COCI 2016/2017

Round #5, January 21st, 2017

Tasks

Task	Time limit	Memory limit	Score
Tuna	1 s	64 MB	50
Pareto	1 s	64 MB	80
Unija	1 s	64 MB	100
Ronald	1 s	64 MB	120
Poklon	5 s	512 MB	140
Strelice	1 s	512 MB	160
Total			650

Fisherman Šime caught N tunas last night. With the help of a special app, he offerred them for sale to a famous Japanese company that specializes in purchasing quality fish. In what way does the app estimate the value, or the price, of a tuna?
Based on the photo of the tuna, the app returns two estimated values, P_1 and P_2. If the difference between the estimates is less than or equal to X, then the higher value is taken. If the difference is strictly larger than X, the app returns a third estimate P_3 and then that estimate is taken as the final value of the tuna.
Write a programme that will, based on the given estimates (sometimes two, sometimes three of them) for each of N tunas, output the total value of caught tunas.

INPUT

The first line of input contains the integer N ($1 \le N \le 20$), the number of tunas from the task.

The second line of input contains the integer X ($1 \le X \le 10$), the number from the task.

Then, N blocks follow in one of the two following forms:

- In one line, two integers P_1 and P_2 ($1 \le P_1, P_2 \le 100$) from the task,

or

- In one line, two integers P_1 and P_2 ($1 \le P_1, P_2 \le 100$) from the task, and in the second line integer P_3 ($1 \le P_3 \le 100$) from the task.

OUTPUT

The first and only line of output must contain the total value of caught tunas.

SAMPLE TESTS

input	input	input
5 2 3 4 2 1 5 3 4 4 4 2	4 2 3 5 2 8 4 6 5 6 3 7	3 10 20 50 30 20 40 50 70 20 10
output	**output**	**output**
19	22	90

Clarification of the second test case:

Šime caught 4 tunas. For the first tuna, the app returned two estimates (3 and 5). Since the difference between these two estimates is less than or equal to 2, the value of the first tuna is **5**. For the second tuna, the difference between the first two estimates (2 and 8) is higher than 2, so the app returned a third estimate, **4**. The third tuna's value is **6** (6 – 5 ≤ 2), and the value of the fourth tuna is taken as the third estimate, **7**, because the difference between the given estimates (6 and 3) is higher than 2.

Paret's principle, also known as the "80/20 rule", states that in many situations 80% of results come from 20% of (the most important) causes. For instance, Microsoft found that, by fixing 20% of most commonly reported bugs, they would eliminate 80% of downtime in their systems. In the business world, it is often said that 80% of income comes from 20% of the most important clients. In the world of mobile games, when it comes to games with free basic functionality, 50% of profit comes from 0.5% of players. Some say that 80% of your success will come from 20% of your activities.

It is a known fact that 80% of the world's goods is owned by 20% of (the richest) people. Your task is to check the validity of this rule based on the bank accounts owned by clients of a single bank. Is it true that 20% of accounts hold 80% of the total money? Does a stronger claim hold, for instance, that only 10% of accounts hold 85% of the total money?

More precisely: based on the given account balances of N bank clients, your task is to find **numbers *A* and *B* with the maximal difference *B - A* such that we can say that precisely *A*% of accounts hold *B*% of the total money** of all clients of the bank.

INPUT

The first line of input contains the integer N ($1 \le N \le 300\,000$), the number of clients in the bank.
The following line contains N integers from the interval [0, 100 000 000], the balances of bank accounts in euros.

OUTPUT

In two lines, you must output two real numbers from the task, A and B, omitting the percentage sign. Solution with the greatest B - A difference will be **unique**. A deviation from the official solution for less than 0.01 will be tolerated.

