[Juniors] - Binary Search #3

A. Magic Powder - 1

1 second, 256 megabytes

This problem is given in two versions that differ only by constraints. If you can solve this problem in large constraints, then you can just write a single solution to the both versions. If you find the problem too difficult in large constraints, you can write solution to the simplified version only.

Waking up in the morning, Apollinaria decided to bake cookies. To bake one cookie, she needs n ingredients, and for each ingredient she knows the value a_i — how many grams of this ingredient one needs to bake a cookie. To prepare one cookie Apollinaria needs to use all n ingredients.

Apollinaria has b_i gram of the i-th ingredient. Also she has k grams of a magic powder. Each gram of magic powder can be turned to exactly 1 gram of any of the n ingredients and can be used for baking cookies.

Your task is to determine the maximum number of cookies, which Apollinaria is able to bake using the ingredients that she has and the magic powder.

Input

The first line of the input contains two positive integers n and k ($1 \le n, k \le 1000$) — the number of ingredients and the number of grams of the magic powder.

The second line contains the sequence $a_1, a_2, ..., a_n$ ($1 \le a_i \le 1000$), where the i-th number is equal to the number of grams of the i-th ingredient, needed to bake one cookie.

The third line contains the sequence $b_1, b_2, ..., b_n$ ($1 \le b_i \le 1000$), where the i-th number is equal to the number of grams of the i-th ingredient, which Apollinaria has.

Output

Print the maximum number of cookies, which Apollinaria will be able to bake using the ingredients that she has and the magic powder.

input

```
3 1
2 1 4
11 3 16
```

output

```
4
```

input

```
4 3
4 3 5 6
11 12 14 20
```

output

```
3
```

In the first sample it is profitably for Apollinaria to make the existing 1 gram of her magic powder to ingredient with the index 2, then Apollinaria will be able to bake 4 cookies.

In the second sample Apollinaria should turn 1 gram of magic powder to ingredient with the index 1 and 1 gram of magic powder to ingredient with the index 3. Then Apollinaria will be able to bake 3 cookies. The remaining 1 gram of the magic powder can be left, because it can't be used to increase the answer.

B. Magic Powder - 2

1 second, 256 megabytes

The term of this problem is the same as the previous one, the only exception — increased restrictions.

Input

The first line contains two positive integers n and k ($1 \le n \le 100\,000, 1 \le k \le 10^9$) — the number of ingredients and the number of grams of the magic powder.

The second line contains the sequence $a_1, a_2, ..., a_n$ ($1 \le a_i \le 10^9$), where the i-th number is equal to the number of grams of the i-th ingredient, needed to bake one cookie.

The third line contains the sequence $b_1, b_2, ..., b_n$ ($1 \le b_i \le 10^9$), where the i-th number is equal to the number of grams of the i-th ingredient, which Apollinaria has.

Output

Print the maximum number of cookies, which Apollinaria will be able to bake using the ingredients that she has and the magic powder.

input

```
1 1000000000
1
1000000000
```

output

```
2000000000
```

input

```
10 1
1000000000 1000000000 1000000000 1000000000 1000000000 1000000000
1000000000 1000000000 1000000000 1000000000
1 1 1 1 1 1 1 1 1 1
```

output

```
0
```

input

```
3 1
2 1 4
11 3 16
```

output

```
4
```

input

```
4 3
4 3 5 6
11 12 14 20
```

output

```
3
```

C. Hamburgers

1 second, 256 megabytes

Polycarpus loves hamburgers very much. He especially adores the hamburgers he makes with his own hands. Polycarpus thinks that there are only three decent ingredients to make hamburgers from: a bread, sausage and cheese. He writes down the recipe of his favorite "Le Hamburger de Polycarpus" as a string of letters 'B' (bread), 'S' (sausage) и 'C' (cheese). The ingredients in the recipe go from bottom to top, for example, recipe "BSCBS" represents the hamburger where the ingredients go from bottom to top as bread, sausage, cheese, bread and sausage again.

