2 -> 4 6 8 10

3 -> 6 9

4 -> 8

5 -> 10

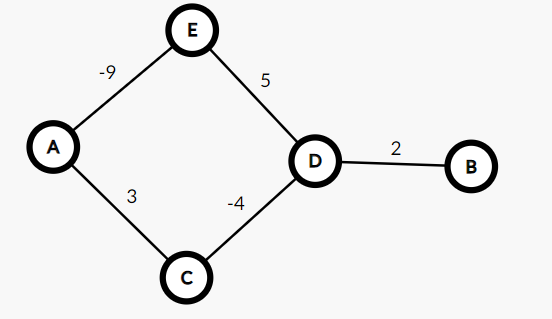
6 ->

7 ->

8 ->

9 ->

10 ->



5 5

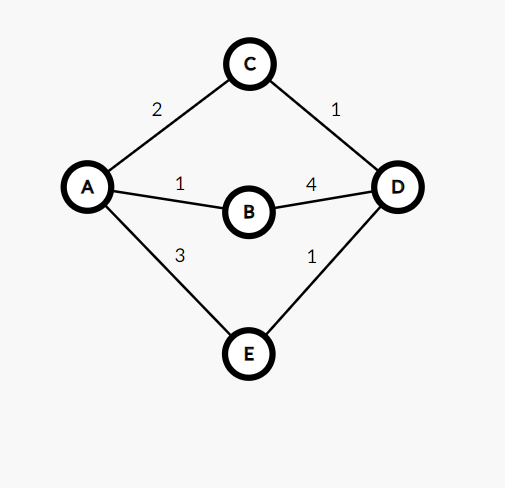
A E -9

A C 3

D E 5

D C -4

D B 2



6 6

A C 2

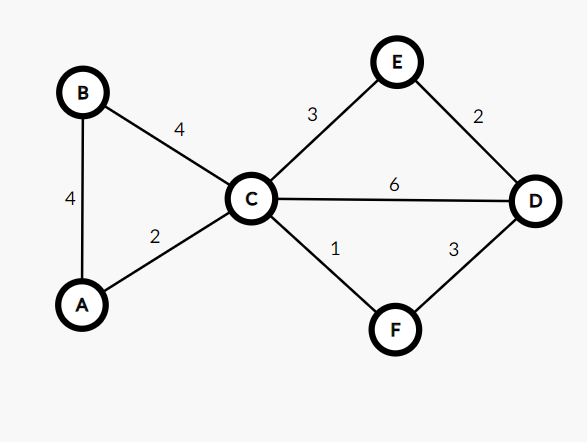
A B 1

A E 3

D E 1

D B 4

D C 1



Simulate **Dijkstra**’s algorithm (the optimised version) on the following graph:  
Use **node-B** as source.

6 8

B C 4

B A 4

A C 2

C D 6

C E 3

C F 1

D E 2

D F 3

B: (C,4), (A,8)

C: (B,4), (A,6), (D,2), (E,3), (F,1)

D: (C,2), (E,2), (F,3)

E: (D,2), (C,3)

F: (D,3), (C,1)

6 8

2 3 4

2 1 4

1 3 2

3 4 6

3 5 3

3 6 1

4 5 2

4 6 3

2: (3,4), (1,8)

3: (2,4), (1,6), (4,2), (5,3), (6,1)

4: (3,2), (5,2), (6,3)

5: (4,2), (3,3)

6: (4,3), (3,1)

Sure! Here's how Dijkstra's algorithm would work on the given graph, starting from node B:

1. Create a distance table with all nodes and set their tentative distances to infinity, except for the source node B which is set to 0.

| **Node** | **Tentative Distance** | **Visited?** |
| --- | --- | --- |
| B | 0 | No |
| A | Infinity | No |
| C | Infinity | No |
| D | Infinity | No |
| E | Infinity | No |
| F | Infinity | No |

1. Set the current node to B and mark it as visited.

| **Node** | **Tentative Distance** | **Visited?** |
| --- | --- | --- |
| B | 0 | Yes |
| A | Infinity | No |
| C | Infinity | No |
| D | Infinity | No |
| E | Infinity | No |
| F | Infinity | No |

1. For each of the current node's neighbors (C and A), calculate their tentative distances by adding the distance from the current node to the neighbor to the current node's tentative distance. If this tentative distance is less than the neighbor's current tentative distance, update the neighbor's tentative distance in the distance table.

| **Node** | **Tentative Distance** | **Visited?** |
| --- | --- | --- |
| B | 0 | Yes |
| A | 4 | No |
| C | 4 | No |
| D | Infinity | No |
| E | Infinity | No |
| F | Infinity | No |

1. Set the current node to C and mark it as visited.

| **Node** | **Tentative Distance** | **Visited?** |
| --- | --- | --- |
| B | 0 | Yes |
| A | 4 | No |
| C | 4 | Yes |
| D | Infinity | No |
| E | Infinity | No |
| F | Infinity | No |

1. For each of the current node's neighbors (B, A, D, E, and F), calculate their tentative distances by adding the distance from the current node to the neighbor to the current node's tentative distance. If this tentative distance is less than the neighbor's current tentative distance, update the neighbor's tentative distance in the distance table.

| **Node** | **Tentative Distance** | **Visited?** |
| --- | --- | --- |
| B | 0 | Yes |
| A | 4 | No |
| C | 4 | Yes |
| D | 10 | No |
| E | 7 | No |
| F | 5 | No |

1. Set the current node to F and mark it as visited.

| **Node** | **Tentative Distance** | **Visited?** |
| --- | --- | --- |
| B | 0 | Yes |
| A | 4 | No |
| C | 4 | Yes |
| D | 10 | No |
| E | 7 | No |
| F | 5 | Yes |

1. For each of the current node's neighbors (C and D), calculate their tentative distances by adding the distance from the current node to the neighbor to the current node's tentative distance. If this tentative distance is less than the neighbor