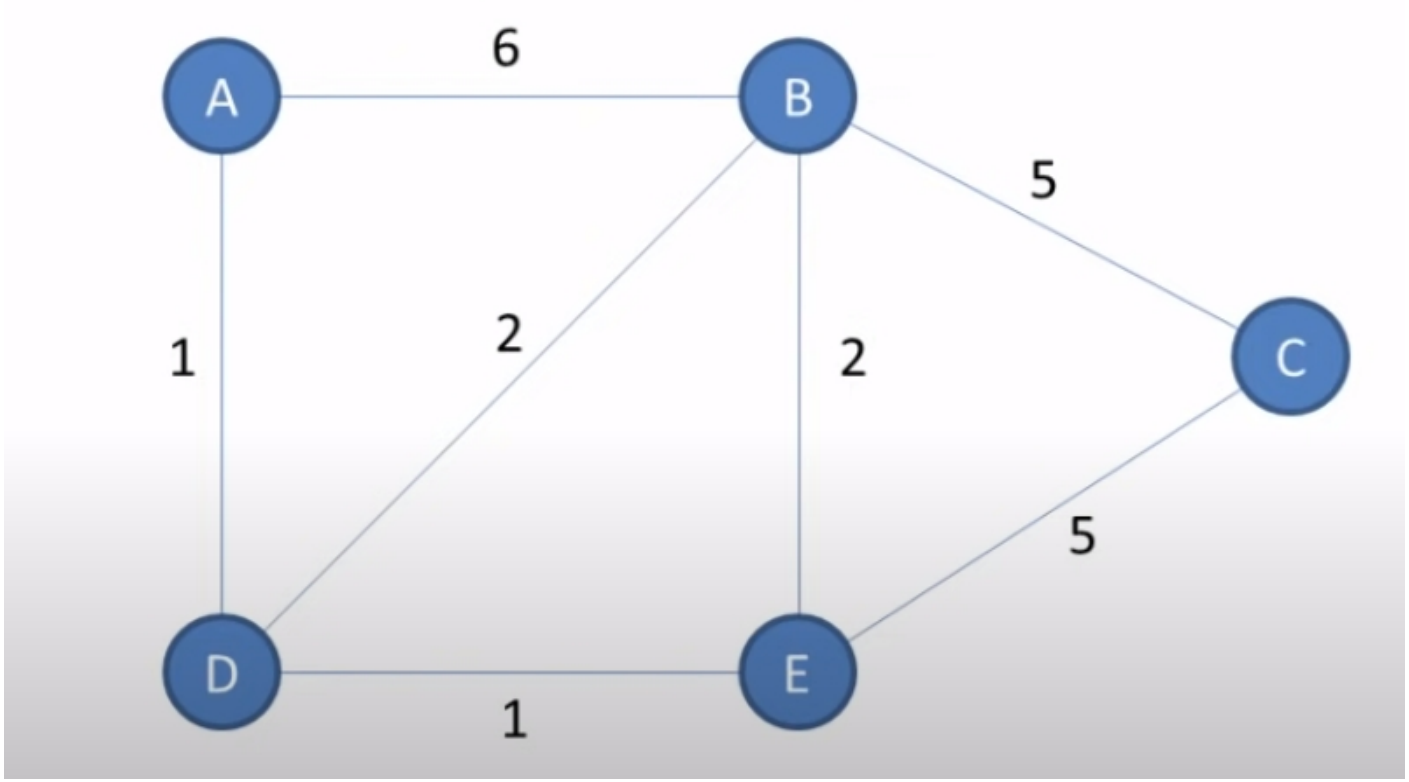
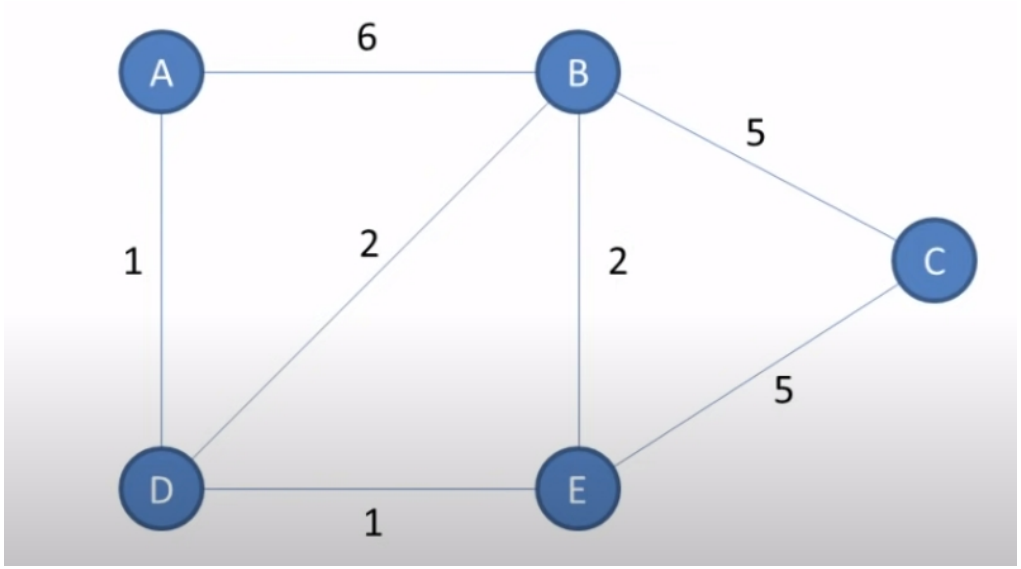
Weighted Graph



