dij

```
#include<stdio.h>
#include<stdlib.h>
#define INFINITY 9999;
#define MAX 5
void dijkstra(int Cost[MAX][MAX], int n, int startnode)
{
    int distance[MAX], Path[MAX];
    int visited[MAX], Count, minidistance, nextnode, i, j;
    for(i=0;i<n;i++)
    {
        distance[i] = Cost[startnode][i];
        Path[i] = startnode;
        visited[i] = 0;
    }
    distance[startnode] = 0;
    visited[startnode] = 1;
    Count = 1;
    while(Count < n-1)
    {
        minidistance = INFINITY;
        for(i=0; i<n; i++)
        {
            if(distance[i] < minidistance && !visited[i])
            {
                minidistance = distance[i];
                nextnode = i;
            }
        }
        visited[nextnode] = 1;
```

```
for (i=0; i<n; i++)
  {
    if (!visited[i])
      {
        if (minidistance + Cost[nextnode][i]
                                  <distance[i])
          {
            distance[i] = minidistance +
                                  Cost[nextnode][i];
            Path[i] = nextnode;
          }
      }
    Count++;
  }
}
for (i=0; i<n; i++)
  {
    if (i != startnode)
      {
        printf ("\n Distance of %d = %d", i,
                                  distance[i]);
        Printf ("\n Path = %d", i);
        j = i;
        do
          {
            j = path[j];
            Printf ("<-%d", j);
          }
        while (j != startnode);
      }
  }
}
```

```
void main()
{ int g[MAX][MAX], i, j, n, u;
  Printf ("\n Enter the no of vertices::");
  Scanf ("%d", &n);
  Printf ("\n Enter the adjancy matrix::\n")
  for (i=0; i<n; i++)
  {
   for (j=0; j<n; j++)
   {
     Scanf ("%d", &g[i][j]);
      if (g[i][j]==0)
      {
        g[i][i] = INFINITY;
   }   }
   Printf ("\n");
  }
  Printf ("\n Enter the starting node::");
  Scanf ("%d", &u);
  dijikstra (g, n, u);
}
```