OJ 10600

ACM contest and Blackout

Prof. Edson Alves

Faculdade UnB Gama

In order to prepare the "The First National ACM School Contest" (in 20??) the major of the city decided to provide all the schools with a reliable source of power (The major is really afraid of blackouts). So, in order to do that, power station "Future" and one school (doesn't matter which one) must be connected; in addition, some schools must be connected as well.

You may assume that a school has a reliable source of power if it's connected directly to "Future", or to any other school that has a reliable source of power. You are given the cost of connection between some schools. The major has decided to pick out two the cheapest connection plans – the cost of the connection is equal to the sum of the connections between the schools. Your task is to help the major – find the cost of the two cheapest connection plans.

Durante os preparativos da “Primeira Maratona Nacional Escolar ACM”(em 20??), o prefeito da cidade decidiu prover todas as escolas com uma fonte de energia confiável (na verdade o prefeito está preocupado com blecautes). Assim, para atingir este objetivo, a estação de energia “Futuro” e uma escola (não importa qual) devem estar conectadas; além disso, algumas outras escolas devem estar conectadas também.

Você pode assumir que uma escola tem uma fonte de energia confiável se ela está conectada diretamente a “Futuro”, ou a qualquer escola que tenha uma fonte de energia confiável. Serão dados os custos de conexão entre algumas escolas. O prefeito tem que decidir entre os dois planos de conexão mais baratos – o custo de conexão é igual a soma das conexões entre todas as escolas. Sua tarefa é ajudar o prefeito – determine os custos dos dois planos mais baratos.

Input

The Input starts with the number of test cases, T $(1 < T < 15)$ on a line. Then T test cases follow. The first line of every test case contains two numbers, which are separated by a space, N $(3 < N < 100)$ the number of schools in the city, and M the number of possible connections among them. Next M lines contain three numbers A_i, B_i, C_i, where C_i is the cost of the connection $(1 < C_i < 300)$ between schools A_i and B_i. The schools are numbered with integers in the range 1 to N.

Output

For every test case print only one line of output. This line should contain two numbers separated by a single space – the cost of two the cheapest connection plans. Let S_1 be the cheapest cost and S_2 the next cheapest cost. It's important, that $S_1 = S_2$ if and only if there are two cheapest plans, otherwise $S_1 < S_2$. You can assume that it is always possible to find the costs S_1 and S_2.

Entrada

A entrada começa com o número de casos de teste T $(1 < T < 15)$ em uma linha. Então seguem T casos de teste. A primeira linha de cada caso de teste contém dois inteiros, separados por um espaço em branco, N $(3 < N < 100)$, o número de escolas na cidade, e M, o número de conexões possíveis entre elas. As próximas M linhas contém três números A_i, B_i, C_i, onde C_i é o custo da conexão $(1 < C_i < 300)$ entre as escolas A_i e B_i. As escolas estão numeradas com inteiros de 1 a N.

Saída

Para cada caso de teste imprima uma única linha. Esta linha deverá conter dois inteiros separados por um único espaço – o custo dos dois planos de conexão mais baratos. Seja S_1 o custo do plano mais barato e S_2 o custo do segundo plano mais barato. Imporante: $S_1 = S_2$ se e somente se há dois planos mais baratos, caso contrário $S_1 < S_2$. Você pode assumir que é sempre possível encontrar os custos S_1 e S_2.