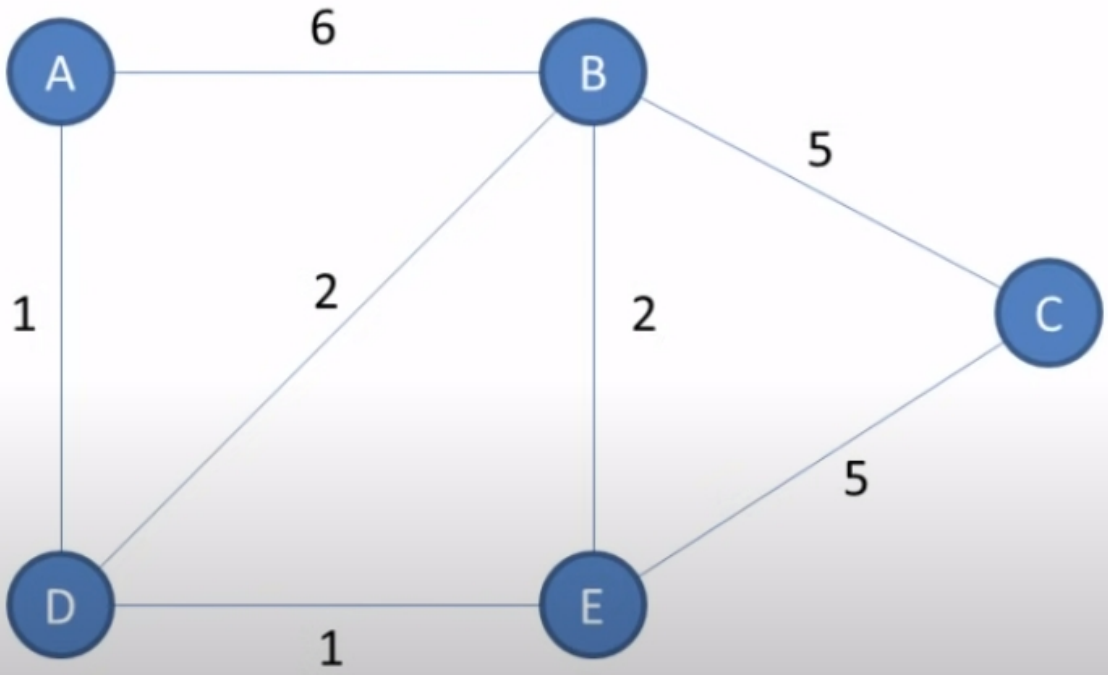
Find all the Shortest Path from Source A:



Vertex	Shortest distance from A	Previous Vertex
A	0	
B	?	
C	?	
D	?	

E	?	

