Todays Content: → {Tomorrow have Class} /
{Monday - Hashmap Implementation}

→ Bellman ford

→ Floyd warshal :

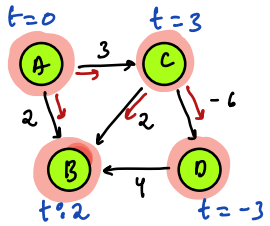
Dijkstra's : Cost of Shortest Path In +ve Weighted Graph (Directed / Undirected)

↳ Only word of the Edge weights.

-ve Edge Weights: A ——(-40)—— B

Why ? Dijkstras fail if graph as -ve edge weights also?

Ex:



: According to Dijkstras Cost of Shortest A → B

A → B : 2

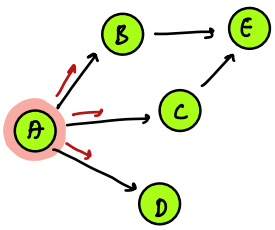
A → C → D → B : 1, correct one

Note: We need a new algo which works for both +ve & -ve Edges

Dijkstras Idea:

→ Blast the node with min value

→ Update all Adjacent nodes



Bellmon Ford / Algo

Given N nodes & E Edges

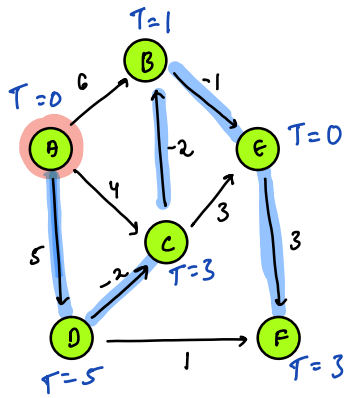
→ : For N-1 Iteration

Use every edge in any order & update nodes}

From Source Node get min cost to all Nodes

Note: -ve Edge Weights only occur in directed graphs

Bellman ford : +ve/-ve edges src = A

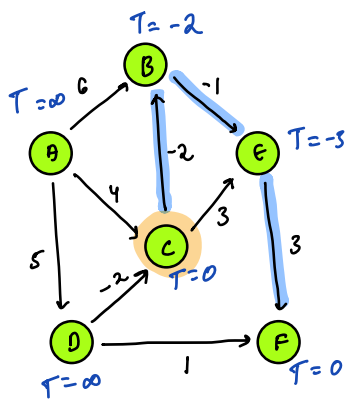


dist[ ]:	A	B	C	D	E	F
	0	∞	∞	∞	∞	∞
→ After 1:	0	2	3	5	5	6
→ After 2:	0	1	3	5	1	4
→ After 3:	0	1	3	5	0	3
After 4:	0	1	3	5	0	3

no change from 3th → 4th Break

Edges: A → B A → C A → D B → E C → E C → B D → C D → F E → F

Bellman ford : +ve/-ve edges src = C



dist[ ]:	A	B	C	D	E	F
	∞	∞	0	∞	∞	∞
After 1:	∞	-2	0	∞	3	6
After 2:	∞	-2	0	∞	-3	0
After 3:	∞	-2	0	∞	-3	0

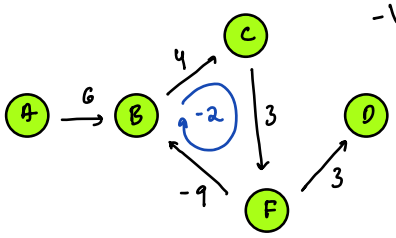
No change : Stop

Significance of ∞ : Node not reachable from Source.

A → B A → C A → D B → E C → E C → B D → C D → F E → F

Pseudo Code :

```
int[] Bellmonford ( int N, int E, int u[], int v[], int w[], int s) {
    int d[N+1] = ∞ ; d[s] = 0          TC: O(N-1) * O(E) = O(NE)
    l = 1 ; l < N ; l++) { // N-1 Iterations     SC: O(n)
        bool change = false
        i = 0 ; i < E ; i++) {
            int u = u[i], v = v[i] w = w[i]
            d[u]   w   d[v]   Cost to reach v via u = d[u] + w
            u      v
            if ( d[u] != ∞ && d[u] + w < d[v]) {
                d[v] = d[u] + w , change = True
            }
        }
        if ( change == false) {
            Break
        }
    }
    // We get length of Shortest Path from src node to all nodes
}
```



-ve cycle: In a directed weighted graph if there exists a cycle whose overall weight is < 0, it is called -ve cycle

Issue with -ve Cycle:

From A we need minimum path length to D

A B C F D : 16

A B C F B C F D : 14

A B C F B C F B C F D : 12 → 10 → 8 → 6 . . $-\infty$

↳ If we keep on iterating on -ve cycle cost keeps decreasing.

