***Dijsktra Algorithm***

public class Main

{

static final int V = 9;

int minDistance(int[] dist, boolean[] sptSet){

int min = Integer.MAX\_VALUE, min\_index = -1;

for(int v=0;v<V;v++){

if(sptSet[v] == false && dist[v]<=min){

min = dist[v];

min\_index = v;

}

}

return min\_index;

}

void printSolution(int[] dist){

System.out.println("vertex => distance from source");

for(int i=0;i<V;i++){

System.out.println(i+" => "+dist[i]);

}

}

void dijkstra(int[][] graph, int src){

int[] dist = new int[V];

boolean[] sptSet = new boolean[V];

for(int i=0;i<V;i++){

dist[i] = Integer.MAX\_VALUE;

sptSet[i] = false;

}

dist[src] = 0;

for(int count=0;count<V-1;count++){

int u = minDistance(dist, sptSet);

sptSet[u] = true;

for(int v=0;v<V;v++){

if(!sptSet[v] && graph[u][v] != 0 && dist[u] != Integer.MAX\_VALUE && dist[u] + graph[u][v] < dist[v] ){

dist[v] = dist[u] + graph[u][v];

}

}

}

printSolution(dist);

}

public static void main(String[] args) {

int graph[][] = new int[][] { { 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, 0, 9, 0, 10, 0, 0, 0 },

{ 0, 0, 4, 14, 10, 0, 2, 0, 0 },

{ 0, 0, 0, 0, 0, 2, 0, 1, 6 },

{ 8, 11, 0, 0, 0, 0, 1, 0, 7 },

{ 0, 0, 2, 0, 0, 0, 6, 7, 0 } };

Main t = new Main();

t.dijkstra(graph, 0);

}

}