SAMPLE TEST

input

```
2
100 200
```

output

```
50.0
66.66666666666666
```

input

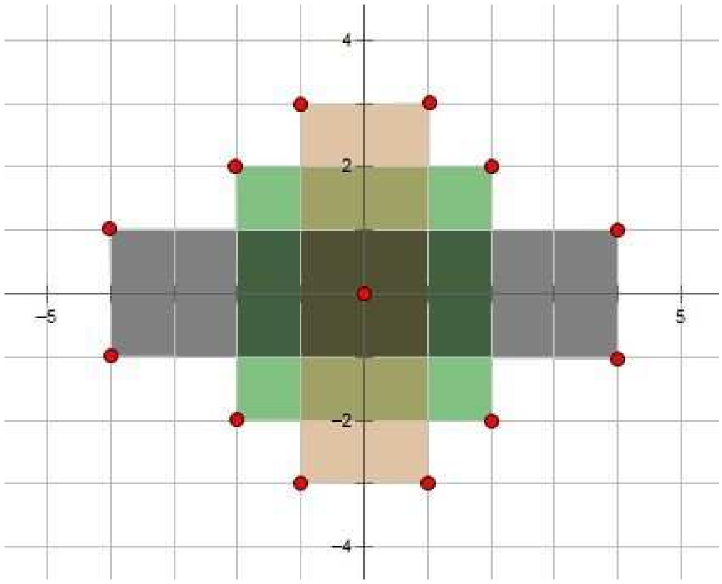
```
8
100100 10 100 1000 1 10100
90100 100100
```

output

```
37.5
96.28172769816027
```

Clarification of the first test case: 50% of accounts (one account, the one with 200 euros), contains two thirds, 66.666667% of the total money.

You are given N rectangles, which are **centered in the center** of the Cartesian coordinate system and their sides are parallel to the coordinate axes. Each rectangle is uniquely identified with its width (along the x-axis) and height (along the y-axis). The lower image depicts the first sample test.



Mirko has coloured each rectangle in a certain color and now wants to know the area of the coloured part of the paper. In other words, he wants to know the number of unit squares that belong to at least one rectangle.

INPUT

The first line of input contains the integer N ($1 \le N \le 1\,000\,000$), the number of rectangles. Each of the following N lines contains **even** integers X and Y ($2 \le X, Y \le 10^7$), dimensions (width and height, respectively) of the corresponding rectangles.

OUTPUT

The first and only line of output must contain the required area.

SCORING

In test cases worth 40% of total points, all numbers from the input will be smaller than 3333.
In test cases worth 50% of total points, not a single rectangle will be located strictly within another rectangle.

SAMPLE TESTS

input	input

```
3
8 2
4 4
2 6
```

output

```
28
```

```
5
2 10
4 4
2 2
8 8
6 6
```

output

```
68
```

There are *N* cities in one country that are connected with two-way air links. One crazy airline president, Ronald Krump, often changes the flight schedule. More precisely, every day he does the following:

- chooses one of the cities,
- introduces flights from that city towards all other cities where these flights do not currently exist, and at the same time cancels all existing flights from that city

For instance, if from city 5 flights exist towards cities 1 and 2, but not towards cities 3 and 4, after Krump's change, there will exist flights from city 5 towards cities 3 and 4, but not towards cities 1 and 2.

The citizens of this country are wondering if a day could come when the flight schedule will be complete. In other words, when between each two different cities a (direct) flight will exist. Write a programme that will, based on the current flight schedule, determine whether it is possible to have a *Complete Day*, or whether this will never happen, no matter what moves Krump makes.

INPUT

The first line of input contains the integer N ($2 \le N \le 1000$), the number of cities. The cities are labeled with numbers from 1 to N.
The second line contains the integer M ($0 \le M < N*(N-1)/2$), the number of current flights. Each of the following M lines contains two different numbers, the labels of the cities that are currently connected.

OUTPUT

The first and only line of output must contain DA (Croatian for "yes") or NE (Croatian for "no").

SAMPLE TESTS

input	input	input
2 0	3 2 1 2 2 3	4 2 1 3 2 4
output	**output**	**output**
DA	NE	DA

Clarification of the first test case: In the first step, Krump will introduce the (only possible) line 1-2.
Clarification of the third case: If Krump first chooses city 1, flights 1-2, 1-4 and 2-4 will exist. If he then chooses city 3, the flight schedule will become complete.