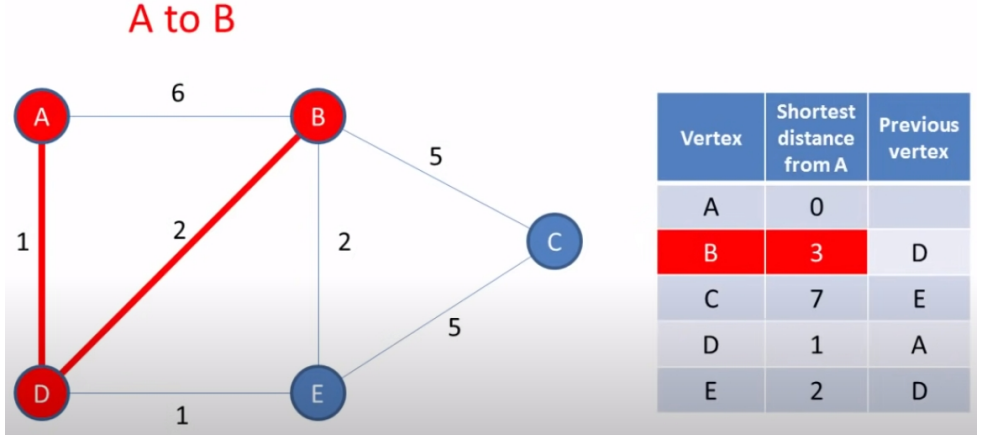
Output:



Vertex	Shortest distance from A	Previous vertex
A	0	
B	3	D
C	7	E
D	1	A
E	2	D

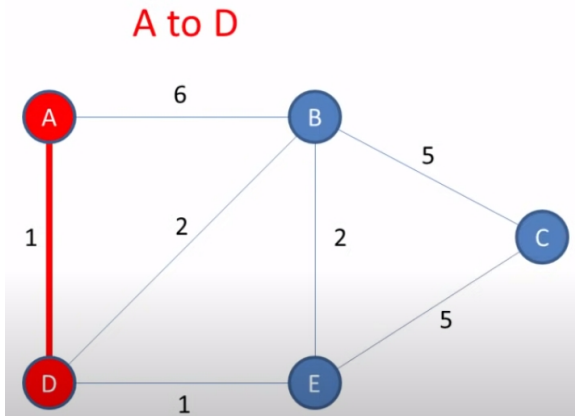
Illustrations:

A to B



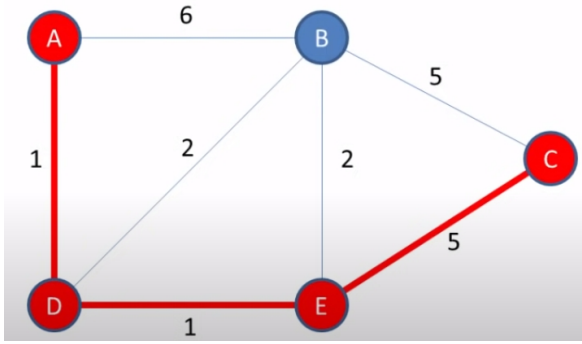
Vertex	Shortest distance from A	Previous vertex
A	0	
B	3	D
C	7	E
D	1	A
E	2	D

A to D



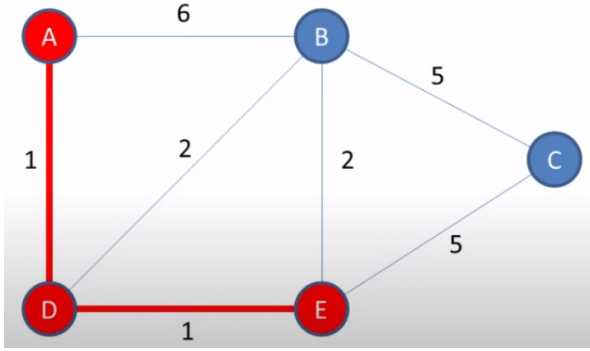
vertex	Shortest distance from A	Previous vertex
A	0	
B	3	D
C	7	E
D	1	A
E	2	D

