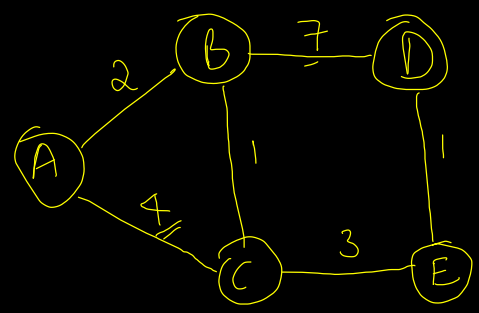
"Work hard every day, not because success is guaranteed, but because effort builds discipline, and discipline opens doors to opportunities you can't even imagine yet. Every page you read, every problem you solve, every hour you dedicate adds up. Stay consistent. Your future self will thank you for not giving up today."

Dijkstra's Algo

```
const graph = {
    A: {B: 2, C: 4},
    B: {A: 2, C: 1, D: 7},
    C: {A: 4, B: 1, E: 3},
    D: {B: 7, E: 1},
    E: {C: 3, D: 1},
}
```

A → E



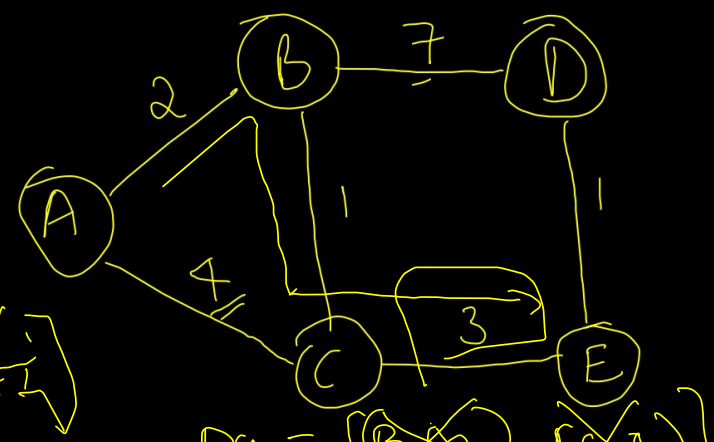
1. visited = [ ]
   unvisited nodes

2. node , distance — pq (priority Queue)
   - Check current node to its neighbours & update shortest distance & if distance of newly found node is shorter add to pq for further exploration
   - Repeat until all pq elements are done & all nodes are visited

→ Weights of edges should be +ve only

→

```
const graph = {
    A: {B: 2, C: 4},
    B: {A: 2, C: 1, D: 7},
    C: {A: 4, B: 1, E: 3},
    D: {B: 7, E: 1},
    E: {C: 3, D: 1},
}
```



Step1 A → E

unvisited = (A, B, C, D, E)

visited = (A, B, C)

pq = [(B, ∞), (C, ∞)]

pq - Minheap

Step2

Node	Distance	PrevNode
A	0	—
B	∞	—
C	∞	—
D	∞	—
E	∞	—

Step 3 =

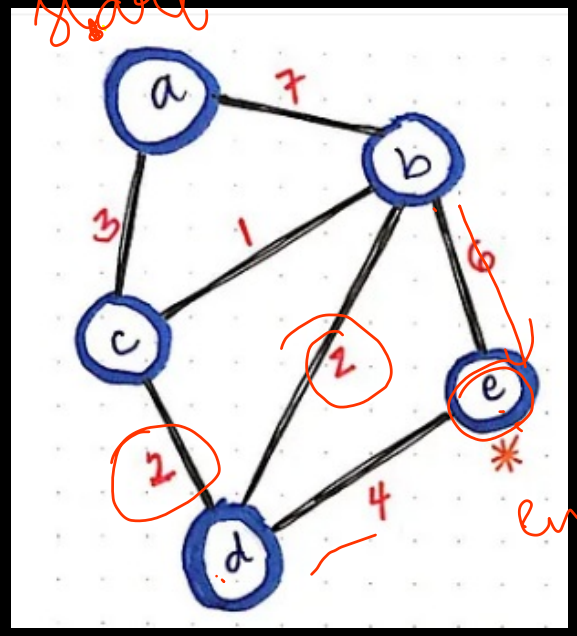
A	0	—
B	2	A
C	3 ~~4~~ 10	A B
D	~~9~~ 7+2	~~B~~ 2
E	3+3	C

pq = [(D, 7), (E, 3)]

A	0	—
B	2	A
C	3	B
D	9	B
E	6	C

A → B → C → E

start



a → e.

unvisited = [b, c]

visited = [a, b, c, d, e]

end

unvisited = [d, c]

Node	Distance	Prev Node
a	0	—
→ b	7 ~~∞~~	A
c	3 ~~∞~~	A
d	~~∞~~ 2+3 = 5	c
e	~~∞~~ 6 7	B d

d → e
5 + 4 → e.
9.

b → e
7 + 6
=

a → d. e

a → c → d → e
3 5
9.