Little Mirko is a very simple man. Mirko's friend Darko has given him an array of *N* natural integers and asked him *Q* queries about the array that Mirko must answer.
Each query consists of two integers, the positions of the left and right end of an interval in the array. The answer to the query is the number of different values that appear exactly twice in the given interval.

INPUT

The first line of input contains the integers *N* and *Q* ($1 \le N, Q \le 500\ 000$).
The second line of input contains *N* natural integers less than 1 000 000 000, the elements of the array.
Each of the following *Q* lines contains two integers, *L* and *R* ($1 \le L \le R \le N$), from the task.

OUTPUT

The output must consist of *Q* lines, each line containing the answer to a query, respectively.

SCORING

In test cases worth 56 points in total, the numbers *N* and *Q* will not be larger than 5000.

SAMPLE TESTS

input
```
5 1
1 2 1 1 1
1 3
```
output
```
1
```

input
```
5 2
1 1 1 1 1
2 4
2 3
```
output
```
0
1
```

input
```
5 2
1 1 2 2 3
1 1
1 5
```
output
```
0
2
```

Clarification of the first test case:
In the interval from the first to the third element, there is only one number (number 1) that appears exactly twice.

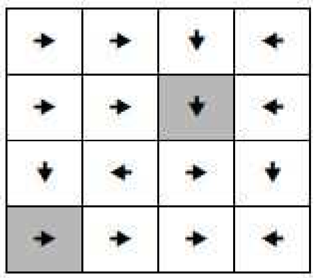
Hansel and Gretel are playing a well-known game of “Arrows” that takes place on a board with R rows and S columns. On each field there is **exactly one** arrow that points to one of the four main directions.

Hansel plays first, and his move is to colour exactly K fields on the board that **are not located** in the final column. Gretel then places a robot on an arbitrary field in the first column. Now the robot can move on his own, by moving from the current field to the field that the arrow points to. If at some point the robot is located in the last column, he stops and the game ends.

The winner of the game is determined in the following way:

- If the robot stopped and the game ended, Hansel is the winner if the robot passed through exactly one coloured field, and Gretel is the winner if the robot passed through zero or more than one coloured fields.
- If the robot did not stop after a finite amount of time (in other words, if the robot is stuck in an infinite loop), Hansel is the winner.

We consider that the robot passed through the starting field, through the fields it moved on throughout the game, and the field it was when the game ended. Also, the arrows will be drawn so that the robot never exists outside the board’s boundaries.



Determine whether Hansel can ensure his victory no matter where Gretel initially places the robot. If the answer is positive, output K fields he can colour in order to win.

INPUT

The first line of input contains the integers R, S, K (1 ≤ R * S ≤ 1 000 000, 1 ≤ K ≤ 50).
Each of the following R lines contains S characters ‘L’, ‘R’, ‘U’ or ‘D’ that denote the direction of the arrow in the corresponding field of the board (L - left, R - right, U - up, D - down).

OUTPUT

If Hansel cannot ensure his victory, output -1.
If Hansel can ensure his victory, output K lines. In each line, output space-separated numbers A and B (1 ≤ A ≤ R, 1 ≤ B ≤ S) that denote the row and column of the field that Hansel has to colour. All coloured fields must be different.
If multiple solutions exist, output any.

SAMPLE TESTS

input	input	input
4 3 1 DRD DUD DUD RUL	3 3 2 RRR RRR RRR	4 4 2 RRDL RRDL DLRD RRRL
output	**output**	**output**
4 2	-1	2 3 4 1

Clarification of the first test case:
If Hansel colours the field (4, 2), the robot will pass through it no matter where Gretel initially places it, so Hansel will be the winner.

Clarification of the second test case:
Since Hansel must colour exactly 2 fields, this means that at least one of the three rows will not contain a coloured field. Gretel can place the robot in that row and it will pass through 0 coloured fields, so Gretel will win.

Clarification of the third test case:
Consult the image from the task.