A. Boys and Girls

time limit per test: 1 second
memory limit per test: 256 megabytes
input: input.txt
output: output.txt

There are n boys and m girls studying in the class. They should stand in a line so that boys and girls alternated there as much as possible. Let's assume that positions in the line are indexed from left to right by numbers from 1 to $n + m$. Then the number of integers i ($1 \leq i < n + m$) such that positions with indexes i and $i + 1$ contain children of different genders (position i has a girl and position $i + 1$ has a boy or vice versa) must be as large as possible.

Help the children and tell them how to form the line.

Input

The single line of the input contains two integers n and m ($1 \leq n, m \leq 100$), separated by a space.

Output

Print a line of $n + m$ characters. Print on the i-th position of the line character "B", if the i-th position of your arrangement should have a boy and "G", if it should have a girl.

Of course, the number of characters "B" should equal n and the number of characters "G" should equal m. If there are multiple optimal solutions, print any of them.

Examples

input

```
3 3
```

output

```
GBGBGB
```

input

```
4 2
```

output

```
BGBGBB
```

Note

In the first sample another possible answer is BGBGBG.

In the second sample answer BBGBGB is also optimal.

B. Physics Practical

time limit per test: 1 second
memory limit per test: 256 megabytes
input: input.txt
output: output.txt

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 $c_1, c_2, ..., c_n$ ($1 \le c_i \le 5000$) — 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.

Examples

input

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

output

```
2
```

input

```
4
4 3 2 4
```

output

```
0
```

Note

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.

C. Text Editor

time limit per test: 1 second
memory limit per test: 256 megabytes
input: input.txt
output: output.txt

Vasya is pressing the keys on the keyboard reluctantly, squeezing out his ideas on the classical epos depicted in Homer's Odysseus... How can he explain to his literature teacher that he isn't going to become a writer? In fact, he is going to become a programmer. So, he would take great pleasure in writing a program, but none — in writing a composition.

As Vasya was fishing for a sentence in the dark pond of his imagination, he suddenly wondered: what is the least number of times he should push a key to shift the cursor from one position to another one?

Let's describe his question more formally: to type a text, Vasya is using the text editor. He has already written n lines, the i-th line contains a_i characters (including spaces). If some line contains k characters, then this line overall contains $(k + 1)$ positions where the cursor can stand: before some character or after all characters (at the end of the line). Thus, the cursor's position is determined by a pair of integers (r, c), where r is the number of the line and c is the cursor's position in the line (the positions are indexed starting from one from the beginning of the line).

Vasya doesn't use the mouse to move the cursor. He uses keys "Up", "Down", "Right" and "Left". When he pushes each of these keys, the cursor shifts in the needed direction. Let's assume that before the corresponding key is pressed, the cursor was located in the position (r, c), then Vasya pushed key:

- "Up": if the cursor was located in the first line ($r = 1$), then it does not move. Otherwise, it moves to the previous line (with number $r - 1$), to the same position. At that, if the previous line was short, that is, the cursor couldn't occupy position c there, the cursor moves to the last position of the line with number $r - 1$;
- "Down": if the cursor was located in the last line ($r = n$), then it does not move. Otherwise, it moves to the next line (with number $r + 1$), to the same position. At that, if the next line was short, that is, the cursor couldn't occupy position c there, the cursor moves to the last position of the line with number $r + 1$;
- "Right": if the cursor can move to the right in this line ($c < a_r + 1$), then it moves to the right (to position $c + 1$). Otherwise, it is located at the end of the line and doesn't move anywhere when Vasya presses the "Right" key;
- "Left": if the cursor can move to the left in this line ($c > 1$), then it moves to the left (to position $c - 1$). Otherwise, it is located at the beginning of the line and doesn't move anywhere when Vasya presses the "Left" key.

You've got the number of lines in the text file and the number of characters, written in each line of this file. Find the least number of times Vasya should push the keys, described above, to shift the cursor from position (r_1, c_1) to position (r_2, c_2).

