Continuous Evaluation (CE3)

Write a C++ program to implement Dijkstra Algorithm. Design your algorithm for the network example of 7 nodes discussed in class. Your program should input a 2D cost matrix of size 7x7. The input matrix should be a symmetric matrix as cost from a to b is same as cost from b to a. In a specific column, the input matrix should have a finite value only for neighbours. All other values should be infinite (or 100).

Your code structure may look like the following:

Function	Utility
void main()	To intialize variables and call functions.
void input()	To input 2D cost matrix.
void costupdate()	Assume a root node. Initialize two 1D arrays (s and d). Each array is of size 7. The 1D array d is used to store updated cost. Implement Dijkstra algorithm and find d.
void display()	To display the final distance vectors.

Your final output may look like the following:

Enter cost (max:100) of the node 0 from 0

0

Enter cost (max:100) of the node 1 from 0

2

Enter cost (max:100) of the node 2 from 0

100

Enter cost (max:100) of the node 3 from 0

3

Enter cost (max:100) of the node 4 from 0

100

Enter cost (max:100) of the node 5 from 0

100

Enter cost (max:100) of the node 6 from 0

100

Enter cost (max:100) of the node 1 from 1

0

Enter cost (max:100) of the node 2 from 1

5

Enter cost (max:100) of the node 3 from 1

100

Enter cost (max:100) of the node 4 from 1

4

Enter cost (max:100) of the node 5 from 1

100

Enter cost (max:100) of the node 6 from 1

100

Enter cost (max:100) of the node 2 from 2

0

Enter cost (max:100) of the node 3 from 2

100

Enter cost (max:100) of the node 4 from 2

100

Enter cost (max:100) of the node 5 from 2

4

Enter cost (max:100) of the node 6 from 2

3

Enter cost (max:100) of the node 3 from 3

0

Enter cost (max:100) of the node 4 from 3

5

Enter cost (max:100) of the node 5 from 3

100

Enter cost (max:100) of the node 6 from 3

100

Enter cost (max:100) of the node 4 from 4

0

Enter cost (max:100) of the node 5 from 4

2

Enter cost (max:100) of the node 6 from 4

100

Enter cost (max:100) of the node 5 from 5

0

Enter cost (max:100) of the node 6 from 5

1

Enter cost (max:100) of the node 6 from 6

0

The input cost matrix:

0 2 100 3 100 100 100

2 0 5 100 4 100 100

100 5 0 100 100 4 3

3 100 100 0 5 100 100

100 4 100 5 0 2 100

100 100 4 100 2 0 1

The updated cost matrix:

0 2 7 3 6 8 9

2 0 5 5 4 6 7

7 5 0 10 6 4 3

3 5 10 0 5 7 8

6 4 6 5 0 2 3

8 6 4 7 2 0 1

9 7 3 8 3 1 0