Отчет

по лабораторной работе «1160. Network»

по дисциплине «Алгоритмы и структуры данных»

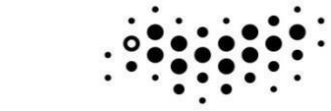
Авторы:

Полит Алексей Денисович

Факультет: СУиР

Группа: R3235

Преподаватель: Тропченко Андрей Александрович



Санкт-Петербург, 2021г

1. Задача

   Andrew is working as system administrator and is planning to establish a new network in his company. There will be N hubs in the company, they can be connected to each other using cables. Since each worker of the company must have access to the whole network, each hub must be accessible by cables from any other hub (with possibly some intermediate hubs).

   Since cables of different types are available and shorter ones are cheaper, it is necessary to make such a plan of hub connection, that the maximum length of a single cable is minimal. There is another problem - not each hub can be connected to any other one because of compatibility problems and building geometry limitations. Of course, Andrew will provide you all necessary information about possible hub connections.

   You are to help Andrew to find the way to connect hubs so that all above conditions are satisfied.

2. Исходные данные

   The first line contains two integer: N - the number of hubs in the network (2 ≤ N ≤ 1000) and M — the number of possible hub connections (1 ≤ M ≤ 15000). All hubs are numbered from 1 to N. The following M lines contain information about possible connections - the numbers of two hubs, which can be connected and the cable length required to connect them. Length is a positive integer number that does not exceed 106. There will be no more than one way to connect two hubs. A hub cannot be connected to itself. There will always be at least one way to connect all hubs.

3. Код программы

```
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

struct cable {
   unsigned short a, b;
   unsigned l;
};

bool compare(cable a, cable b) {
   return a.l < b.l;
}

int main() {

   unsigned short n, m;
```

```
    cin >> n >> m;
    vector<cable> v;

    unsigned short a, b;
    unsigned l;
    for (int i = 0; i < m; ++i) {
        cin >> a >> b >> l;
        v.push_back((cable){a-1, b-1, l});
    }

    sort(v.begin(), v.end(), compare);

    short used_points[n];
    short used_vectors[m];

    fill_n(used_points, n, -1);
    fill_n(used_vectors, m, 0);

    unsigned group_number = 0;
    unsigned max = 0;
    unsigned counter = 0;
    for (unsigned short i = 0; i < m; i++){
            if (used_points[v[i].a] != -1 && used_points[v[i].a] == used_points[v[i].b])
continue;
            else if(used_points[v[i].a] == -1 && used_points[v[i].b] == -1 ){
                ++group_number;
                used_points[v[i].a] = group_number;
                used_points[v[i].b] = group_number;
            }else if ((used_points[v[i].a] != -1 && used_points[v[i].b] == -1
)||(used_points[v[i].a] == -1 && used_points[v[i].b] != -1 )){
                int groupnubmer = used_points[v[i].a]+used_points[v[i].b]+1;
                used_points[v[i].a] = groupnubmer;
                used_points[v[i].b] = groupnubmer;
            }else{
                int mingroupnubmer;
                int maxgroupnubmer;
                if(used_points[v[i].b]>used_points[v[i].a]) {
                    mingroupnubmer = used_points[v[i].a];
                    maxgroupnubmer = used_points[v[i].b];
                    } else {
                    mingroupnubmer = used_points[v[i].b];
                    maxgroupnubmer = used_points[v[i].a];
                }
                for (unsigned short j = 0; j < n; j++)
                    if(used_points[j] == maxgroupnubmer) used_points[j] =
mingroupnubmer;
            }
            used_vectors[i] = 1;
```

```
            max = v[i].l;
            ++counter;
        }

        cout << max << endl << counter << endl;

        for (unsigned short j = 0; j < m; j++)
            if (used_vectors[j])
                cout << v[j].a+1<< " " << v[j].b+1 << endl;
        return 0;
    }
```

9350510	15:08:28 3 май 2021	Aleksey	1160. Network	G++ 9.2 x64	Accepted		0.062	700 КБ

4. Описание алгоритма

В данной задаче используется алгоритм Краска на графах. Хабы - вершины графа, провода - рёбра. Все рёбра записываются в вектор и сортируются. Потом мы добавляем рёбра в граф с присвоением номера каждой вершине, который соответствует номеру подграфа, не соединенного с другими подграфами. Если 2 подграфа объединяются им присваивается наименьший из их номеров. Продолжаем пока все вершины не будут соединены.