A to C



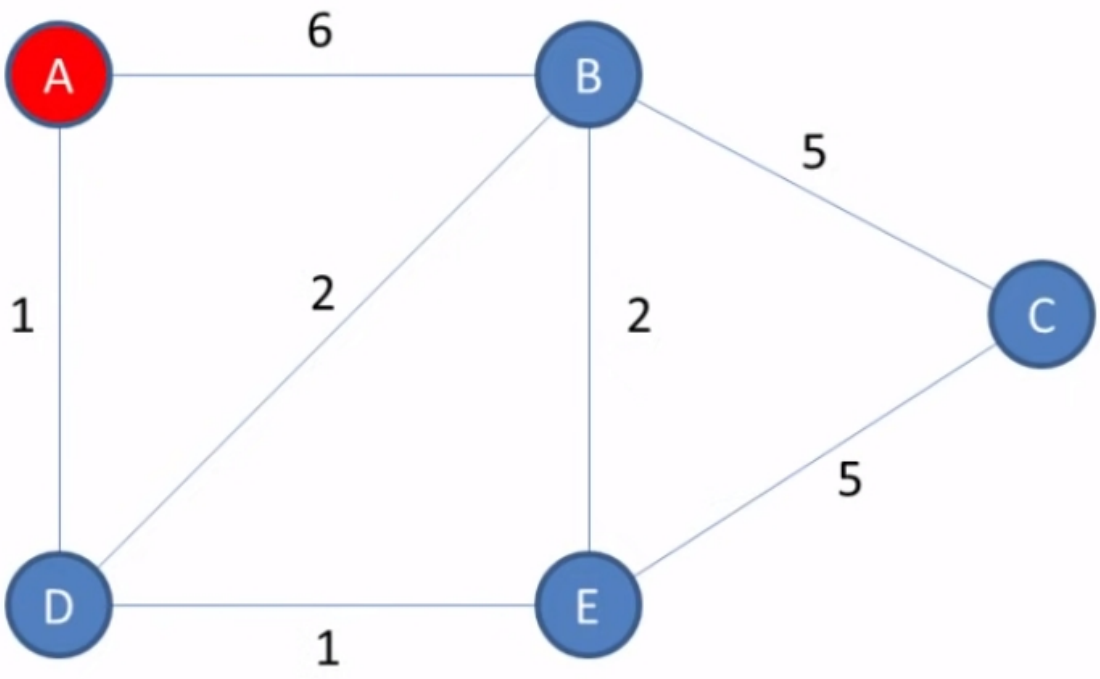
Vertex	Shortest distance from A	Previous vertex
A	0	
B	3	D
C	7	E
D	1	A
E	2	D

A to E



Vertex	Shortest distance from A	Previous vertex
A	0	
B	3	D
C	7	E
D	1	A
E	2	D

Working: Dijkstra's Algorithm



Vertex	Shortest distance from A	Previous vertex
A		
B		
C		
D		
E		

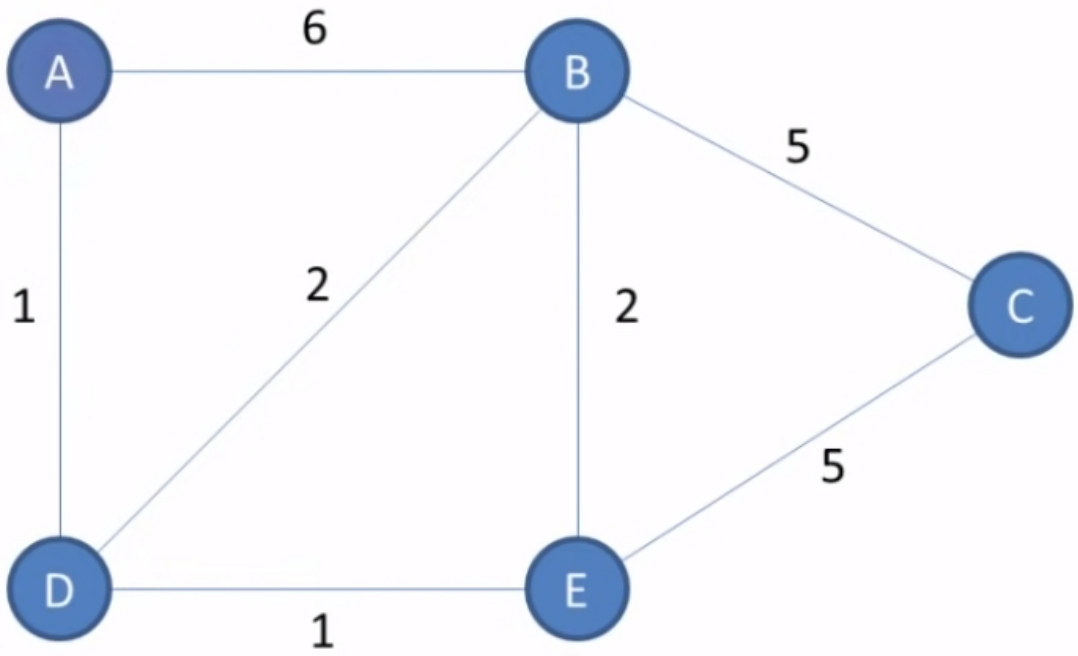
Visited = [ ]

Unvisited = [A, B, C, D, E]

Distance from:

>> A to A = 0 and

>> to others are unknown, set to ∞.

