public class Dijkstra {

public static final int M = 10000;

public static void main(String[] args) {

int[][] weight1 = {

{0,4,M,2,M},

{4,0,4,1,M},

{M,4,0,1,3},

{2,1,1,0,7},

{M,M,3,7,0}

};

int start = 0;

int[] shortPath = dijkstra(weight1, start);

for (int i = 0; i < shortPath.length; i++)

System.out.println("从" + start + "出发到" + i + "的最短距离为：" + shortPath[i]);

}

public static int[] dijkstra(int[][] weight, int start) {

int n = weight.length;

int[] shortPath = new int[n];

String[] path = new String[n];

for (int i = 0; i < n; i++)

path[i] = new String(start + "-->" + i);

int[] visited = new int[n];

shortPath[start] = 0;

visited[start] = 1;

for (int count = 1; count < n; count++) {

int k = -1;

int dmin = Integer.MAX\_VALUE;

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

if (visited[i] == 0 && weight[start][i] < dmin) {

dmin = weight[start][i];

k = i;

}

}

shortPath[k] = dmin;

visited[k] = 1;

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

if (visited[i] == 0 && weight[start][k] + weight[k][i] < weight[start][i]) {

weight[start][i] = weight[start][k] + weight[k][i];

path[i] = path[k] + "-->" + i;

}

}

}

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

System.out.println("从" + start + "出发到" + i + "的最短路径为：" + path[i]);

}

System.out.println("=====================================");

return shortPath;

}

}

import java.util.\*;

public class Floyd {

public static int[][] path;

public static int[][] floyd(int[][] C,int n)

{

path=new int[n][n];

for(int i=0;i<n;i++)

{

for(int j=0;j<n;j++)

{

path[i][j]=-1;

}

}

for(int k=0;k<n;k++)

{

for(int i=0;i<n;i++)

{

for(int j=0;j<n;j++)

{

if(C[i][k]!=Integer.MAX\_VALUE&&C[k][j]!=Integer.MAX\_VALUE&&C[i][k]+C[k][j]<C[i][j])

{

C[i][j]=C[i][k]+C[k][j];

path[i][j]=k;

}

}

}

}

return C;

}

public static void main(String[] args) {

int a=Integer.MAX\_VALUE;

int[][] C= {{0,a,10,a,30,100},{a,0,5,a,a,a},{a,a,0,50,a,a},{a,a,a,0,a,10},

{a,a,a,20,0,60},{a,a,a,a,a,0}};

int n=C.length;

int[][] route=floyd(C,n);

for(int i=0;i<n;i++)

{

for(int j=0;j<n;j++)

{

if(i==j) continue;

else if(route[i][j]==a)

;

else

System.out.println(i+"到"+j+"之间的最短路径长度为："+route[i][j]);

}

}

}

}