Laboratory work #8

Student: CAO Xinyang  
Student ID: 20321308  
Timus Name: hduads2022\_20321308

Mail: c.x\_yang@foxmail.com

Problem #1160

Screenshot from Timus:



Explanation of algorithm:

This problem needs to figure out the minimum spanning tree. This code uses Kruskal algorithm: first sort all edges in order of weight, then traverse each edge(u,v) in turn, if the minimum spanning tree already contains u and v, skip; If the minimum spanning tree does not contain u or v, add the edge. If the weight of (u,v) is the smallest of all edges, then the minimum spanning tree must contain that edge.

Computational complexity of algorithm:

F(N,M) = M + M \* LogM + N + (M + N)

T(N,M) = O(M \* LogM)

Source code:

import java.util.Arrays;

import java.util.Scanner;

class Edge implements Comparable<Edge> {

int u;

int v;

int length;

Edge(int u, int v, int length) {

this.u = u;

this.v = v;

this.length = length;

}

public int compareTo(Edge e) {

if (length > e.length)

return 1;

else if (length < e.length)

return -1;

return 0;

}

}

public class App {

public static int search(int[] c, int x) {

if (c[x] == x)

return x;

else

return c[x] = search(c, c[x]);

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int n = scanner.nextInt();

int m = scanner.nextInt();

Edge[] edges = new Edge[m];

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

int u = scanner.nextInt();

int v = scanner.nextInt();

int length = scanner.nextInt();

edges[i] = new Edge(u - 1, v - 1, length);

}

Arrays.sort(edges);

int[] c = new int[n];

int[] u = new int[m];

int[] v = new int[m];

int count = 0;

int maxLength = 0;

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

c[i] = i;

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

int x = search(c, edges[i].u);

int y = search(c, edges[i].v);

if (x == y) {

continue;

} else {

c[x] = y;

u[count] = edges[i].u + 1;

v[count] = edges[i].v + 1;

if (edges[i].length > maxLength)

maxLength = edges[i].length;

count++;

}

}

System.out.println(maxLength);

System.out.println(count);

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

System.out.println(u[i] + " " + v[i]);

}

scanner.close();

}

}