Polycarpus has n_b pieces of bread, n_s pieces of sausage and n_c pieces of cheese in the kitchen. Besides, the shop nearby has all three ingredients, the prices are p_b rubles for a piece of bread, p_s for a piece of sausage and p_c for a piece of cheese.

Polycarpus has r rubles and he is ready to shop on them. What maximum number of hamburgers can he cook? You can assume that Polycarpus cannot break or slice any of the pieces of bread, sausage or cheese. Besides, the shop has an unlimited number of pieces of each ingredient.

Input
The first line of the input contains a non-empty string that describes the recipe of "Le Hamburger de Polycarpus". The length of the string doesn't exceed 100, the string contains only letters 'B' (uppercase English B), 'S' (uppercase English S) and 'C' (uppercase English C).

The second line contains three integers n_b, n_s, n_c $(1 \le n_b, n_s, n_c \le 100)$ — the number of the pieces of bread, sausage and cheese on Polycarpus' kitchen. The third line contains three integers p_b, p_s, p_c $(1 \le p_b, p_s, p_c \le 100)$ — the price of one piece of bread, sausage and cheese in the shop. Finally, the fourth line contains integer r $(1 \le r \le 10^{12})$ — the number of rubles Polycarpus has.

Please, do not write the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use the `cin`, `cout` streams or the `%I64d` specifier.

Output
Print the maximum number of hamburgers Polycarpus can make. If he can't make any hamburger, print `0`.

input
```
BBBSSC
6 4 1
1 2 3
4
```
output
```
2
```

input
```
BBC
1 10 1
1 10 1
21
```
output
```
7
```

input
```
BSC
1 1 1
1 1 3
1000000000000
```
output
```
200000000001
```

D. Vasya and Robot

1 second, 256 megabytes

Vasya has got a robot which is situated on an infinite Cartesian plane, initially in the cell $(0, 0)$. Robot can perform the following four kinds of operations:

- `U` — move from (x, y) to $(x, y + 1)$;
- `D` — move from (x, y) to $(x, y - 1)$;
- `L` — move from (x, y) to $(x - 1, y)$;
- `R` — move from (x, y) to $(x + 1, y)$.

Vasya also has got a sequence of n operations. Vasya wants to modify this sequence so after performing it the robot will end up in (x, y).

Vasya wants to change the sequence so the length of changed subsegment is minimum possible. This length can be calculated as follows: $maxID - minID + 1$, where $maxID$ is the maximum index of a changed operation, and $minID$ is the minimum index of a changed operation. For example, if Vasya changes `RRRRRRR` to `RLRRLRL`, then the operations with indices 2, 5 and 7 are changed, so the length of changed subsegment is $7 - 2 + 1 = 6$. Another example: if Vasya changes `DDDD` to `DDRD`, then the length of changed subsegment is 1.

If there are no changes, then the length of changed subsegment is 0. Changing an operation means replacing it with some operation (possibly the same); Vasya can't insert new operations into the sequence or remove them.

Help Vasya! Tell him the minimum length of subsegment that he needs to change so that the robot will go from $(0, 0)$ to (x, y), or tell him that it's impossible.

Input
The first line contains one integer number n $(1 \le n \le 2 \cdot 10^5)$ — the number of operations.

The second line contains the sequence of operations — a string of n characters. Each character is either `U`, `D`, `L` or `R`.

The third line contains two integers x, y $(-10^9 \le x, y \le 10^9)$ — the coordinates of the cell where the robot should end its path.

Output
Print one integer — the minimum possible length of subsegment that can be changed so the resulting sequence of operations moves the robot from $(0, 0)$ to (x, y). If this change is impossible, print -1.

input
```
5
RURUU
-2 3
```
output
```
3
```

input
```
4
RULR
1 1
```
output
```
0
```

input
```
3
UUU
100 100
```
output
```
-1
```

In the first example the sequence can be changed to **`L`**`U`**`L`**`UU`. So the length of the changed subsegment is $3 - 1 + 1 = 3$.

