Adjacency List Representation:

- Vertex A:
  - A → B (4)
  - A → C (2)
- Vertex B:
  - B → C (5)
  - B → D (10)
- Vertex C:
  - C → D (3)
- Vertex D:
  - No outgoing edges.

Step 1: Initialization

We'll apply Dijkstra's Algorithm to find the shortest path from source vertex A to all other vertices.

Initial Setup:

- Set the distance to the source vertex A as 0, and the distance to all other vertices as ∞.
- The graph has four vertices: A, B, C, D.

Vertex	Distance
A	0
B	∞
C	∞
D	∞

Step 2: Apply Dijkstra's Algorithm
Iteration 1: Start from Vertex A

- The current vertex is A with a distance of 0.
- Update the distances to the neighbors of A:

- A → B (4): Distance(B)=0+4=4
- A → C (2): Distance(C)=0+2=2

- **Updated Distance Table:**

Vertex	Distance
A	0
B	4
C	2
D	∞

Iteration 2: Choose the next vertex (C)

- The smallest distance among unvisited vertices is 2 (from vertex C).
- The current vertex is C with a distance of 2.
- Update the distances to the neighbors of C:
  - C → D (3): Distance(D)=2+3=5
- Updated Distance Table:

Vertex	Distance
A	0
B	4
C	2
D	5

Iteration 3: Choose the next vertex (B)

- The smallest distance among unvisited vertices is 4 (from vertex B).
- The current vertex is B with a distance of 4.
- Update the distances to the neighbors of B:
  - B → C (5): Distance(C)=4+5=9
  - Distance(C)=4+5=9 (but Distance(C)=2, so no update is needed).
  - B → D (10): Distance(D)=4+10=14 (but Distance(D)=5, so no update is needed).

- Updated Distance Table (no change in distances):

Vertex	Distance
A	0
B	4
C	2
D	5

Iteration 4: Choose the next vertex (D)

- The smallest distance among unvisited vertices is 5 (from vertex D), and there are no more neighbors to update.
- The algorithm terminates.

Final Shortest Distances from Source A:

Vertex	Distance from A
A	0
B	4
C	2
D	5

Conclusion
Using Dijkstra's Algorithm, we found the shortest distances from source vertex A to all other vertices:

- A → A: Distance = 0
- A → B: Distance = 4
- A → C: Distance = 2
- A → D: Distance = 5