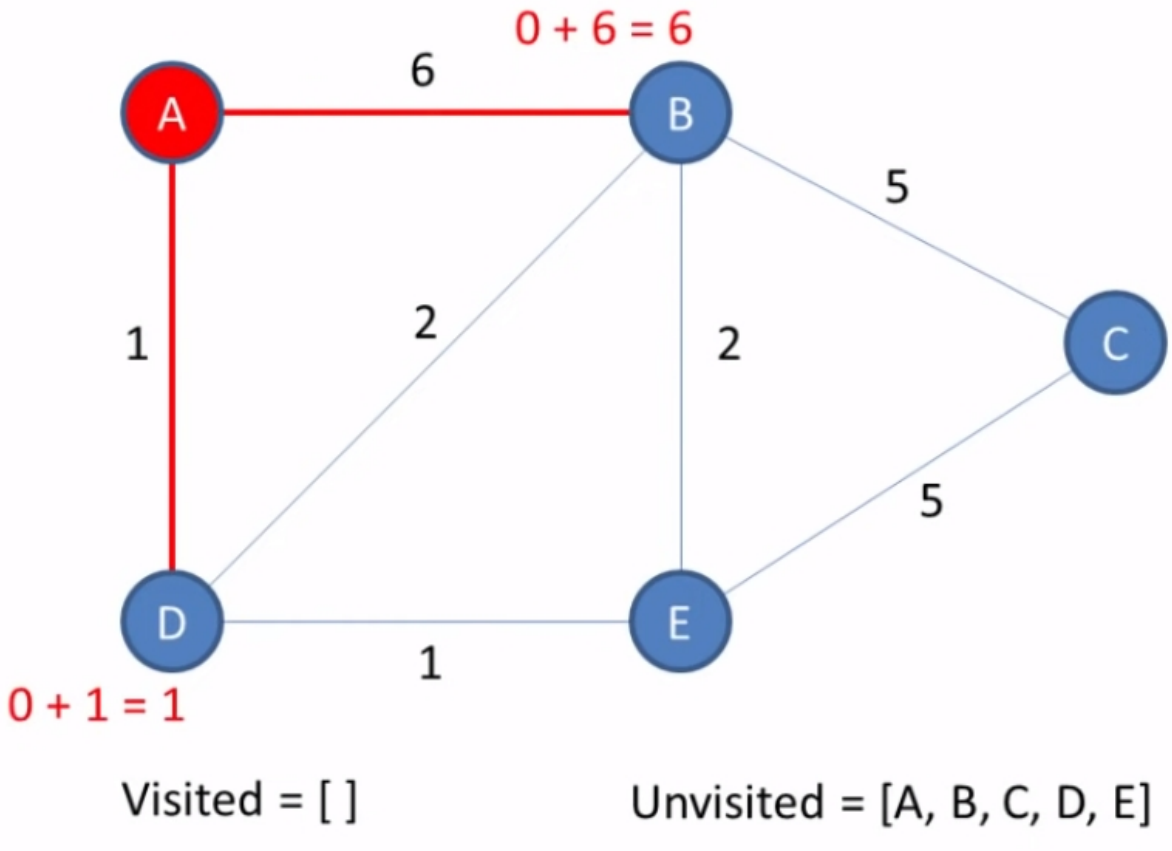


Vertex	Shortest distance from A	Previous vertex
A	0	
B	∞	
C	∞	
D	∞	
E	∞	

Visited = [ ]

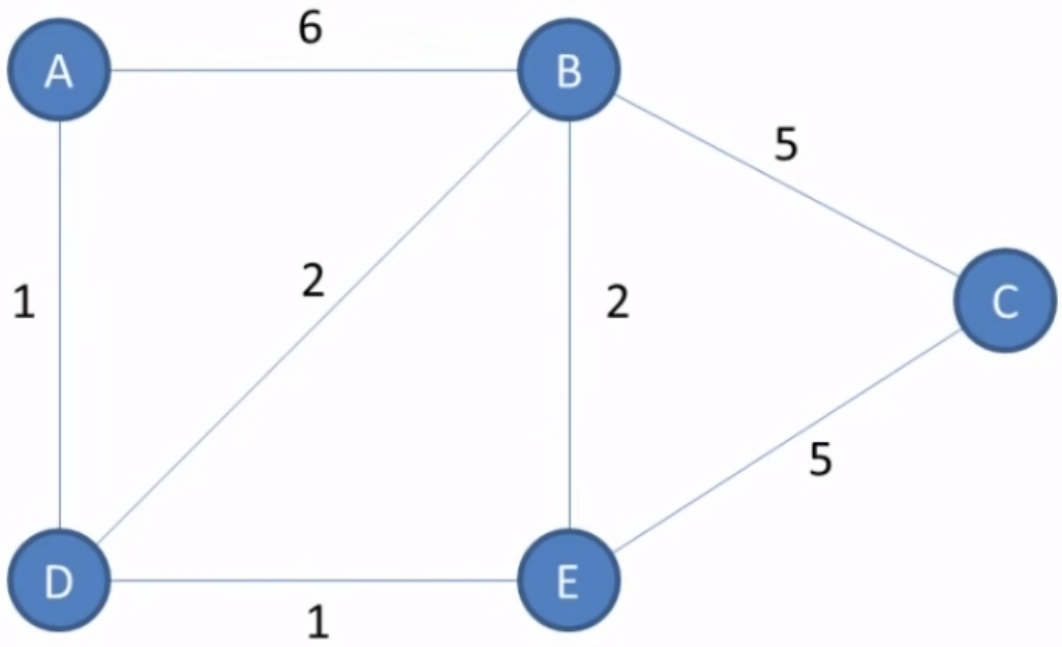
Unvisited = [A, B, C, D, E]