In the second example the given sequence already leads the robot to (x, y), so the length of the changed subsegment is 0.

In the third example the robot can't end his path in the cell (x, y).

E. Points on Line

2 seconds, 256 megabytes

Little Petya likes points a lot. Recently his mom has presented him n points lying on the line OX. Now Petya is wondering in how many ways he can choose three distinct points so that the distance between the two farthest of them doesn't exceed d.

Note that the order of the points inside the group of three chosen points doesn't matter.

Input
The first line contains two integers: n and d $(1 \le n \le 10^5;\ 1 \le d \le 10^9)$. The next line contains n integers $x_1, x_2, ..., x_n$, their absolute value doesn't exceed 10^9 — the x-coordinates of the points that Petya has got.

It is guaranteed that the coordinates of the points in the input **strictly increase**.

Output

Print a single integer — the number of groups of three points, where the distance between two farthest points doesn't exceed d.

Please do not use the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use the `cin`, `cout` streams or the `%I64d` specifier.

input

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

output

```
4
```

input

```
4 2
-3 -2 -1 0
```

output

```
2
```

input

```
5 19
1 10 20 30 50
```

output

```
1
```

In the first sample any group of three points meets our conditions.

In the seconds sample only 2 groups of three points meet our conditions: `{-3, -2, -1}` and `{-2, -1, 0}`.

In the third sample only one group does: `{1, 10, 20}`.

F. k-Multiple Free Set

2 seconds, 256 megabytes

A k-multiple free set is a set of integers where there is no pair of integers where one is equal to another integer multiplied by k. That is, there are no two integers x and y $(x < y)$ from the set, such that $y = x \cdot k$.

You're given a set of n distinct positive integers. Your task is to find the size of it's largest k-multiple free subset.

Input

The first line of the input contains two integers n and k $(1 \le n \le 10^5, 1 \le k \le 10^9)$. The next line contains a list of n distinct positive integers $a_1, a_2, ..., a_n$ $(1 \le a_i \le 10^9)$.

All the numbers in the lines are separated by single spaces.

Output

On the only line of the output print the size of the largest k-multiple free subset of $\{a_1, a_2, ..., a_n\}$.

input

```
6 2
2 3 6 5 4 10
```

output

```
3
```

In the sample input one of the possible maximum 2-multiple free subsets is {4, 5, 6}.

G. Kefa and Company

2 seconds, 256 megabytes

Kefa wants to celebrate his first big salary by going to restaurant. However, he needs company.

Kefa has n friends, each friend will agree to go to the restaurant if Kefa asks. Each friend is characterized by the amount of money he has and the friendship factor in respect to Kefa. The parrot doesn't want any friend to feel poor compared to somebody else in the company (Kefa doesn't count). A friend feels poor if in the company there is someone who has at least d units of money more than he does. Also, Kefa wants the total friendship factor of the members of the company to be maximum. Help him invite an optimal company!

Input

The first line of the input contains two space-separated integers, n and d $(1 \le n \le 10^5, 1 \le d \le 10^9)$ — the number of Kefa's friends and the minimum difference between the amount of money in order to feel poor, respectively.

Next n lines contain the descriptions of Kefa's friends, the $(i+1)$-th line contains the description of the i-th friend of type m_i, s_i $(0 \le m_i, s_i \le 10^9)$ — the amount of money and the friendship factor, respectively.

Output

Print the maximum total friendship factir that can be reached.

input

```
4 5
75 5
0 100
150 20
75 1
```

output

```
100
```

input

```
5 100
0 7
11 32
99 10
46 8
87 54
```

output

```
111
```

In the first sample test the most profitable strategy is to form a company from only the second friend. At all other variants the total degree of friendship will be worse.

In the second sample test we can take all the friends.

