12/01/23

Bellmann Ford algm: Program for distance vecto algm to find suitable path for transmission

```
#include <stdio.h>
#include <stdlib.h>
int BellmanFord ( int G[20][20], int V, int E, int edge
[20][20]) {
int i, u, v, k, distance [20], parent [20], s, flag=1;
for (i=0; i< V; i++)
distance [i]= 1000,
Parent [i] = -1;
Printf ("Enter source :");
scanf ("%d", &s);
distance [s-1] =0;
for (i=0; i < V-1; i++) {
for ( k=0; k<E; k++){
u=edge [k][0], v = edge [k][1];
if (distance [u] + ~~dis~~ G[u][v] < distance [v])
distance [v] = distance [u]+ G[u][v];
parent [v] = u;
}}
for ( k=0; k<E; k++){
u= edge [k][0] , v= edge [k][1];
if (distance [u] +G[u][v] < distance [v])
flag =0;
}
if (flag)
for(i=0; i < V; i++)
Printf (" vertex %d → cost = %d parent = %d \n
i+1, distance [i], parent [i]+1);
return flag;
```

```
}
int main(){
 int V, edge[20][2], G[20][20], i, j, k=0;
 Pf("Enter no. of vertices:");
 Scanf("%d", &V);
 Pf("Enter graph in matrix form: \n");
 for(i=0; i<V; i++)
   for(j=0; j<V; j++){
    Scanf("%d", &G[i][j]);
    if(G[i][j] != 0)
    edge[k][0]=i, edge[k+1][1]=j;
 }
if(Bellman_ford(G, V, k, edge))
   Printf("\nNo negetive weight cycle \n");
else
   Printf("\n Negative weight cycle exists \n");
   return 0;
}
```

O/P:

```
Enter no. of vertices: 5
Enter graph in matrix :
```

Bellmann:

O/P: Enter no. of routers (<10): 6
Enter 99 if corresponding router is not adjacent to router A:
B C D E F
Enter matrix: 4 5 99 99 99
A C D E F
Enter 99 if corresponding router is not adjacent to B:
A B D E F
Enter matrix: 4 11 9 7 99
Enter 99 if corresponding router is not adjacent to C:
A B C E F
Enter matrix: 5 11 99 3 99
Enter 99 if corresponding router is not adjacent to D:
A B C D F
Enter matrix: 99 9 99 13 2
Enter 99 if corresponding router is not adjacent to E:
A B C D E
Enter matrix: 99 7 3 13 2
Enter 99 if corresponding router is not adjacent to F:
A B C D E
Enter matrix: 99 99 99 2 6

Routes Table entries for A:
Destination Routes: A B C D E F
Outgoing Line : A B C D E F
Hop Count : 0 4 5 99 99 99

Routes Table entries for B:
Destination Router: A B C D E F
Outgoing Line : A B C D E F
Hop Count : 4 0 11 9 7 99

Routes Table entries for C:
Destination Routes : A B C D E F
Outgoing Line : A B C D E F
Hop count : 5 11 0 99 3 99

Routes Table entries for D:
Destination Routes : A B C D E F
Outgoing Line : A B C D E F
Hop Count : 99 9 99 0 13 2

Routes Table entries for Router E:

Destination Routers:	A	B	C	D	E	F
Outgoing Line :	A	B	C	D	E	F
Hop Count :	99	7	3	13	0	2

Rout Table entries for routes F:

Destination Routes :	99	99	99	2	6	0