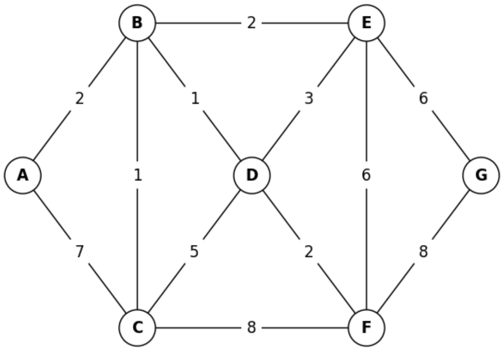
1 Dijjkstra's Shortest Path



Using Dijkstra's algorithm, find the shortest path from node A to G. Please provide a table which shows the path weight and predecessor from A to every node, labelling the visited node at each step.

Solution

iteration	node visited	A	B	C	D	E	F	G
0	A	start:0	A: 2	A: 7	none	none	none	none
1	B	start:0	A: 2	B: 3	B: 3	B: 4	none	none
2	C	start:0	A: 2	B: 3	B: 3	B: 4	C: 11	none
3	D	start:0	A: 2	B: 3	B: 3	B: 4	D: 5	none
4	E	start:0	A: 2	B: 3	B: 3	B: 4	D: 5	E: 10
5	F	start:0	A: 2	B: 3	B: 3	B: 4	D: 5	E: 10

The path with min weight is: $G \leftarrow E \leftarrow B \leftarrow A$

The solution immediately above is sufficient for full credit.A step-by-step explanation is given below:

- Start algorithm: visit start node A
- add path $A \rightarrow B$ with cost 2 (no previous path to B)
- add path $A \rightarrow C$ with cost 7 (no previous path to C)

iteration	node visited	A	B	C	D	E	F	G
0	A	start:0	A: 2	A: 7	none	none	none	none

- Continue algorithm: visit node with min cost among unvisited: B has cost 2
- update to path A → B → C with cost 3 (previous path A → C had cost 7)
- add path A → B → D with cost 3 (no previous path to D)
- add path A → B → E with cost 4 (no previous path to E)

iteration	node visited	A	B	C	D	E	F	G
1	B	start:0	A: 2	B: 3	B: 3	B: 4	none	none

- Continue algorithm: visit node with min cost among unvisited: C has cost 3
- ignore path A → C → D with cost 8 (previous path A → B → D had cost 3)
- add path A → C → F with cost 11 (no previous path to F)

iteration	node visited	A	B	C	D	E	F	G
2	C	start:0	A: 2	B: 3	B: 3	B: 4	C: 11	none

- Continue algorithm: visit node with min cost among unvisited: D has cost 3
- ignore path A → ... → D → E with cost 6 (previous path A → B → E had cost 4)
- update to path A → ... → D → F with cost 5 (previous path A → C → F had cost 11)

iteration	node visited	A	B	C	D	E	F	G
3	D	start:0	A: 2	B: 3	B: 3	B: 4	D: 5	none

- Continue algorithm: visit node with min cost among unvisited: E has cost 4
- ignore path A → ... → E → F with cost 10 (previous path A → ... → D → F had cost 5)
- add path A → ... → E → G with cost 10 (no previous path to G)

iteration	node visited	A	B	C	D	E	F	G
4	E	start:0	A: 2	B: 3	B: 3	B: 4	D: 5	E: 10

- Continue algorithm: visit node with min cost among unvisited: F has cost 5
- ignore path A → ... → F → G with cost 13 (previous path A → ... → E → G had cost 10)
- End algorithm: next to visit is destination: G has cost 10

iteration	node visited	A	B	C	D	E	F	G
5	F	start:0	A: 2	B: 3	B: 3	B: 4	D: 5	E: 10