H. Sagheer and Nubian Market

2 seconds, 256 megabytes

On his trip to Luxor and Aswan, Sagheer went to a Nubian market to buy some souvenirs for his friends and relatives. The market has some strange rules. It contains n different items numbered from 1 to n. The i-th item has base cost a_i Egyptian pounds. If Sagheer buys k items with indices $x_1, x_2, ..., x_k$, then the cost of item x_j is $a_{x_j} + x_j \cdot k$ for $1 \le j \le k$. In other words, the cost of an item is equal to its base cost in addition to its index multiplied by the factor k.

Sagheer wants to buy as many souvenirs as possible without paying more than S Egyptian pounds. Note that he cannot buy a souvenir more than once. If there are many ways to maximize the number of souvenirs, he will choose the way that will minimize the total cost. Can you help him with this task?

Input

The first line contains two integers n and S $(1 \le n \le 10^5$ and $1 \le S \le 10^9)$ — the number of souvenirs in the market and Sagheer's budget.

The second line contains n space-separated integers $a_1, a_2, ..., a_n$ $(1 \le a_i \le 10^5)$ — the base costs of the souvenirs.

Output

On a single line, print two integers k, T — the maximum number of souvenirs Sagheer can buy and the minimum total cost to buy these k souvenirs.

input

```
3 11
2 3 5
```

output

```
2 11
```

input

```
4 100
1 2 5 6
```

output

```
4 54
```

input

```
1 7
7
```

output

```
0 0
```

In the first example, he cannot take the three items because they will cost him $[5, 9, 14]$ with total cost 28. If he decides to take only two items, then the costs will be $[4, 7, 11]$. So he can afford the first and second items.

In the second example, he can buy all items as they will cost him $[5, 10, 17, 22]$.

In the third example, there is only one souvenir in the market which will cost him 8 pounds, so he cannot buy it.

I. Two Cakes

1 second, 256 megabytes

It's New Year's Eve soon, so Ivan decided it's high time he started setting the table. Ivan has bought two cakes and cut them into pieces: the first cake has been cut into a pieces, and the second one — into b pieces.

Ivan knows that there will be n people at the celebration (including himself), so Ivan has set n plates for the cakes. Now he is thinking about how to distribute the cakes between the plates. Ivan wants to do it in such a way that all following conditions are met:

1. Each piece of each cake is put on some plate;
2. Each plate contains at least one piece of cake;
3. No plate contains pieces of both cakes.

To make his guests happy, Ivan wants to distribute the cakes in such a way that the minimum number of pieces on the plate is maximized. Formally, Ivan wants to know the maximum possible number x such that he can distribute the cakes according to the aforementioned conditions, and each plate will contain at least x pieces of cake.

Help Ivan to calculate this number x!

Input

The first line contains three integers n, a and b ($1 \le a, b \le 100$, $2 \le n \le a + b$) — the number of plates, the number of pieces of the first cake, and the number of pieces of the second cake, respectively.

Output

Print the maximum possible number x such that Ivan can distribute the cake in such a way that each plate will contain at least x pieces of cake.

input

```
5 2 3
```

output

```
1
```

input

```
4 7 10
```

output

```
3
```

In the first example there is only one way to distribute cakes to plates, all of them will have 1 cake on it.

In the second example you can have two plates with 3 and 4 pieces of the first cake and two plates both with 5 pieces of the second cake. Minimal number of pieces is 3.

J. Worms

1 second, 256 megabytes

It is lunch time for Mole. His friend, Marmot, prepared him a nice game for lunch.

Marmot brought Mole n ordered piles of worms such that i-th pile contains a_i worms. He labeled all these worms with consecutive integers: worms in first pile are labeled with numbers 1 to a_1, worms in second pile are labeled with numbers $a_1 + 1$ to $a_1 + a_2$ and so on. See the example for a better understanding.

Mole can't eat all the worms (Marmot brought a lot) and, as we all know, Mole is blind, so Marmot tells him the labels of the best juicy worms. Marmot will only give Mole a worm if Mole says correctly in which pile this worm is contained.

Poor Mole asks for your help. For all juicy worms said by Marmot, tell Mole the correct answers.

