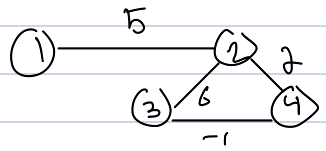
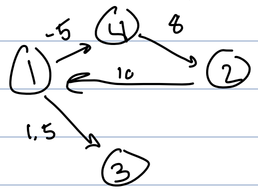
Weighted graph: $G = (V, E)$, $W: E \to \mathbb{R}$

↳ Graph + Weight function



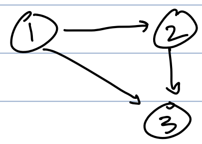
- Edges have weight.
- Undirected, weighted

- Can have directed, weighted graph



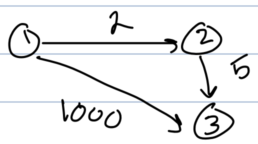
Computing distance:

Unweighted:



shortest = $1 \to 3$, use BFS

weighted:



shortest = $1 \to 2 \to 3$

Dijkstra's Algorithm

- Used for computing shortest paths from a node to all other nodes on a directed graph with positive edge weights.

- inputs: G, W, S
  - G — directed graph
  - W — weight function
  - S — source

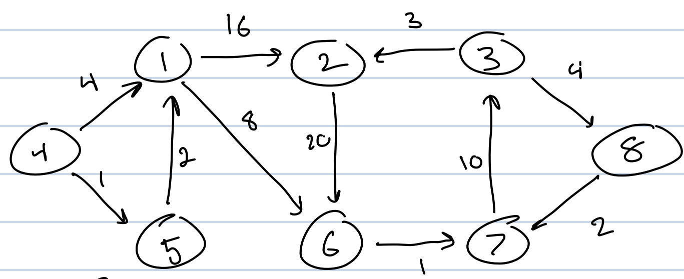
☆ Dijkstra(G, w, s)

```
for each u ∈ G.V:
    u.d = ∞   // hold distance from s → u, goal is it's the shortest
    u.p = NIL
s.d = 0
Q = Ø   // empty min-queue
S = Ø   // empty set; used for proof
for each u ∈ G.V
    INSERT(Q, u)   // use u.d for pointer
```

```
While Q ≠ ∅
    u = Extract-Min(Q)
    for each out-neighbor v of u:
        if v.d > u.d + w(u,v)
            v.d = u.d + w(u,v)
            v.p = u
            DECREASE-KEY(Q, v, v.d)
    S = S ∪ {u}
```

$= O((m+n)\log n)$

Ex)



	d	p
1	3	5
2	∞	NIL
3	∞	NIL
4	0	NIL
5	1	4
6	∞	NIL
7	∞	NIL
8	∞	NIL

S = {4 }

Q = 1, 2, 3, 6, 7, 8

u = 5

S = 4