Start from 0, calculate the distance of its neighbors:



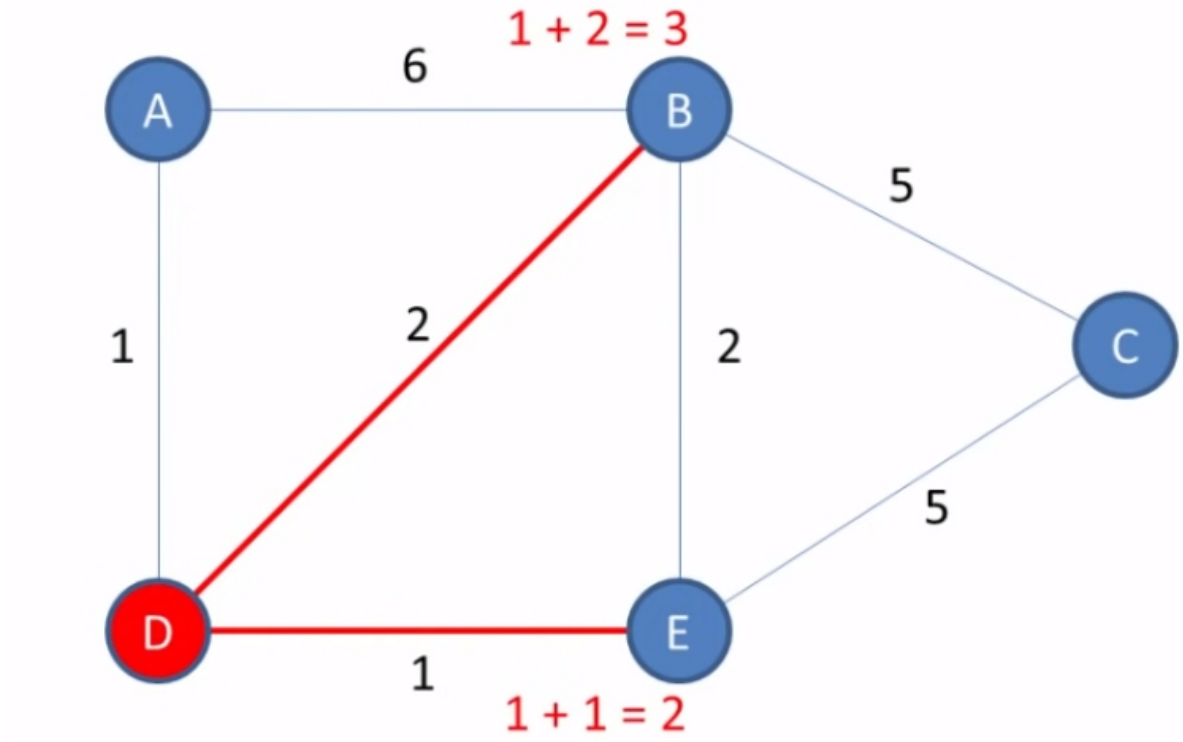
Vertex	Shortest distance from A	Previous vertex
A	0	
B	∞	
C	∞	
D	∞	
E	∞	

The calculated distance is less than that of known distance, we update the distance in the table and also update the previous vertices.