Input

The first line contains a single integer n ($1 \le n \le 10^5$), the number of piles.

The second line contains n integers $a_1, a_2, ..., a_n$ ($1 \le a_i \le 10^3$, $a_1 + a_2 + ... + a_n \le 10^6$), where a_i is the number of worms in the i-th pile.

The third line contains single integer m ($1 \le m \le 10^5$), the number of juicy worms said by Marmot.

The fourth line contains m integers $q_1, q_2, ..., q_m$ ($1 \le q_i \le a_1 + a_2 + ... + a_n$), the labels of the juicy worms.

Output

Print m lines to the standard output. The i-th line should contain an integer, representing the number of the pile where the worm labeled with the number q_i is.

input

```
5
2 7 3 4 9
3
1 25 11
```

output

```
1
5
3
```

For the sample input:

- The worms with labels from $[1, 2]$ are in the first pile.
- The worms with labels from $[3, 9]$ are in the second pile.
- The worms with labels from $[10, 12]$ are in the third pile.
- The worms with labels from $[13, 16]$ are in the fourth pile.
- The worms with labels from $[17, 25]$ are in the fifth pile.

K. Planning The Expedition

1 second, 256 megabytes

Natasha is planning an expedition to Mars for n people. One of the important tasks is to provide food for each participant.

The warehouse has m daily food packages. Each package has some food type a_i.

Each participant must eat exactly one food package each day. Due to extreme loads, each participant must eat the same food type throughout the expedition. Different participants may eat different (or the same) types of food.

Formally, for each participant j Natasha should select his food type b_j and each day j-th participant will eat one food package of type b_j. The values b_j for different participants may be different.

What is the maximum possible number of days the expedition can last, following the requirements above?

Input

The first line contains two integers n and m ($1 \le n \le 100$, $1 \le m \le 100$) — the number of the expedition participants and the number of the daily food packages available.

The second line contains sequence of integers $a_1, a_2, \ldots, a_m$ ($1 \le a_i \le 100$), where a_i is the type of i-th food package.

Output

Print the single integer — the number of days the expedition can last. If it is not possible to plan the expedition for even one day, print `0`.

input

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

output

```
2
```

input

```
100 1
1
```

output

```
0
```

input

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

output

```
1
```

input

```
3 9
42 42 42 42 42 42 42 42 42
```

output

```
3
```

In the first example, Natasha can assign type 1 food to the first participant, the same type 1 to the second, type 5 to the third and type 2 to the fourth. In this case, the expedition can last for 2 days, since each participant can get two food packages of his food type (there will be used 4 packages of type 1, two packages of type 2 and two packages of type 5).

In the second example, there are 100 participants and only 1 food package. In this case, the expedition can't last even 1 day.

L. Physics Practical

1 second, 256 megabytes

One day Vasya was on a physics practical, performing the task on measuring the capacitance. He followed the teacher's advice and did as much as n measurements, and recorded the results in the notebook. After that he was about to show the results to the teacher, but he remembered that at the last lesson, the teacher had made his friend Petya redo the experiment because the largest and the smallest results differed by more than two times. Vasya is lazy, and he does not want to redo the experiment. He wants to do the task and go home play computer games. So he decided to cheat: before Vasya shows the measurements to the teacher, he will erase some of them, so as to make the largest and the smallest results of the remaining measurements differ in no more than two times. In other words, if the remaining measurements have the smallest result x, and the largest result y, then the inequality $y \le 2 \cdot x$ must fulfill. Of course, to avoid the teacher's suspicion, Vasya wants to remove as few measurement results as possible from his notes.

Help Vasya, find what minimum number of measurement results he will have to erase from his notes so that the largest and the smallest of the remaining results of the measurements differed in no more than two times.

Input

The first line contains integer n ($2 \le n \le 10^5$) — the number of measurements Vasya made. The second line contains n integers $c_1, c_2, \ldots, c_n$ ($1 \le c_i \le 5000$) — the results of the measurements. The numbers on the second line are separated by single spaces.