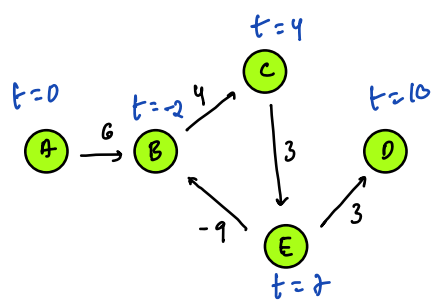
If Cost keeps decreasing min Cost path from S → D doesn't exist

Q) Given a directed weighted Graph check if it contains a -ve cycle

↓

Idea: According to Belmm ford:

- From S node, if we want to get min cost to all to all nodes
- iterate on all Edges n-1 times



	A	B	C	D	E
	0	∞	∞	∞	∞
Itr1:	0	6	∞	∞	∞
Itr2:	0	4	10	16	13
Itr3:	0	2	8	14	11
Itr4:	0	0	6	12	9

Ideally according to belmon after 4^{th} itr we need stop

Itr5: 0 -2 4 10 7 : It further reduced -ve cycle exist

Edges: B → C ✓ C → E ✓ A → B ✓ E → B ✓ E → D ✓

```
int[] Bellmonford ( int N, int E, int u[], int v[], int w[], int s) {
    int d[N+1] = ∞ ; d[s] = 0          TC: O(N*E)  SC: O(N)
    l = 1 ; l <= N ; l++) { // N iteration
        bool change = false
        i = 0 ; i < E ; i++) {
            int u = u[i], v = v[i] w = w[i]
            d[u]  w  d[v]   Cost to reach v via u = d[u] + w
            u     v
            if ( d[u] != ∞ && d[u] + w < d[v]) {
                d[v] = d[u] + w , change = True
            }
        }
        if (l == N && change = True) { return True }
        if (change == False) { return False // no cycle }
    }
    // We get length of Shortest Path from src node to all nodes
}
```

Directed Graph : Topological Sort Bellmonford

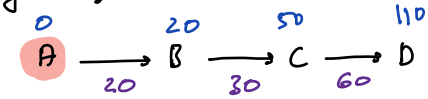
✓ ✓ → -ve Cycle

✓ x → +ve cycle

Weighted Graph :

	Directed	Undirected
+ve Edge	Dijkras O(E log E) or Bellmmford O(NE)	Dijkras O(E log E) or Bellmonford O(NE)
-ve Edge	Bellmon ford	-ve Edge weight are not used as undirected graphs

Why n-1? :



: Src node = A

Edges: C → D ✓
B → C ✓
A → B ✓

dist:

	A	B	C	D
	0	∞	∞	∞
→ Itr1 :	0	20	∞	∞
→ Itr2 :	0	20	50	∞
→ Itr3 :	0	20	50	110

Obs: In bellman ford after each iteration, for atleast 1 node we get its min cost from src node

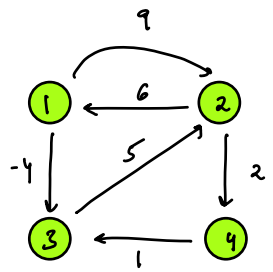
Floyd Warshall:

: All pairs shortest path

: From every node, calculate shortest path to all other nodes

Eg:

Eg:



dist[5][5]: no intermediate nodes

	0	1	2	3	4
0					
1					
2					
3					
4					

```
int[][] floydWarshall(int N, int E, int u[], int v[], int w[]){

}
```

Classification :

Direction	Weighted	Negative	Shortest Path/Cost
*	*	*	BFS :
✓	*	*	BFS
*	✓	*	Dijkstras → Sing Source : $E \log E$ Bellmon ford : Single Source : NE Floyd Warshal : All pair shortest : N^3
✓	✓	*	Dijkstras → $E \log E$ Bellmon ford : Single Source : NE Floyd Warshal : All pair shortest : N^3
*	✓	✓	not possible
✓	✓	✓	Bellmon ford : Single Source Floyd Warshal : All pair shortest