Exemplo de entrada e saída

Exemplo de entrada e saída

```
5 8
```

Exemplo de entrada e saída

```
5 8
```

de escolas

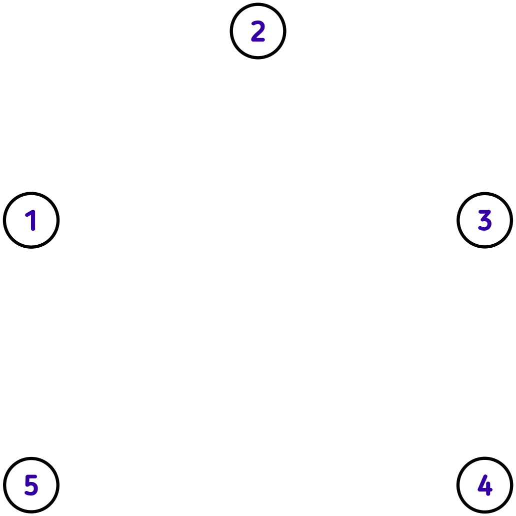
Exemplo de entrada e saída

5 8

de conexões

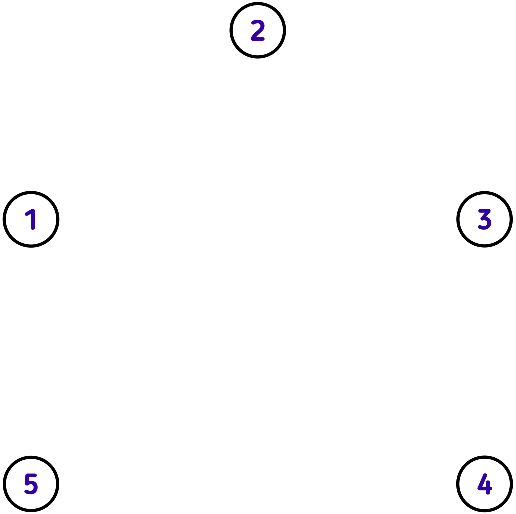
Exemplo de entrada e saída

5 8



Exemplo de entrada e saída

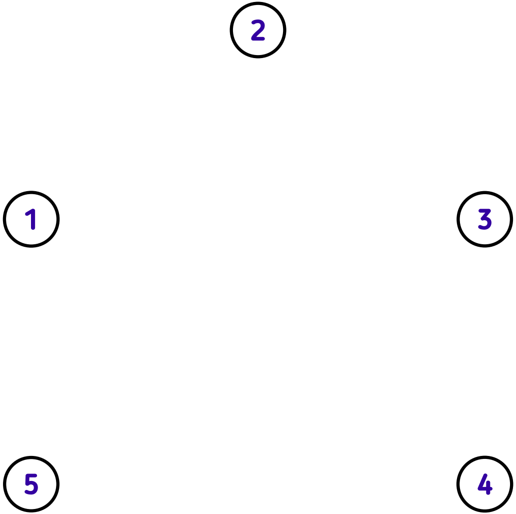
```
5 8
1 3 75
```



Exemplo de entrada e saída

```
5 8
1 3 75
```

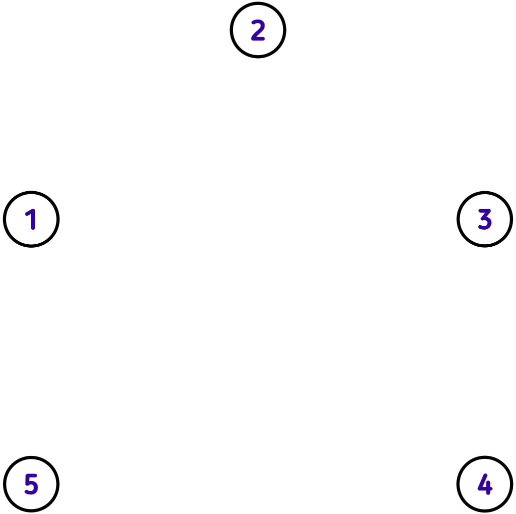
escola A



