For shortest path problem, we are given a weight function $W: E \to \mathbb{R}$ where each edge is assigned a weight. We also assign a weight of infinity for edges that don't exist.

our goal is to find a path from source s to all other vertices s.t. the sum of the weights of edges along the path is minimal.

1. Bellman - Ford.

```
Relax (u,v)
    if d[v] > d[u] + w(u,v)
        d[v] ← d[u] + w(u,v)
        π[v] ← u.
```

$d[v]$ = current calculated shortest distance from s to v.

$w(u,v)$ = edge weights of (u,v)

$\pi[v]$ = the current parent of v that lead to the shortest path.

```
Initialize (V, E, s)
    for v ∈ V
        d[v] ← ∞
        π(v) = null.
        d[s] = 0
        π[s] = s.
```

```
Bellman - Ford (V, E, s)
    initialize (V, E, s)
        for i = 1 : |V|-1
            for each edge (u,v) ∈ E
                Relax (u,v).
    for each edge (u,v) ∈ E
        if d[v] > d[u] + w(u,v)
            return False
```

} this returns that ∃ a negative weight cycle.

Because the graph can only contain positive weight cycle, we would not have a cycle in our shortest path, so the shortest path have at most $|V|-1$ edges, so after $|V|-1$ iterations of relaxation, we would have the shortest path.

Running time: $O(VE)$, $(V-1)$ iterations, each iteration we relax $|E|$ edges,

Dijkstra:

In fact we can be more selective on the edges that we relax on each iteration.

2 Dijkstra:

Dijkstra (V, E, s)

initialize (V, E, s). Q is a min priority queue.

push all $v \in V \to Q$

while Q not empty

$u =$ Q.pop;

for every v s.t. $(u,v) \in E$.

relax (u, v).

So each time we only relax edges whose starting point currently have the smallest $d[u]$ value.

Runtime: $O(V \cdot (\text{extract-min}) + E \cdot (\text{decrease-key}))$.

we only look through each vertex once and relax each edge once problem 1: if we increase each edge weights by 1, we still find the shortest path.

Ans. False,

problem 2: if all edges in graph have distinct weights, shortest path are distinct.

Ans. False,

problem 3: Given graph $G = (V, E, w)$. given $\delta(s, u)$ for all $u \in V$ but we are not given $\pi(u)$ for any u, how to find the shortest path from s to a given t.

ans: start with t, one of $v \in V$ s.t. $\delta(v) + w(v,t) = \delta(t)$, then recursively work on v, the running time is $O(V+E)$ since we hit each edge and vertex at most once.

Note: A shortest path should not contain a cycle, for ex. if there exist a zero-weight cycle, the shortest path should ignore it.

problem 4: modified shortest path, if all we care is to minimize the maximum edge weight along a path, how to find shortest path.

answer: change relax function.

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

$d(u) = \max\{w(v,u), d(v)\}$