Output

Print a single integer — the minimum number of results Vasya will have to remove.

input

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

output

```
2
```

input

```
4
4 3 2 4
```

output

```
0
```

In the first sample you can remove the fourth and the sixth measurement results (values 8 and 7). Then the maximum of the remaining values will be 5, and the minimum one will be 3. Or else, you can remove the third and fifth results (both equal 3). After that the largest remaining result will be 8, and the smallest one will be 4.

M. Karen and Coffee

2.5 seconds, 512 megabytes

To stay woke and attentive during classes, Karen needs some coffee!

Karen, a coffee aficionado, wants to know the optimal temperature for brewing the perfect cup of coffee. Indeed, she has spent some time reading several recipe books, including the universally acclaimed "The Art of the Covfefe".

She knows n coffee recipes. The i-th recipe suggests that coffee should be brewed between l_i and r_i degrees, inclusive, to achieve the optimal taste.

Karen thinks that a temperature is *admissible* if at least k recipes recommend it.

Karen has a rather fickle mind, and so she asks q questions. In each question, given that she only wants to prepare coffee with a temperature between a and b, inclusive, can you tell her how many admissible integer temperatures fall within the range?

Input

The first line of input contains three integers, n, k ($1 \le k \le n \le 200000$), and q ($1 \le q \le 200000$), the number of recipes, the minimum number of recipes a certain temperature must be recommended by to be admissible, and the number of questions Karen has, respectively.

The next n lines describe the recipes. Specifically, the i-th line among these contains two integers l_i and r_i ($1 \le l_i \le r_i \le 200000$), describing that the i-th recipe suggests that the coffee be brewed between l_i and r_i degrees, inclusive.

The next q lines describe the questions. Each of these lines contains a and b, ($1 \le a \le b \le 200000$), describing that she wants to know the number of admissible integer temperatures between a and b degrees, inclusive.

Output

For each question, output a single integer on a line by itself, the number of admissible integer temperatures between a and b degrees, inclusive.

input

```
3 2 4
91 94
92 97
97 99
92 94
93 97
95 96
90 100
```

output

```
3
3
0
4
```

input

```
2 1 1
1 1
200000 200000
90 100
```

output

```
0
```

In the first test case, Karen knows 3 recipes.

1. The first one recommends brewing the coffee between 91 and 94 degrees, inclusive.
2. The second one recommends brewing the coffee between 92 and 97 degrees, inclusive.
3. The third one recommends brewing the coffee between 97 and 99 degrees, inclusive.

A temperature is *admissible* if at least 2 recipes recommend it.

She asks 4 questions.

In her first question, she wants to know the number of admissible integer temperatures between 92 and 94 degrees, inclusive. There are 3: 92, 93 and 94 degrees are all admissible.

In her second question, she wants to know the number of admissible integer temperatures between 93 and 97 degrees, inclusive. There are 3: 93, 94 and 97 degrees are all admissible.

In her third question, she wants to know the number of admissible integer temperatures between 95 and 96 degrees, inclusive. There are none.

In her final question, she wants to know the number of admissible integer temperatures between 90 and 100 degrees, inclusive. There are 4: 92, 93, 94 and 97 degrees are all admissible.

In the second test case, Karen knows 2 recipes.

1. The first one, "wikiHow to make Cold Brew Coffee", recommends brewing the coffee at exactly 1 degree.
2. The second one, "What good is coffee that isn't brewed at at least 36.3306 times the temperature of the surface of the sun?", recommends brewing the coffee at exactly 200000 degrees.

A temperature is *admissible* if at least 1 recipe recommends it.

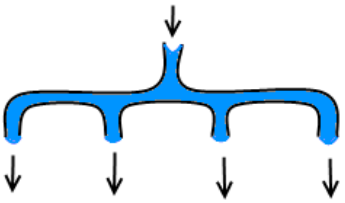
In her first and only question, she wants to know the number of admissible integer temperatures that are actually reasonable. There are none.