Exemplo de entrada e saída

```
5 8
1 3 75
```

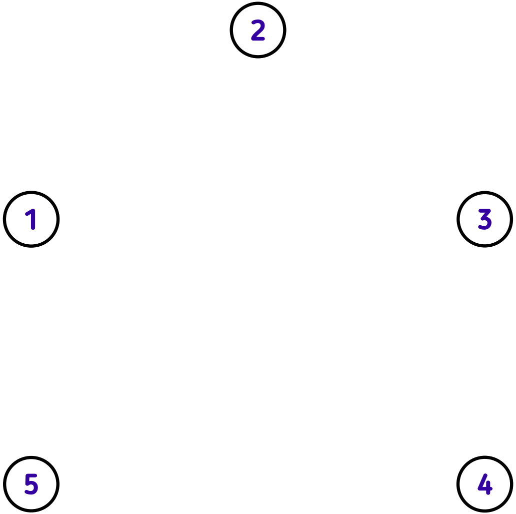
escola B



Exemplo de entrada e saída

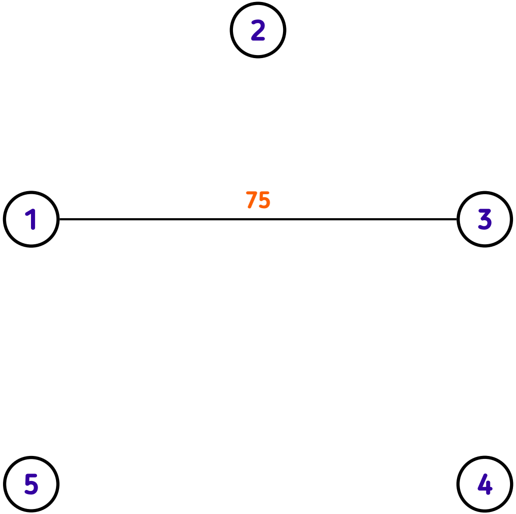
```
5 8
1 3 75
```

custo



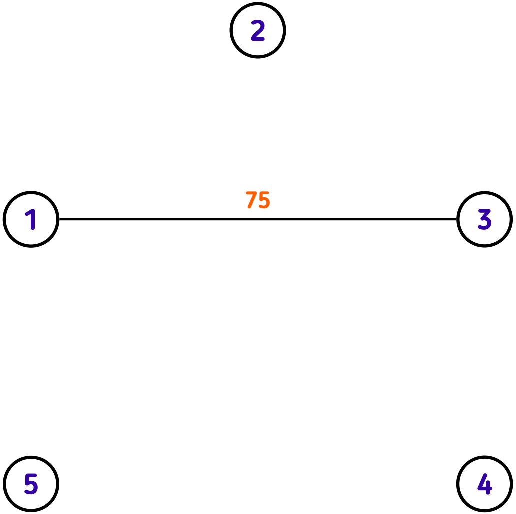
Exemplo de entrada e saída

```
5 8
1 3 75
```



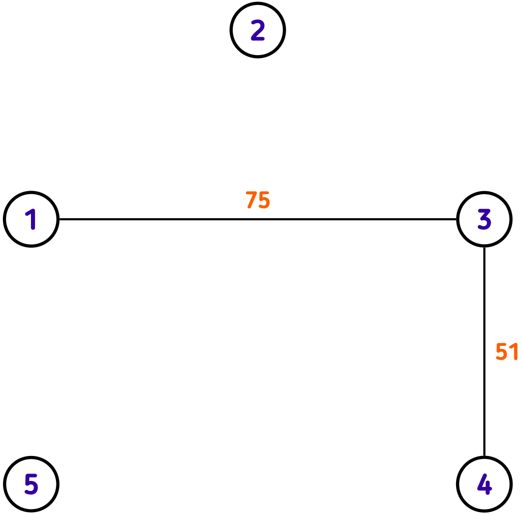
Exemplo de entrada e saída

```
5 8
1 3 75
3 4 51
```



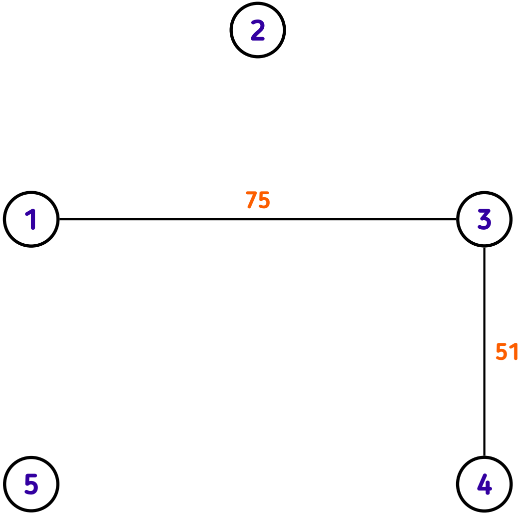
Exemplo de entrada e saída

```
5 8
1 3 75
3 4 51
```