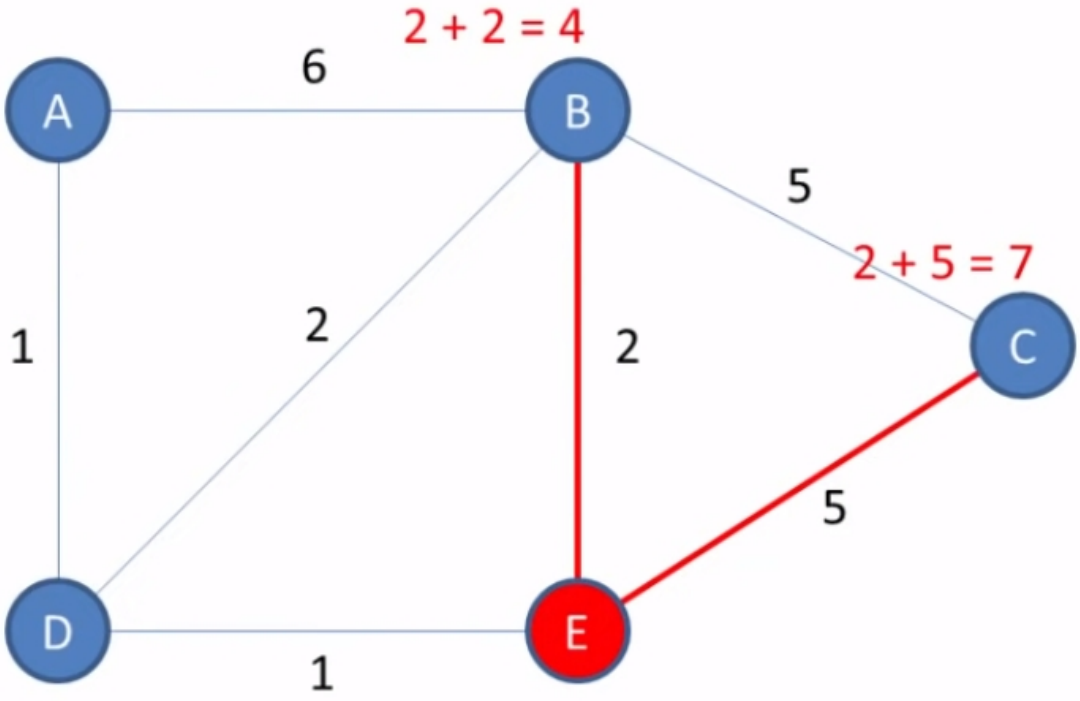
Vertex	Shortest distance from A	Previous vertex
A	0	
B	6	A
C	∞	
D	1	A
E	∞	

Now the algorithm begins to repeat, It starts visiting again from the smallest known distance of the visited vertex, i.e. D.



Vertex	Shortest distance from A	Previous vertex
A	0	
B	3	A
C	∞	
D	1	A
E	2	

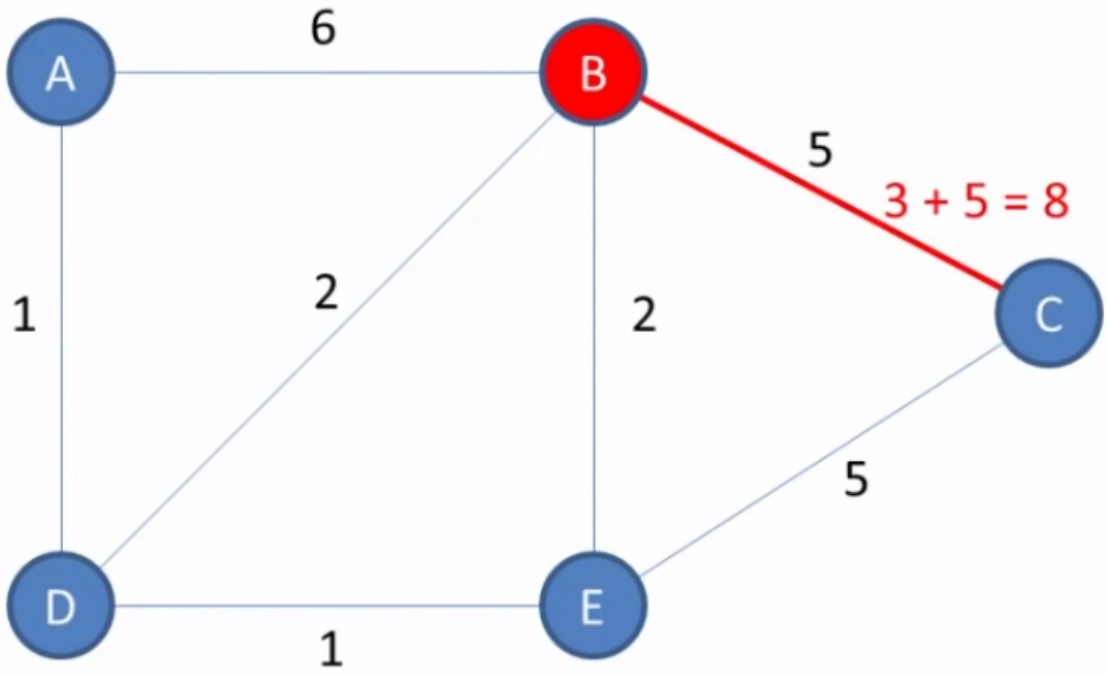
Again start visiting from smallest know distance, i.e. E.



