South African Programming Olympiad
Day 2

Overview

Problem	guards	feast	rope
Source	guards.java guards.py guards.c guards.cpp guards.pas	N/A	rope.java rope.py rope.c rope.cpp rope.pas
Input file	stdin	feast.in	stdin
Output file	stdout	feast.out	stdout
Time limit	1 second. Python: 10 seconds.	N/A	2 seconds. Python: 20 seconds.
Memory limit	256MiB	N/A	256MiB
Detailed feedback	No	No	No
Total points	**100**	**100**	**100**

The maximum total score is 300 points.

Firing Guards

Introduction

The local lord has noticed that his guards are being inefficient. Instead of patrolling all walls of his castle equally, some walls are over guarded, and some are not even guarded at all. He has thus decided to fire some guards, as an example to the rest.

Help his steward see how bad the problem is in order to have evidence with which to placate his lordship!

Task

The castle is made up of N towers (numbered $1, 2, \ldots, N$). The towers are connected by M different length walls, upon which the guards walk.

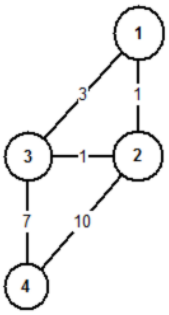
There are K guards. Each guard patrols between two towers, a_i and b_i. Guards are lazy, and will thus take the path between the towers that is the shortest. A wall is "guarded well" if exactly one guard walks over it. A wall is "under guarded" if no guards walk over it, and "over guarded" if more than one guard walks over it.

The frugal lord wishes to know the meterage of under- and over guarded walls. That is, he wants to know what the sum of the lengths of the walls that are under guarded is, and the sum of the lengths of the over guarded walls. If a wall is walked over w times, it should be counted $w - 1$ times in the latter case. Thus a 5m wall that 4 guards cover is counted as 15m of over guarded wall. In other words, it is the number of "wasted" meters which guards walk.

Example

For example, if there are 4 towers, and 5 walls as shown in the diagram, and two guards with end towers 1 and 3 and 1 and 4 respectively, the guards will take the route $1 \leftrightarrow 2 \leftrightarrow 3$ and $1 \leftrightarrow 2 \leftrightarrow 3 \leftrightarrow 4$ respectively.

As a result, the two walls of length 1 are covered twice and the diagonal walls are not covered. Thus 2m of walls is over guarded and 13m is under guarded.



Input (stdin)

The first line contains three space-separated integers, N, M and K.

The next M lines contain three space-separated integers, x_i, y_i and ℓ_i indicating that there is a wall from tower x_i to tower y_i with length ℓ_i.

The next K lines contain two space-separated integers, a_i and b_i, the numbers of the starting and ending towers for the ith guard.

Sample input

```
4 5 2
2 1 1
2 3 1
1 3 3
2 4 10
3 4 7
1 3
1 4
```

Output (stdout)

Your output should consist of two integers, space-separated, representing the number of over guarded and under guarded meters respectively.

Sample output

```
2 13
```

Constraints

- $1 \leq k \leq 1\,000$
- $1 \leq n, m \leq 200\,000$

Additionally,

Subtask	Points	Constraint
1	30	$k < 5$, $n \leq 10$, $m \leq 15$
2	20	$n < 100$, $m < 1\,500$
3	50	No additional constraints.

Time limit

1 second. Python: 10 seconds.

Scoring

A correct solution will score 100% while an incorrect solution will score 0%.

A Feast for a King

Introduction

King Bruce is coming to visit your manor! You will be hosting a great feast to honour the occasion. You have decided to serve exactly two barrels of wine. The trouble is that the king brings an entourage, and you don't know how many people will be there.

To make things worse, you need to serve exactly enough wine for the number of people there, as is considered polite in the kingdom of Bruceland. To get around this, you want to tell your vineyards to start producing wine barrels in a number of different sizes, so that no matter how many people show up to the feast, you can use exactly two barrels of wine to serve them exactly.