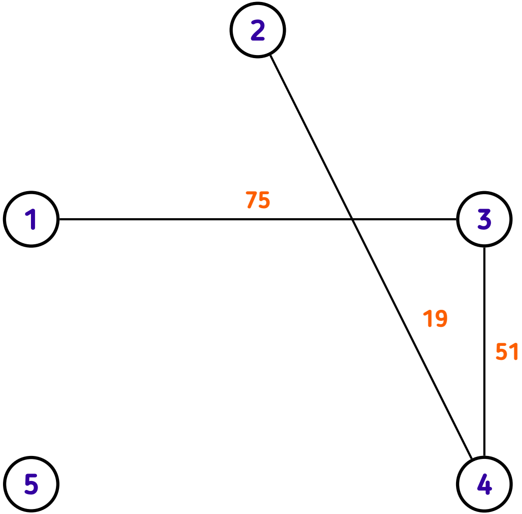
Exemplo de entrada e saída

```
5 8
1 3 75
3 4 51
2 4 19
```



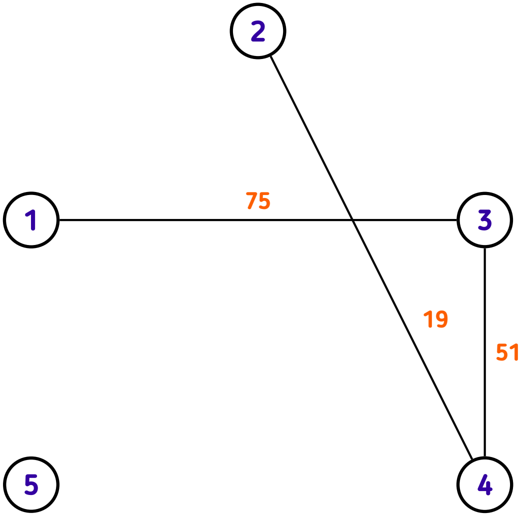
Exemplo de entrada e saída

```
5 8
1 3 75
3 4 51
2 4 19
```



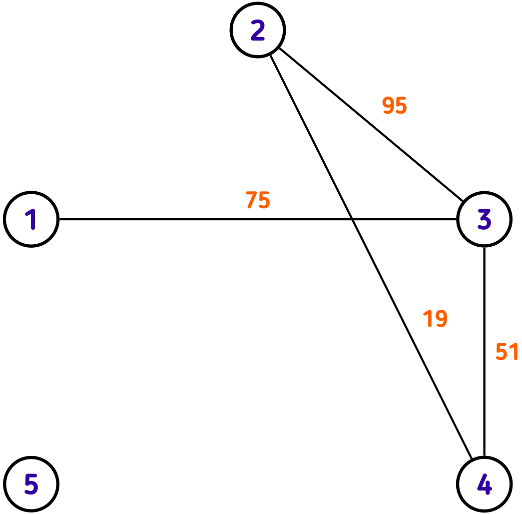
Exemplo de entrada e saída

```
5 8
1 3 75
3 4 51
2 4 19
3 2 95
```



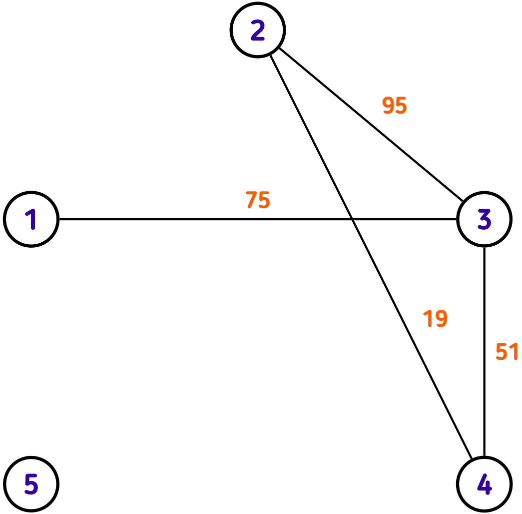
Exemplo de entrada e saída

```
5 8
1 3 75
3 4 51
2 4 19
3 2 95
```



