.



DIJKSTRA (G, w, S)
1) INITIALISE - SINGLE - SOURCE (G, S)
2) $S \leftarrow \phi$
3) $\phi \leftarrow V[G]$
4) while $Q = \phi$
do $u \leftarrow$ Extract min (Q)
$S \leftarrow S \cup \{u\}$
for each vertex $v \in Adj(u)$
do RELAX (u, v, w)

INITIALISE - SINGLE - SOURCE (G, S)
for each vertex $v \in V[G]$
do $d[v] \leftarrow \infty$
$\pi[v] \leftarrow NIL$

Scanned with CamScanner

RELAX (u, v, w)

if $d[v] > d[u] + w(u,v)$

then $d[v] \leftarrow d[u] + w(u,v)$

$\pi[v] \leftarrow u$

s · [

S = [S, y, z, t, x]

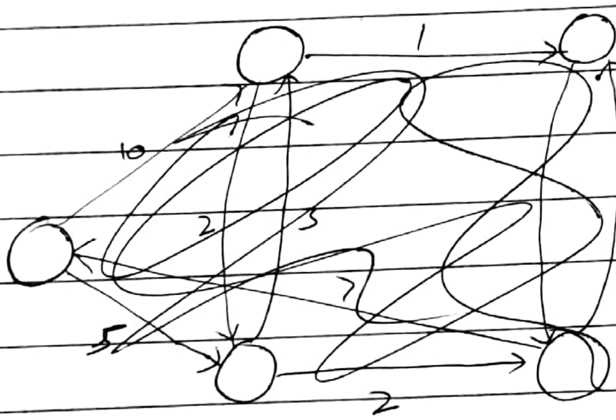
Q = [s, t, x, y, z]

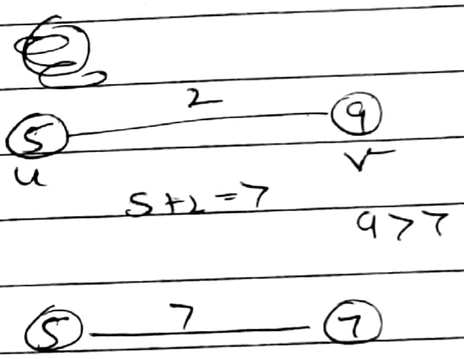
Q = [t, x, y, z]

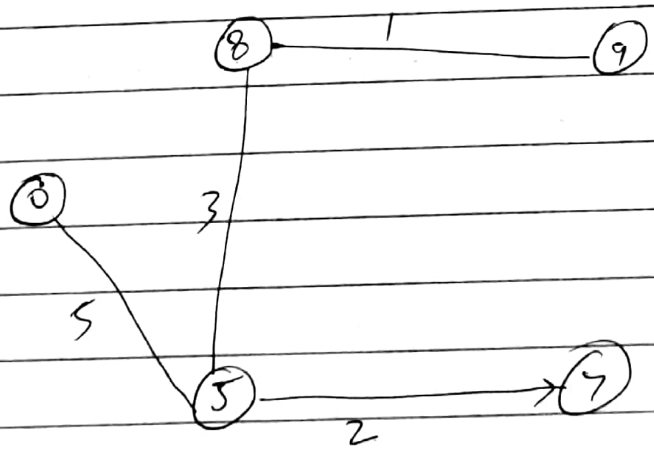
Q = [t, x, z]

Q = [t, x]

Q = [x]

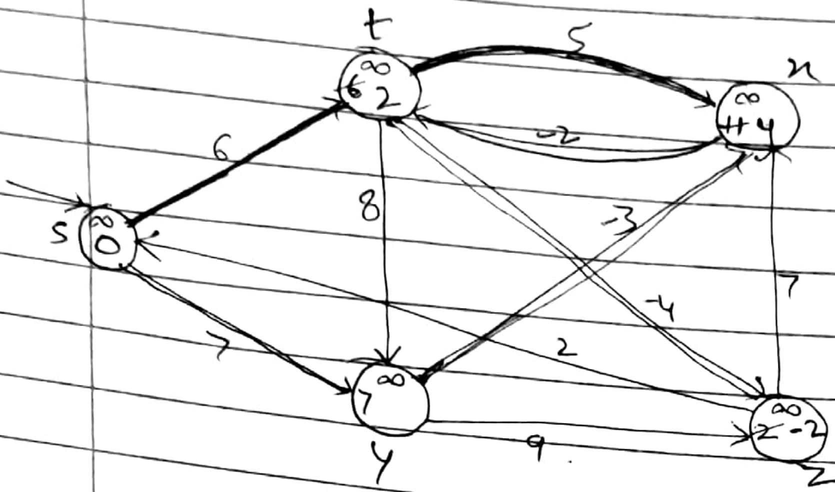






Bellman-ford-Algorithm

Solve the problem with weight may be negative but not negative cycle.




```
BELLMAN-FORD (G,w,s)
INITIALIZE-SINGLE-SOURCE (G,s)
for i ← 1 to |V[G]| - 1
        do for each edge (u,v) ∈ E[G]
              do RELAX (u,v,w)
for each edge (u,v) ∈ E[G]
        do if d[v] > d[u] + w[u,v]        -ve weight cycle
              then Return FALSE
RETURN TRUE
```

```
INITIALIZE-SINGLE-SOURCE (G,s)
for each vertex v ∈ V[G]
        do d[v] ← ∞
           π[v] ← NIL
d[s] ← 0
```

Scanned with CamScanner

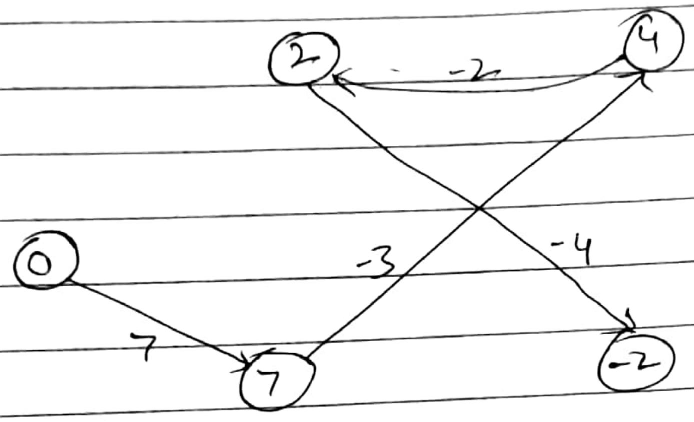
_/_/_

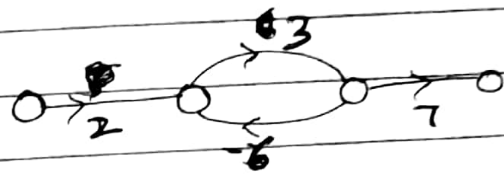
RELAX (u, v, w)

1) if d[v] > d[u] + w(u, v)

2) then d[v] ← d[u] + w(u, v)

3) π[v] ← u





Dynamic Programming

→ It is a technique for solving recursively or non recursively by memorization or tabulation method.

→ Recomputation is avoided, Reusability is involved.

→ Greedy approach does not give optimal solution guarantee but Dynamic Programing gives guarantee of optimal solution by using principle of optimatimality.

Principle of optimality: A problem is said to satisfy principle of optimality if the sub solution of an optimal solution of the problem are themselves optimal solution for their subproblems.