Bellman Ford

$n = |V|$ $m = |E|$ source $s \in V$.
$G = (V, E)$ $w : E \to R$. directed.

$\delta(u, v)$: dist from u to v.
$P(u, v)$: shortest path from u to v.

To find $\delta(s, v)$ and $P(s, v)$ $\forall$ $v \in V$.

if there are NO negative cycles, all shortest paths in G possess optimal substr property.
i.e. if $P(s, y) = P(s, x) :: (x, y)$ is to y,
then (s, x) is the shortest path to x.

Dijsktra → based on weights of all in neigh
Bellman Ford → " " no of edges.
Floyd Warshall → based on max- index in path.

Observⁿ: $P(s, v) \to i$ edges $\Rightarrow P(s, x) \to (i-1)$ edges
and (x, v)

$P(v, i)$: shortest $(s \to v)$ at most i edges.
$L(v, i)$: length of $P(v, i)$.

$$L(v, i) = \min\left(L(v, i-1), \min_{(x,v) \in E} \left(L(x, i-1) + w(x, v) \right) \right)$$

Base Case:
if $(s, v) \in E$ then $L[v, 1] \leftarrow w(s, v)$ else ∞.
and $L[s, 1] \leftarrow 0$.

$O(mn)$ time, $O(n)$ space.

Key observation:

i) If no negative wt cycle, then $L[v,n] = L[v,n-1] \; \forall \; v \in V$.

ii) If there is a path from s to v of K edges, if after j^{th} iter^n $L[v,j]$ has a finite value, then $j \leq K$.

Detecting negative cycle:
execute one more iter^n of BF.

- If $L[v,n] \neq L[v,n-1]$ for any $v \in V$ then negative cycle.
- Else $L[v,n-1]$ is the distance to $v \; \forall \; v \in V$.

Floyd-Warshall Algo

(All Pairs Shortest Paths Algo)

Dijsktra : $O(m + n \log n)$
Bellman Ford : $O(mn)$

Data Str for reporting shortest path from s:
Shortest paths tree rooted at s.
Time taken $= O(|P(u,v)|)$
(all pairs $\Rightarrow \forall s \Rightarrow$ $O(n^2)$ data str)

$P_K(i,j)$: shortest path from i to j with intermediate vertices of index $\leq K$.

$D_K(i,j)$: length of $P_K(i,j)$.

0 intermediate vertices

$\delta(i,j) = P_n(i,j)$

$$D_0(i,j) = \begin{cases} W(i,j) \\ \infty \end{cases}$$

$$D_0(i,i) = 0$$

$$D_K(i,j) = \min\left(D_{K-1}(i,j),\ D_{K-1}(i,K) + D_{K-1}(K,j)\right)$$

$O(n^3)$ time, $O(n^2)$ space.

we can do inplace replacement

if $(i,j) \in E$ then $D[i,j] = w(i,j)$;
else $D[i,j] = \infty$;

$\forall\ i\quad D[i,i] = 0$;

```
for (K = 1 to n)
   for ( i = 1 to n)
      for (j = 1 to n) {
   D[i,j] = min (D[i,j], D[i,K] + D[K,j]);
   Path [i,j] = Path [i,K]; }
```

Path $[i,j] = j$ if $(i,j) \in E$ else $\boxed{-1}$

path: $\{u\}$ while (u != V){
u= Path[u, V]; path.pb(u); }