N. Pipeline

0.4 seconds, 256 megabytes

Vova, the Ultimate Thule new shaman, wants to build a pipeline. As there are exactly n houses in Ultimate Thule, Vova wants the city to have exactly n pipes, each such pipe should be connected to the water supply. A pipe can be connected to the water supply if there's water flowing out of it. Initially Vova has only one pipe with flowing water. Besides, Vova has several splitters.

A splitter is a construction that consists of one input (it can be connected to a water pipe) and x output pipes. When a splitter is connected to a water pipe, water flows from each output pipe. You can assume that the output pipes are ordinary pipes. For example, you can connect water supply to such pipe if there's water flowing out from it. At most one splitter can be connected to any water pipe.



The figure shows a 4-output splitter

Vova has one splitter of each kind: with 2, 3, 4, ..., k outputs. Help Vova use the minimum number of splitters to build the required pipeline or otherwise state that it's impossible.

Vova needs the pipeline to have exactly n pipes with flowing out water. Note that some of those pipes can be the output pipes of the splitters.

Input

The first line contains two space-separated integers n and k ($1 \le n \le 10^{18}$, $2 \le k \le 10^{9}$).

Please, do not use the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use the `cin`, `cout` streams or the `%I64d` specifier.

Output

Print a single integer — the minimum number of splitters needed to build the pipeline. If it is impossible to build a pipeline with the given splitters, print -1.

input

```
4 3
```

output

```
2
```

input

```
5 5
```

output

```
1
```

input

```
8 4
```

output

```
-1
```

O. Too Easy Problems

2 seconds, 256 megabytes

You are preparing for an exam on scheduling theory. The exam will last for exactly T milliseconds and will consist of n problems. You can either solve problem i in exactly t_i milliseconds or ignore it and spend no time. You don't need time to rest after solving a problem, either.

Unfortunately, your teacher considers some of the problems too easy for you. Thus, he assigned an integer a_i to every problem i meaning that the problem i can bring you a point to the final score only in case you have solved no more than a_i problems overall (including problem i).

Formally, suppose you solve problems $p_1, p_2, ..., p_k$ during the exam. Then, your final score s will be equal to the number of values of j between 1 and k such that $k \le a_{p_j}$.

You have guessed that the real first problem of the exam is already in front of you. Therefore, you want to choose a set of problems to solve during the exam maximizing your final score in advance. Don't forget that the exam is limited in time, and you must have enough time to solve all chosen problems. If there exist different sets of problems leading to the maximum final score, any of them will do.

Input

The first line contains two integers n and T ($1 \le n \le 2 \cdot 10^5$; $1 \le T \le 10^9$) — the number of problems in the exam and the length of the exam in milliseconds, respectively.

Each of the next n lines contains two integers a_i and t_i ($1 \le a_i \le n$; $1 \le t_i \le 10^4$). The problems are numbered from 1 to n.

Output

In the first line, output a single integer s — your maximum possible final score.

In the second line, output a single integer k ($0 \le k \le n$) — the number of problems you should solve.

In the third line, output k distinct integers $p_1, p_2, ..., p_k$ ($1 \le p_i \le n$) — the indexes of problems you should solve, in any order.

If there are several optimal sets of problems, you may output any of them.

input

```
5 300
3 100
4 150
4 80
2 90
2 300
```

output

```
2
3
3 1 4
```

input

```
2 100
1 787
2 788
```

output

```
0
0
```

input

```
2 100
2 42
2 58
```

output

```
2
2
1 2
```

In the first example, you should solve problems 3, 1, and 4. In this case you'll spend $80 + 100 + 90 = 270$ milliseconds, falling within the length of the exam, 300 milliseconds (and even leaving yourself 30 milliseconds to have a rest). Problems 3 and 1 will bring you a point each, while problem 4 won't. You'll score two points.

In the second example, the length of the exam is catastrophically not enough to solve even a single problem.

In the third example, you have just enough time to solve both problems in $42 + 58 = 100$ milliseconds and hand your solutions to the teacher with a smile.