Dijkstra's shortest path algorithm. Shaeeau Reug

```
import sys
class Graph()
def-init(self, vertices):
self. V = vertices.
self. graph=[[0 for column in range [vertices)]
for row in range (vertices)]
def printSolution (self, dist):
print "Vertex \t Distance from Source")
for node in range(self. V):
 print node, "\t", dist[node]
for v in range (self.v):
if dist[v]<min and sptSet[v]== False:
 min = dist[v]
min_index = v
 return min_index
dist = [sys.maxint]* self.v
dist[src] =0
sptSet = [False]* self.v
for count in range (self.v):
 u = self.minDistance(dist, sptSet)
sptSet [u]= true
for v in range (self.v):
if self. graph [u][v] > 0 and sptSet[v]==
      False and \ dist[v] >dist[u]+self.graph[u][v]
dist [v] = dist[u]+self.graph[u][v]
self.printSolution (dist)
g = Graph(9).
g.graph = [[0,4,0,0,0,0,0,8,0],
           [4,0,8,0,0,0,0,11,0],
            [0,8,0,7,0,4,0,0,2],
[0, 0, 7, 0, 9, 14, 0, 0, 0],
[0, 0, 4, 14, 10, 9 2, 0, 0],
[8, 11, 0, 0, 0, 1, 0, 7].
[0, 0, 2, 0, 0, 0, 67, 7, 0].
];
g.dijkstra(0);
```

——X——