Vertex	Shortest distance from A	Previous vertex
A	0	
B	3	D
C	7	
D	1	A
E	2	D

We do not update B (It is already small), but update C.

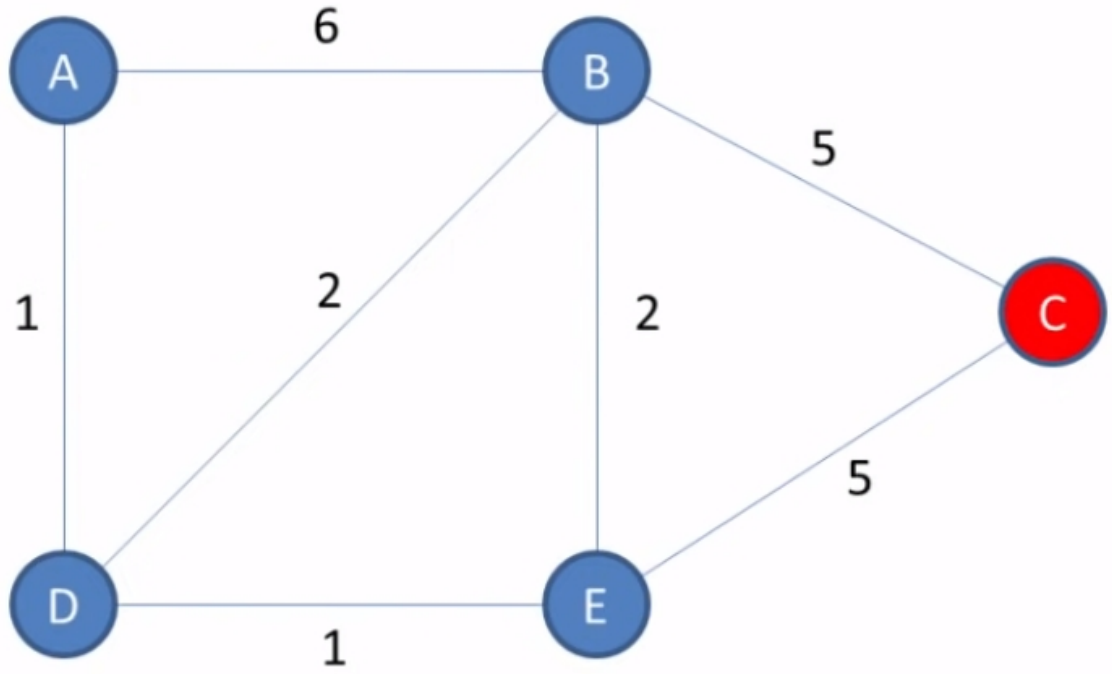
Again, for smallest path vertex now is B, only a neighbor C is not visited.



Vertex	Shortest distance from A	Previous vertex
A	0	
B	3	D
C	7	E
D	1	A
E	2	D

We do not need to update C, but update to visited list.

Again, there a vertex C, i.e. not visited.



Vertex	Shortest distance from A	Previous vertex
A	0	
B	3	D
C	7	E
D	1	A
E	2	D

Nothing do here. And no more vertices are left.

Final algorithm: Dijkstra's Shortest Path

Let the distance from start vertex from stat = **0**

Set, the distance over other vertices = ∞ (**a large value**)

do,

a. Visit the unvisited vertex with the smallest known distance from the start vertex.

b. FOR each unvisited neighbors for current vertex,

    If calculated distance < known distance, then

        1. Update the shortest distance.

        2. Update the previous vertex the current one.

    End If.

    NEXT unvisited neighbor.

c. Add the current vertex to the list of visited vertices.

while vertex remain unvisited.