Program :

```
#include <stdio.h>
#include <stdlib.h>
int Bellman_Ford(int G[20][20], int V, int E, int edge[20][2]) {
   int i, u, v, k, distance[20], parent[20], S, flag;
   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] + G[u][v] < distance[v]) {
            distance[v] = distance[u] + G[u][v];
            parent[v] = u;
         } }
   }
   flag = 1;
   for (k = 0; k < E; k++) {
      u = edge[k][0], v = edge[k][1];
      if (distance[u] + G[u][v] < distance[v]) {
         flag = 0;
         break;
      }
   }
   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, E = 0, edge[20][2], G[20][20], i, j;
```

```
    printf("BELLMAN FORD\n");
    printf("Enter number of vertices: ");
    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[E][0] = i;
                edge[E][1] = j;
                E++;
            } }
    }
    if (Bellman_Ford(G, V, E, edge)) {
        printf("\nNo negative weight cycle\n");
    } else {
        printf("\nNegative weight cycle exists\n");
    }
    return 0;
}
```

Output :