Bellman Ford Algorithm Tracing

main()

num_ver [ ] Source [ ] A[ ][ ] Max_Value = 999

Enter number of vertices

num_ver [5]

Enter the adjacency matrix

A[sn][dn] => A[5][5]

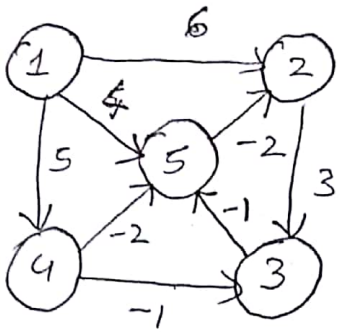
A[1][1] = 0 // indexing starts from 1

A[1][2] = 6

A[1][3] = 0

A[1][4] = 5

A[1][5] = 4



A[2][1] = 0 A[3][1] = 0 A[4][1] = 0 A[5][1] = 0

A[2][2] = 0 A[3][2] = 0 A[4][2] = 0. A[5][2] = -2

A[2][3] = 3 A[3][3] = 0 A[4][3] = -1 A[5][3] = 0

A[2][4] = 0 A[3][4] = 0 A[4][4] = 0 A[5][4] = 0

A[2][5] = 0 A[3][5] = -1 A[4][5] = -2 A[5][5] = 0

if A[sn][dn] == 0

then A[sn][dn] = Max_Value // 999

All values of adjacency matrix are filled with 999 where A[sn][dn] = 0, other than condition where sn == dn,

if sn == dn

then A[sn][dn] = 0

A[sn][dn]	1	2	3	4	5
1	0	6	999	5	4
2	999	0	3	999	999
3	999	999	0	999	-1
4	999	999	-1	0	-2
5	999	-2	999	999	0

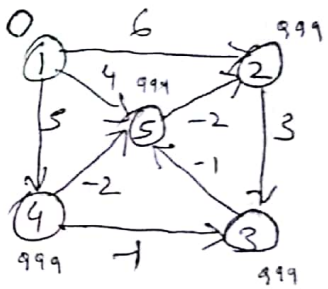
Enter source vertex:
source = 1

BellmanFord(5) // calls constructor
{
  D[ ] = 5+1 = 6 // D[6]
}

BellmanFordEvaluation(1, A[ ][ ]) // Function to evaluate
{
  D[source] = 0; // D[1] = 0
  Rest all node = maxvalue → 999

D[6]

0	1	2	3	4	5
	0	999	999	999	999



If D[dn] > D[sn] + weight of edge uv
then D[dn] = D[sn] + weight of edge uv

This is executed when A[sn][dn] != 999 // Edge exist

I node 1
A[1][1] => 0 != 999 ✓
D[1] > D[1] + A[1][1]
0 > 0 + 0 F D[1] = 0

A[1][2] => 6 != 999 ✓
D[2] > D[1] + A[1][2]
999 > 0 + 6 T D[2] = 6

A[1][3] => 999 != 999 ✗ D[3] = 999

A[1][4] => 5 != 999 ✓
D[4] > D[1] + A[1][4]
999 > 0 + 5 T D[4] = 5

A[1][5] => 4 != 999 ✓
D[5] > D[1] + A[1][5]
999 > 4 T D[5] = 4

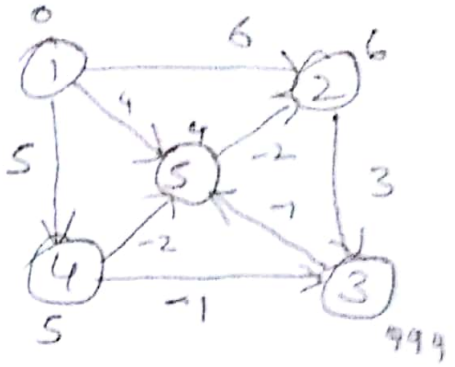
D[6]

0	1	2	3	4	5
	0	6	999	5	4

II A[2][1] = 9991 = 999

A[2][2]

