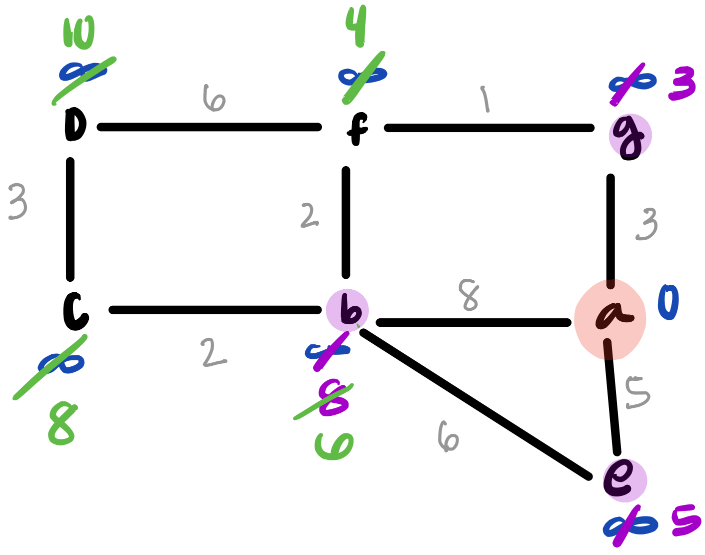
Dijkstra's — shortest path from one node to all nodes.



Vertex	shortest distance from A	prev. v
A	0	
B	~~8~~ 6	~~A~~ F
C	8	B
D	10	F
E	5	A
F	4	G
G	3	A

1) Pick node, usually alphab. and it = 0 and all other nodes = ∞ b/c. they haven't been visited yet.

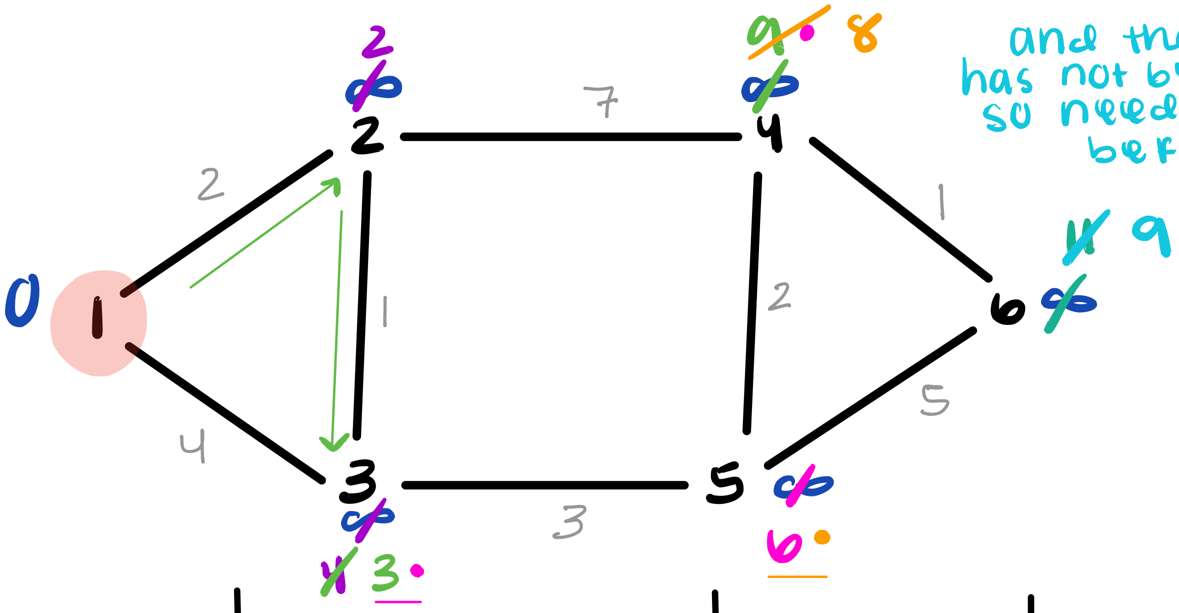
2) can go from a to all nodes it is attached to in order of weight and then assign those nodes = to the edge weight.

3) Then choose the node with the smallest value and repeat step 2 for all nodes attached and replace their value to

4) then the distance from A to current node would be the distance value

Dijkstra's

shortest path from one node to all nodes.



and then 4-6 has not been checked so need to check before done.

Vertex	Shortest d	prev v.	Visited order
1	0	x	1
2	~~∞~~ 2	1	2
3	~~∞~~ ~~4~~ 3	~~1~~ 2	3
4	~~∞~~ ~~9~~ 8	~~2~~ 5	5
5	~~∞~~ 6	3	4
6	~~∞~~ ~~11~~ 9	~~5~~ 4	6

1) start v. =0, others = ∞

2) check every vertex attached to initial vertex

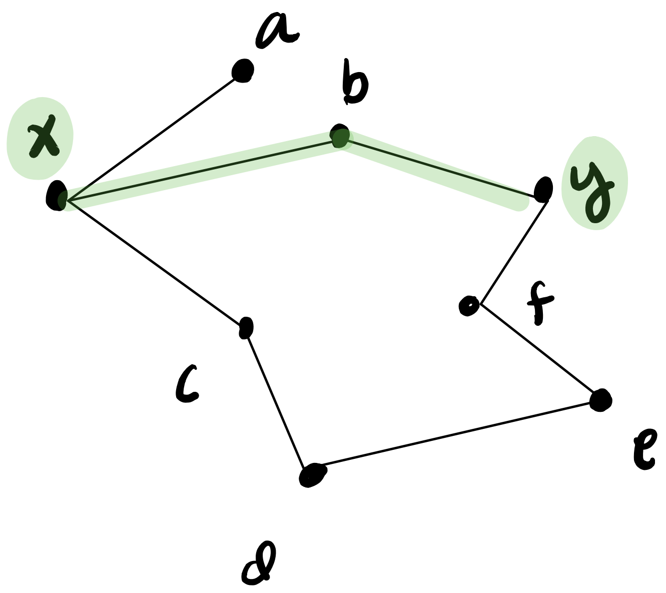
3) pick min value from attached vertices and updated vertex value if shorter path exists

4) Repeat step 3 with final vertices from step 3 and continue until final node.

Note: you need to check all attached vertices from current node before it can be officially "visited" (esp. node 4)

SSSP:

BFS	G is unweighted	$O(v+e)$
DFS	G is DAG	$O(v+e)$
Dijkstra	no negative edges	$O(n^2)$ $O(e \log v)$
Bellman-F.		$O(v \cdot e)$ ↳ $= O(v^2)$



finds shortest path from node x to y but also to all other nodes.