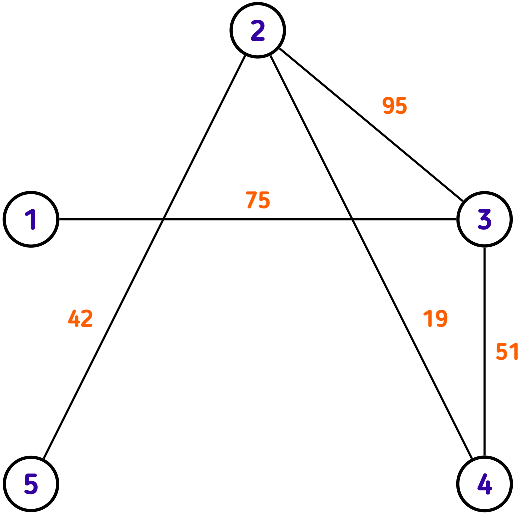
Exemplo de entrada e saída

```
5 8
1 3 75
3 4 51
2 4 19
3 2 95
2 5 42
```



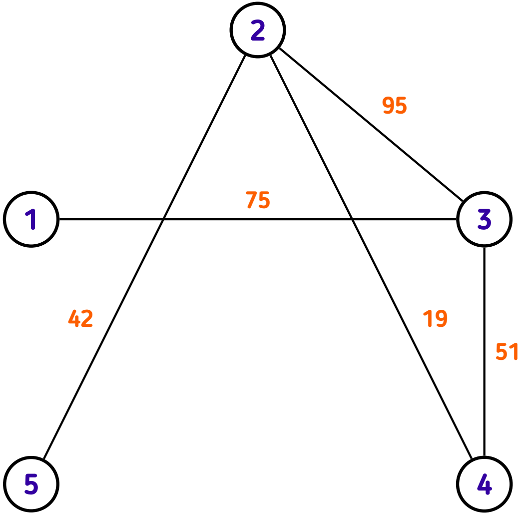
Exemplo de entrada e saída

```
5 8
1 3 75
3 4 51
2 4 19
3 2 95
2 5 42
```



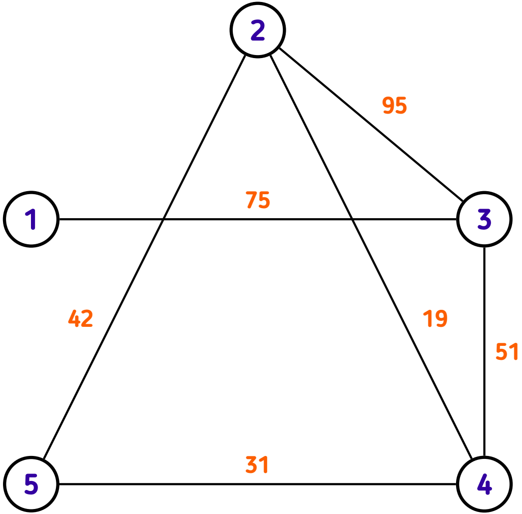
Exemplo de entrada e saída

```
5 8
1 3 75
3 4 51
2 4 19
3 2 95
2 5 42
5 4 31
```



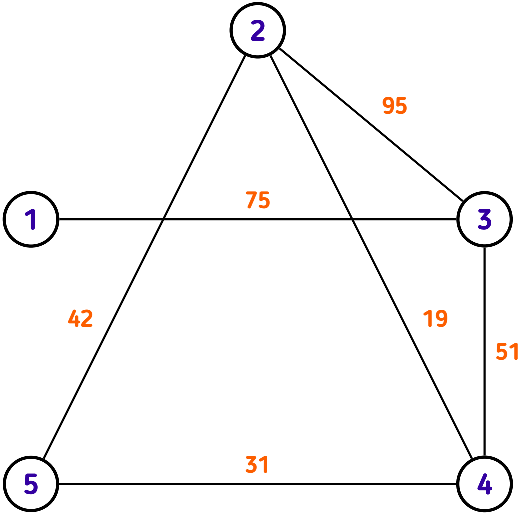
Exemplo de entrada e saída

```
5 8
1 3 75
3 4 51
2 4 19
3 2 95
2 5 42
5 4 31
```