Present Graph →



II A[2][1] = 9991 = 999 F

A[2][2] = 0 ! = 999 T

D[1] = 0

D[2] > D[2] + A[2][2]

6 > 6 + 0

D[2] = 6

A[2][3] = 3 ! = 999 T

D[3] > D[2] + A[2][3]

999 > 6 + 3 T

D[3] = 9

A[2][4] = 9991 = 999 F

A[2][5] = 9991 = 999 F

D[4] = 5

D[5] = 4

D[6)

0	1	2	3	4	5
	0	6	9	5	4

III A[3][1] 9991 = 999 F

D[1] = 0

A[3][2] 9991 = 999 F

D[2] = 6

A[3][3] 0 1 = 999 T

D[3] > D[3] + A[3][3]

9 > 9 + 0

D[3] = 9

A[3][4] 9991 = 999 F

D[4] = 5

A[3][5] -1 1 = 999 T

D[5] > D[3] + A[3][5]

4 > 9 - 1 T

D[5] = 4

IV A[4][1] 9991 = 999 F

D[1] = 0

A[4][2] 9991 = 999 F

D[2] = 6

A[4][3] -1 1 = 999 T

D[3] > D[4] + A[4][3]

9 > 5 - 1

D[3] = 4

A[4][4] 0 1 = 999 T

D[4] > D[4] + A[4][4]

D[4] = 5

DE A[4][5] -2 != 999 T

D[5] > D[4] + A[4][5]

4 > 5 - 2 D[5] = 2

D[6]

0	1	2	3	4	5
	0	6	4	5	3

V A[5][1] 999 != 999 F D[1] = 0

A[5][2] -2 != 999 T

D[2] > D[5] + A[5][2]

6 > 3 - 2 D[2] = 1

A[5][3] 999 != 999 D[3] = 4

A[5][4] 999 != 999 D[4] = 5

A[5][5] 0 != 999

D[5] > D[5] + A[5][5]

3 > 3 + 0 D[5] = 3

D[6]

1	2	3	4	5
0	-1	4	5	3

node = 2

I A[1][1] 0 != 999 ✓

D[1] > D[1] + A[1][1] D[1] = 0

A[1][2] 6 != 999

D[2] > D[1] + A[1][2] D[2] = 1

1 > 1 + 6

A[1][3] 999 != 999 × D[3] = 4

A[1][4] 5 != 999 ✓

D[4] > D[1] + A[1][4]

5 > 0 + 5 D[4] = 5

A[1][5] 4 != 999

D[5] > D[1] + A[1][5]

3 > 0 + 4 F D[5] = 3

D[6]

1	2	3	4	5
0	1	4	5	3

In similar ways in A[2][1.....5]

D[6] =

1	2	3	4	5
0	1	4	5	3

sn dn

For A[3][1.....5]
A[4][1.....5]
A[5][1.....5]

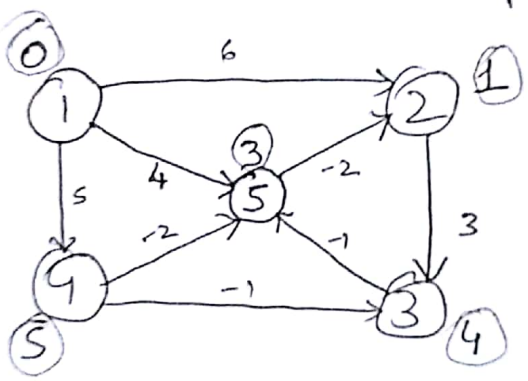
D[6] =

1	2	3	4	5
0	1	4	5	3

node = 3, 4

D[6] =

1	2	3	4	5
0	1	4	5	3



distance of source 1 to 1 is D[1] = 0
distance of source 1 to 2 is D[2] = 1
distance of source 1 to 3 is D[3] = 4
distance of source 1 to 4 is D[4] = 5
distance of source 1 to 5 is D[5] = 3