УНИВЕРСИТЕТ ИТМО

Национальный исследовательский университет ИТМО
(Университет ИТМО)

Факультет систем управления и робототехники

Дисциплина: Алгоритмы и структуры данных
Отчет по практической работе (1160 задача).

Студент:
Евстигнеев Дмитрий
Группа: *R3242*
Преподаватель:
Тропченко Андрей Александрович

Санкт-Петербург
2021

Цель: написать программу для решения задачи №1160 на сайте Timus Online

Задача:

1160. Network

Ограничение времени: 1.0 секунды

Ограничение памяти: 64 МБ

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.

Исходные данные

The first line contains two integer: N - the number of hubs in the network ($2 \le N \le 1000$) and M — the number of possible hub connections ($1 \le M \le 15000$). 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 10^6. 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.

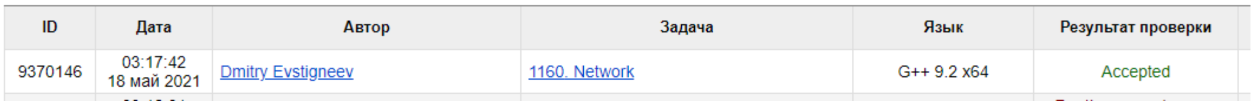
Результат

Output first the maximum length of a single cable in your hub connection plan (the value you should minimize). Then output your plan: first output P - the number of cables used, then output P pairs of integer numbers - numbers of hubs connected by the corresponding cable. Separate numbers by spaces and/or line breaks.

Пример

исходные данные	результат
4 6	1
1 2 1	4
1 3 1	1 2
1 4 2	1 3
2 3 1	2 3
3 4 1	3 4
2 4 1	

Принято системой (JUDGE_ID: 231802FR):



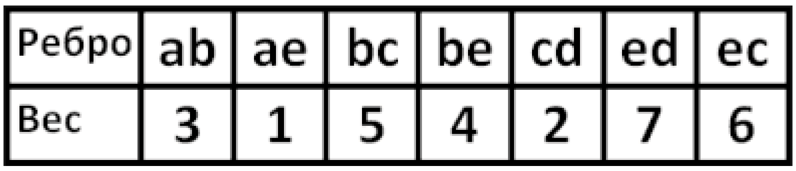
ID	Дата	Автор	Задача	Язык	Результат проверки
9370146	03:17:42 18 май 2021	Dmitry Evstigneev	1160. Network	G++ 9.2 x64	Accepted

Решение на языке C++:

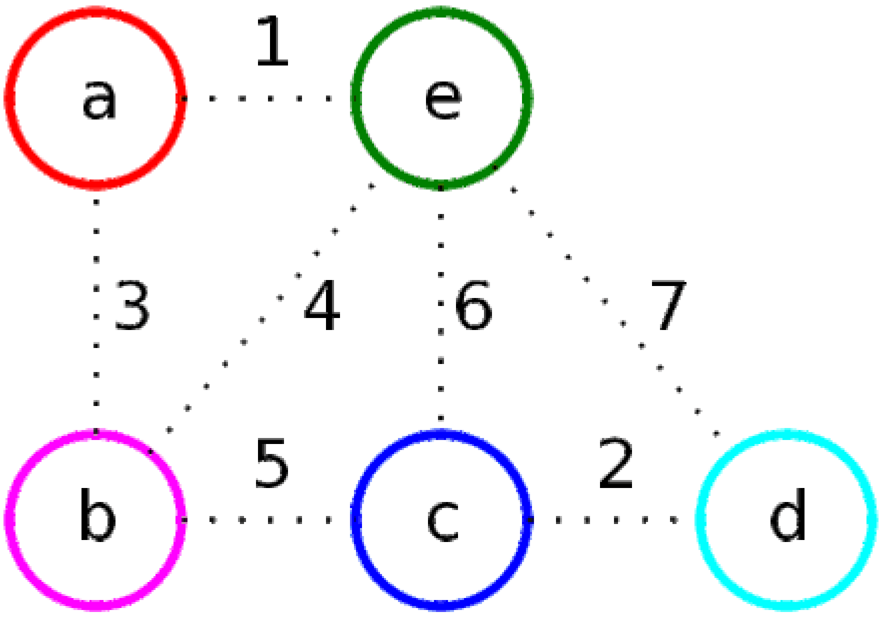
```
#include <stdio.h>
#include <vector>
#include <algorithm>

struct node
{
	node* parent;
	int rank; // For the union-by-rank heuristic
	int n;  // Number of this node
	node() : parent(this), rank(0) { }
};

struct edge
{
	node* a, *b;
	int weight;
	edge(node* a, node* b, int weight) : a(a), b(b), weight(weight) {}
};

void link(node* a, node* b) // Disjoint set union
{
	if (a->rank > b->rank)
		b->parent = a;
	else
	{
		a->parent = b;
		if (a->rank == b->rank)
			b->rank++;
	}
}

node* find(node* s) // Disjoint set find
{
	if (s != s->parent)
		s->parent = find(s->parent);
	return s->parent;
}

void join(node* a, node* b)
{
	link(find(a), find(b));
}

std::vector<node> nodes;
std::vector<edge> edges;
std::vector<edge*> ans;

int main()
{
	int N, M;
	scanf("%d %d", &N, &M);
	nodes.reserve(N + 1);
	for (int i = 0; i < N + 1; i++)
	{
		nodes.push_back(node());
		nodes.back().parent = &nodes.back();
		nodes.back().n = i;
	}
	edges.reserve(M);
	ans.reserve(M);
	for (int i = 0; i < M; i++)
	{
        int x, y, w;
        scanf("%d %d %d", &x, &y, &w);
        edges.emplace_back(&nodes[x], &nodes[y], w);
    }
    std::sort(edges.begin(), edges.end(), [](edge& a, edge& b) { return a.weight < 
b.weight; });
    int max = 0;
    for (edge& e : edges)
    {
        if (find(e.a) != find(e.b))
        {
            join(e.a, e.b);
            max = std::max(max, e.weight);
            ans.push_back(&e);
        }
    }
    printf("%d\n", max);
    printf("%d\n", ans.size());
    for (auto e : ans)
        printf("%d %d\n", e->a->n, e->b->n);
    return 0;

}
```

Суть алгоритма:

Алгоритм Краскала — эффективный алгоритм построения минимального остовного дерева взвешенного связного неориентированного графа.



Ребро	ab	ae	bc	be	cd	ed	ec
Вес	3	1	5	4	2	7	6


a
e
b
c
d
1
3
4
6
7
5
2