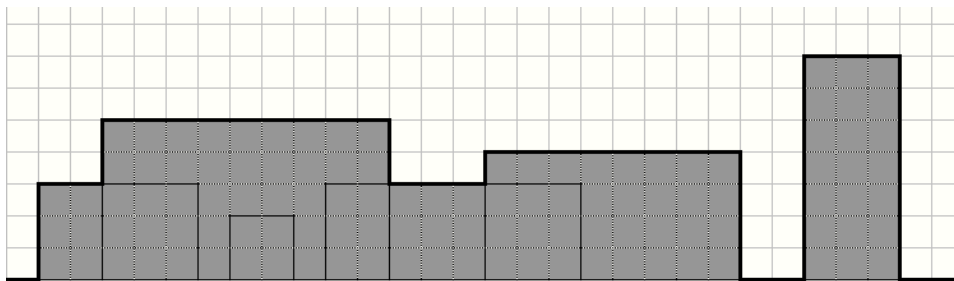


| TASK | OGRADA | PODJELA |
|--------------------------------------|-----------------|----------------|
| input | standard input | |
| output | standard output | |
| time limit (per test case) | 0.5 seconds | 1 second |
| memory limit (heap+stack) | 128 MB | |
| points | 100 | 100 |
| | 200 | |

Mirko built a wooden fence next to his house. He built his fence by slamming rectangular wooden boards into the ground. For each board we know its height, width and distance of its left edge from Mirko's house.

Mirko was really enjoying building his fence, so much that he stopped paying attention as to how he was building it – now that he is done, some of the boards are overlapping, and there are holes in the fence, total chaos!

Mirko is now looking at his fence and is shocked – he could have built a fence the same fence (when looked from up front) using fewer boards. If we connect the top sides of all boards as in the image below, we get the **skyline** of the fence.



The second example input. The skyline is depicted by a bold line.

Mirko decided to remove some of the boards from the fence so that the **skyline remains the same** and that **as few boards as possible** remain in the fence.

Write a program that determines which boards should remain in the fence.

INPUT

The first line contains an integer N ($1 \leq N \leq 100\,000$), the number of boards initially in Mirko's fence.

Each of the following N lines describes one board with three integers X , W and H ; these are the distance of the left edge of the board from Mirko's house, width and height of the board, respectively.

No two boards will share the same (X, W, H) triplet. All these numbers will be at most a billion (10^9).

Boards are numbered 1 to N .

OUTPUT

On the first line output B , the smallest number of boards that need to remain for the skyline to stay the same.

On the second line output B integers separated by spaces, the numbers of boards that should remain in the fence. This part of the output is not necessarily unique.

SCORING

For each test case, correctly solving the first part (the smallest number of remaining boards) is worth 70% of points.

EXAMPLES

| | |
|---------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------|
| input 5 10 11 30 19 1 20 15 9 10 7 7 20 3 12 40 output 3 1 3 5 | input 6 15 8 4 10 8 3 25 3 7 3 9 5 1 5 3 7 2 2 output 5 1 2 3 4 5 |
|---------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------|

The farmer's guild produced a certain amount of grain and delivered it to a large food company. The food company paid each of N farmers the **same amount** X . The farmers know that they did not all do the same amount of work so they want to redistribute the money they got more fairly.

The farmers all live in different villages. Villages are connected by roads so that there is a **unique way** to get from one village to every other. In a single **transaction** one farmer uses his tractor to visit a village **neighbouring** his own and hands **any amount** of money to the farmer living there.

For each farmer we know the amount he deserved. Your program must determine:

- The smallest number of transactions for every farmer to get the amount he deserved.
- The transactions themselves, in the correct order. When determining the order of transactions, keep in mind that a farmer cannot give an amount of money larger than he currently has.

It is possible that the food company overpaid the farmers (paid more than the farmers ask for). In such a case, the farmers don't care how the excess money is distributed, as long as each of them gets at least as much as he deserved.

INPUT

The first line contains the integer N ($1 \leq N \leq 2000$), the number of farmers.

The second line contains the integer X ($0 \leq X \leq 10000$), the amount that was paid to each farmer.

The third line contains N integers, the amounts the farmers deserved. The sum of these amounts will be at most $N \cdot X$.

The following $N-1$ lines describe the roads. Each road is described by a pair of integers between 1 and N , the indices of villages the road connects. Roads are two-way.

OUTPUT

On the first line output the smallest required number of transactions K .

On the following K lines output the transactions. Each transaction is described by three integers A , B and C . These mean that the farmer from village A travels to village B and hands C units of money to the farmer living there. There must be a road connecting villages A and B .

The transactions and their order may not be unique.

SCORING

For each test case, correctly solving the first part (smallest number of transactions) is worth 70% of points.

EXAMPLES

input

```
5
1
0 2 2 0 1
1 2
1 3
3 4
3 5
```

output

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

input

```
8
6
2 5 2 8 6 6 10 2
1 6
1 5
5 4
4 7
5 3
5 8
4 2
```

output

```
4
3 5 4
8 5 4
5 4 6
4 7 4
```