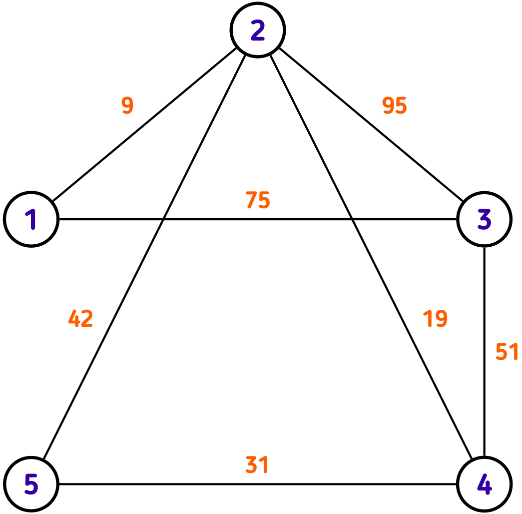
Exemplo de entrada e saída

```
5 8
1 3 75
3 4 51
2 4 19
3 2 95
2 5 42
5 4 31
1 2 9
```



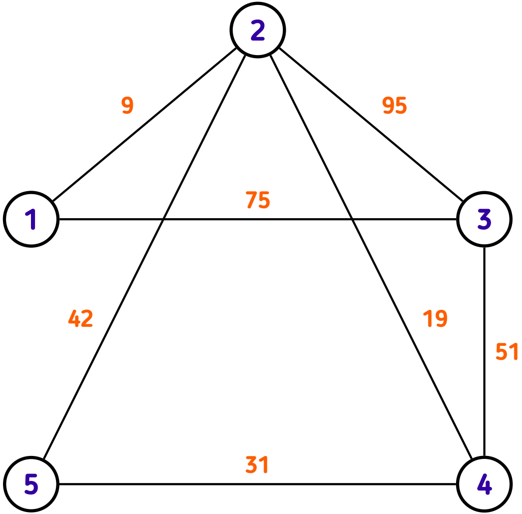
Exemplo de entrada e saída

```
5 8
1 3 75
3 4 51
2 4 19
3 2 95
2 5 42
5 4 31
1 2 9
```



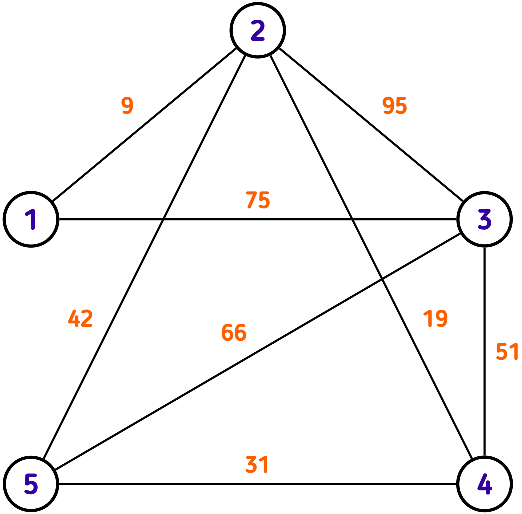
Exemplo de entrada e saída

```
5 8
1 3 75
3 4 51
2 4 19
3 2 95
2 5 42
5 4 31
1 2 9
3 5 66
```



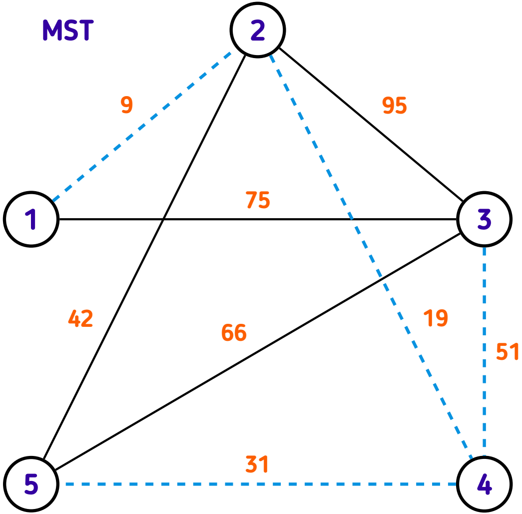
Exemplo de entrada e saída

```
5 8
1 3 75
3 4 51
2 4 19
3 2 95
2 5 42
5 4 31
1 2 9
3 5 66
```



