Name: Ravi Sajjanar
USN : 1BM19CS127
Course : ADA Lab Test 2
Date : 08/07/2021.
Sign :

Program: 7

From a given vertex in a weighted connected graph, find shortest path to others vertices using Dijkstra's algorithm.

Modification:

print number of nodes along the shortest paths;

Program Code:

```
# include <stdio.h>
int minDistance (int dist[], int sptSet[], intV]
{ int min = 999, min_index;
  int v;
     for (v=0; v<V; v++)
        if (sptSet [v] == 0 && dist [v] <= min)
            min = dist [v], min_index = v;

   return min_index;
}.
```

```
int printSolution(int src, int dist[], int V) {
  int i;
  printf("\n\t Vertex \t\t Distance from source\n");
    for (i=0; i<V; i++)
      printf("\t %c --> %c \t\t %d \n", sr+65,
                          i+65, dist[i]);
    }

void dijkstra(int graph[10][10], int src, int V) {
  int dist[V];
  int i, count, u, v;
  int sptSet[V];
    for (i=0; i<V; i++)
       dist[i]= 999, sptSet[i]=0;

  dist[src]= 0;
  for (count=0; count < V-1; count++) {
     u = minDistance(dist, sptSet, V);
     spt Set [u]= 1;

  for ( v=0; v < V; v++) {
     if (!sptSet[v] && graph[u][v] &&
             dist[u]!= 999 && dist[u] +
       graph[u][v] < dist[v]) {
     dist [v]= dist[u]+ graph[u][v];
        }
  }
  printSolution(src, dist, V);
  }
}
```

```
int main() {
    int i, j, V;
     int graph[10][10];
     print (" Enter the number of vertices \n");
     scanf (" %d", &V);
     printf (" Enter the adjacency matrix \n"];

 for (i=0; i<V; i++) {
     for (j=0; j<V; j++) {
         scanf (" %d", &graph [i][j]);
     }

     disjkstra (graph, 0, V);

   return 0;

 }
```

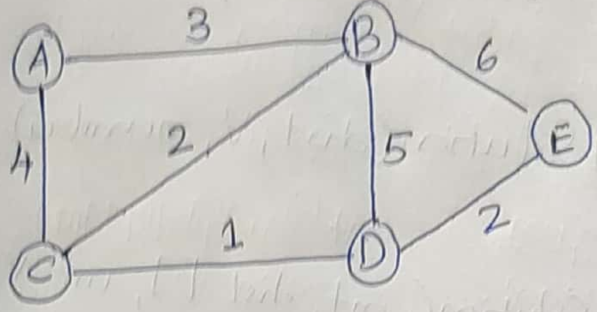
Modification:

to print number of nodes along the Shortest path:

```
void dijkstra (int graph[9][9], int src, int V] {
  int  i, count, u, v;
   int spt Set [V];
   for (i=0; i<V; i++)
      dist [i] = 999, spt Set[i]=0;

    dist [Src] = 0;
     int number [V];
     for ( Count=0; Count < V-1; Count ++) {
          u= minDistance ( dist, sptSet, V);
           number [u] = Count ;

       spt Set[u]= 1;
              for [v=0; v < V ; v++) {

if (!SptSet [v] && graph [u] [v] && dist [u]! =999 &&
        dist [u] + graph [u] [v] < dist [V]) {

  dist [v] = dist [u] + graph [u] [v];
  }}.
for (i=0; i<V ; i++) {
     if (spt Set [i] ==0 ) {
          int   max=0;
               for (int j=0 ; j<V; j++) {
 if ( number [j]> maxx && spt Set [j]!=0
          max = number [j] ;
   }
```

```
    number [i] = max;
    }
  }
  printSolution(dist, V, number);

int printSolution( int dist [], int V, int number[]) {
        int i;
  printf("vertex \t\t Distance from Source\n");
    for (i=0; i<V; i++)
       printf ("%d %d %d", i, dist (i), number[i] );
    }
}.
```

Graph:



⇓

Adjacency Matrix

$$\begin{array}{c|ccccc} & A & B & C & D & E \\ \hline A & \infty & 3 & 4 & \infty & \infty \\ B & 3 & \infty & 2 & 5 & 6 \\ C & 4 & 2 & \infty & 1 & \infty \\ D & \infty & 5 & 1 & \infty & 2 \\ E & \infty & 6 & \infty & 2 & \infty \end{array}$$