In order to minimise the cost to your vineyards, find as small a set of barrel sizes as possible to do this.

Task

You are told that the number of people at the feast will be one of N different positive integers $X_1, X_2, \ldots, X_N$. You must find a set of positive integer barrel sizes (measured in servings) $A_1, A_2, \ldots, A_M$ such that, for any i with $1 \leq i \leq N$, you can serve exactly X_i people with exactly two barrels from the sizes listed. You can use the same barrel size twice.

Example

If there are 4, 5 or 6 people coming to the feast, then a set of barrel sizes which can serve them is $\{2, 3\}$, because $4 = 2 + 2$, $5 = 2 + 3$ and $6 = 3 + 3$.

Input (feast.in)

The first line of input contains a single integer N. The i-th of the next N lines contains a single integer X_i.

Sample input

```
3
4
5
6
```

Output (feast.out)

The first line of output should contain an integer M. The i-th of the next M lines should contain an integer A_i. The j-th of the next N lines should then contain two numbers x and y, $1 \leq x, y \leq M$, and $A_x + A_y$ must equal X_j.

Sample output

```
2
2
3
1 1
1 2
2 2
```

Constraints

- $1 \leq N \leq 1\,000\,000$
- $2 \leq X_i \leq 10^{15}$

Scoring

If you find a set of barrel sizes with M elements, and the best solution finds a set of barrel sizes with $L < N$ elements, you will score $\max\left(0, \lfloor 10\frac{N-M}{N-L} \rfloor\right)$ points for the test case.

If no one finds a solution using fewer than N barrels, then no one will score any points for that test case.

Rope

Introduction

King Æthelred has commissioned you to build a newly-developed catapult to use against the Danes. A core component of the build is two long stretches of rope, but unfortunately your supplies of rope are severely limited. You have, however, managed to collect all the little pieces of rope you can find and plan to tie them together to form the longer ropes you require.

Task

You have a collection of N pieces of rope. Each rope has a length ℓ_i and a strength s_i. If you have two pieces of rope of lengths ℓ_1 and ℓ_2 and strengths s_1 and s_2 respectively, you may join them together to form a longer rope of length $\ell_1 + \ell_2$ and strength $\min(s_1, s_2)$.

You need to create two ropes of length at least L to use in the catapult. You would clearly like these ropes to be as strong as possible so that the catapult doesn't collapse. The ropes work together in the catapult and so it is the sum of their strengths that is of interest to you.

Some ways of combining the pieces of rope together result in stronger ropes than others. Out of all these ways, determine the maximum possible value of the sum of the strengths of the resulting two ropes. You are not required to use all the pieces of rope you have available, only that each of the two ropes has length at least L.

Example

Suppose you have 5 pieces of rope with the following lengths and strengths.

Length	Strength
3	1
2	3
3	2
4	3
1	3

Suppose you require your two ropes to be no shorter than 5 units long. You can form a rope of strength 3 by joining the 4^{th} and 5^{th} ropes together and one of strength 2 by joining the 2^{nd} and 3^{rd}. Each of these have length 5 and together they have total strength $2 + 3 = 5$. This is the best possible solution.

Input (stdin)

The first line of the input contains two space-separated integers, N and L. The next N lines each contain two space-separated integers, ℓ_i and s_i, the length and strength of the i^{th} rope respectively.

Sample input

```
5 5
3 1
2 3
3 2
4 3
1 3
```

Output (stdout)

Output a single integer, the largest possible sum of the strengths of the two ropes.

Sample output

```
5
```

Constraints

- $1 \leq N \leq 1\,000$
- $1 \leq L \leq 1\,000$
- $1 \leq \ell_i \leq 1\,000$ for each rope
- $1 \leq s_i \leq 1\,000\,000$ for each rope
- There will always be at least one way to obtain ropes of the desired lengths.

Additionally, in 50% of the test cases:

- $N \leq 20$

Time limit

2 seconds. Python: 20 seconds.

Scoring

A correct solution will score 100% while an incorrect solution will score 0%.