9. Bellman ford

```
#include <stdio.h>
#include <stdlib.h>

int Bellman_ford(int G[20][20], int V, int
                 edge[20][20])

{
int i, u, v, k, distance[20], parent[20],
    flag = 1;

for (i = 0; i < V; i++)
    distance[i] = 1000; parent[i] = -1;
    printf("Enter Source");
    scanf("%d", &s);
    distance("%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] + 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, dist[i], parent[i]+1);
return flag;
}
int main() {
  int V, edge[20][2], G[20][20], i, j, k=0;
printf("enter no of vertices \n");
  scanf("%d", &V);
printf("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] = j;
      }
    }
  if(BellmanFord(G, V, k, edge))
  printf("No negative weight cycle\n");
  else
  printf("Negative weight cycle exists\n");
  return 0;
}
```

Output:-

BELLMAN FORD.
Enter no. of Vertices = 5
Enter graph in matrix form
0 6 0 7 0
0 0 5 8 -4
0 -2 0 0 0
0 0 -3 0 9
2 0 7 0 0
Enter Source : 1
Vertex 1 → Cost = 0 parent = 0
Vertex 2 → Cost = 2 parent = 3
Vertex 3 → Cost = 4 parent = 4
Vertex 4 → Cost = 7 parent = 1
Vertex 5 → Cost = 2 parent = 2.