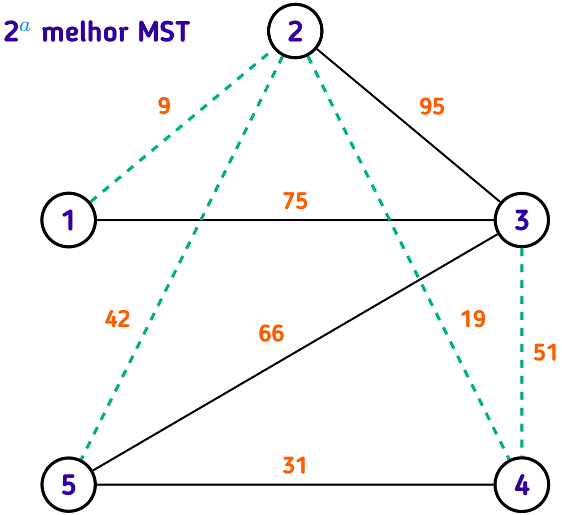
Exemplo de entrada e saída

```
5 8
1 3 75
3 4 51
2 4 19
3 2 95
2 5 42
5 4 31
1 2 9
3 5 66
```



Exemplo de entrada e saída

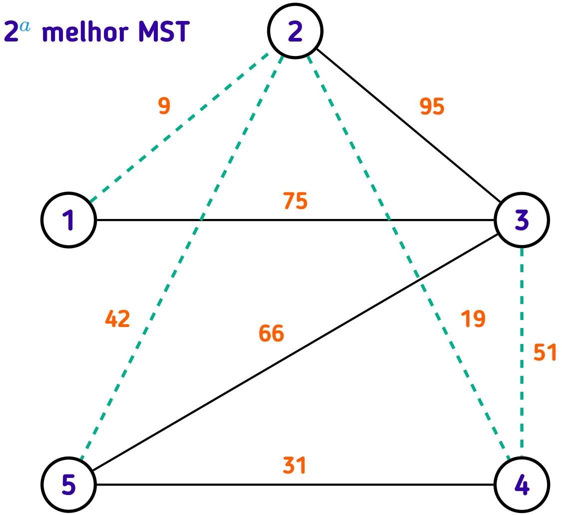
```
5 8
1 3 75
3 4 51
2 4 19
3 2 95
2 5 42
5 4 31
1 2 9
3 5 66
```



Exemplo de entrada e saída

```
5 8
1 3 75
3 4 51
2 4 19
3 2 95
2 5 42
5 4 31
1 2 9
3 5 66
```

```
110 121
```



Solução

Solução

- O problema consiste em determinar a segunda melhor MST

Solução

- O problema consiste em determinar a segunda melhor MST
- O texto do problema garante a existência desta segunda melhor MST

Solução

- O problema consiste em determinar a segunda melhor MST
- O texto do problema garante a existência desta segunda melhor MST
- É preciso modificar o algoritmo de Kruskall para que ele ignore a aresta indicada e retorne as arestas que formam a MST

Solução

- ⋆ O problema consiste em determinar a segunda melhor MST
- ⋆ O texto do problema garante a existência desta segunda melhor MST
- ⋆ É preciso modificar o algoritmo de Kruskall para que ele ignore a aresta indicada e retorne as arestas que formam a MST
- ⋆ **Cuidado:** Se a aresta removida for uma ponte, o grafo não terá uma MST!

```cpp
pair<int, int> solve(int N, vector<edge>& es)
{
    sort(es.begin(), es.end());

    auto [best, mst] = kruskal(N, es);
    int _2nd_best = oo;

    for (auto blocked : mst)
    {
        auto [cost, __] = kruskal(N, es, blocked);
        _2nd_best = min(_2nd_best, cost);
    }

    return { best, _2nd_best };
}
```

```
pair<int, vector<int>>
kruskal(int N, vector<edge>& es, int blocked = -1)
{
    vector<int> mst;
    UnionFind ufds(N);
    int cost = 0;

    for (int i = 0; i < (int) es.size(); ++i)
    {
        auto [w, u, v] = es[i];

        if (i != blocked and not ufds.same_set(u, v)) {
            cost += w;
            ufds.union_set(u, v);
            mst.emplace_back(i);
        }
    }

    return { (int) mst.size() == N - 1 ? cost : oo, mst };
}
```