Input

The first line of the input contains an integer n ($1 \le n \le 100$) — the number of lines in the file. The second line contains n integers $a_1, a_2, ..., a_n$ ($0 \le a_i \le 10^5$), separated by single spaces. The third line contains four integers r_1, c_1, r_2, c_2 ($1 \le r_1, r_2 \le n, 1 \le c_1 \le a_{r_1} + 1, 1 \le c_2 \le a_{r_2} + 1$).

Output

Print a single integer — the minimum number of times Vasya should push a key to move the cursor from position (r_1, c_1) to position (r_2, c_2).

Examples

input

```
4
2 1 6 4
3 4 4 2
```

output

```
3
```

input

```
4
10 5 6 4
1 11 4 2
```

output

```
6
```

input

```
3
10 1 10
1 10 1 1
```

output

```
3
```

Note

In the first sample the editor contains four lines. Let's represent the cursor's possible positions in the line as numbers. Letter s represents the cursor's initial position, letter t represents the last one. Then all possible positions of the cursor in the text editor are described by the following table.

```
123

12

123s567

1t345
```

One of the possible answers in the given sample is: "Left", "Down", "Left".

D. Table with Letters - 2

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: input.txt
output: output.txt

Vasya has recently started to learn English. Now he needs to remember how to write English letters. He isn't sure about some of them, so he decided to train a little.

He found a sheet of squared paper and began writing arbitrary English letters there. In the end Vasya wrote n lines containing m characters each. Thus, he got a rectangular $n \times m$ table, each cell of the table contained some English letter. Let's number the table rows from top to bottom with integers from 1 to n, and columns — from left to right with integers from 1 to m.

After that Vasya looked at the resulting rectangular table and wondered, how many subtables are there, that matches both following conditions:

- the subtable contains at most k cells with "a" letter;
- all letters, located in all four corner cells of the subtable, are equal.

Formally, a subtable's definition is as follows. It is defined by four integers x_1, y_1, x_2, y_2 such that $1 \le x_1 < x_2 \le n$, $1 \le y_1 < y_2 \le m$. Then the subtable contains all such cells (x, y) (x is the row number, y is the column number), for which the following inequality holds $x_1 \le x \le x_2, y_1 \le y \le y_2$. The corner cells of the table are cells (x_1, y_1), (x_1, y_2), (x_2, y_1), (x_2, y_2).

Vasya is already too tired after he's been writing letters to a piece of paper. That's why he asks you to count the value he is interested in.

Input

The first line contains three integers n, m, k ($2 \le n, m \le 400$; $0 \le k \le n \cdot m$).

Next n lines contain m characters each — the given table. Each character of the table is a lowercase English letter.

Output

Print a single integer — the number of required subtables.

Examples

input

```
3 4 4
aabb
baab
baab
```

output

```
2
```

input

```
4 5 1
ababa
ccaca
ccacb
cbabc
```

output

```
1
```

Note

There are two suitable subtables in the first sample: the first one's upper left corner is cell $(2, 2)$ and lower right corner is cell $(3, 3)$, the second one's upper left corner is cell $(2, 1)$ and lower right corner is cell $(3, 4)$.

E. Printer

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: input.txt
output: output.txt

Let's consider a network printer that functions like that. It starts working at time 0. In each second it can print one page of a text. At some moments of time the printer receives printing tasks. We know that a printer received n tasks. Let's number the tasks by consecutive integers from 1 to n. Then the task number i is characterised by three integers: t_i is the time when the task came, s_i is the task's volume (in pages) and p_i is the task's priority. The priorities of all tasks are distinct.

When the printer receives a task, the task goes to the queue and remains there until all pages from this task are printed. The printer chooses a page to print each time when it either stops printing some page or when it is free and receives a new task. Among all tasks that are in the queue at this moment, the printer chooses the task with the highest priority and next second prints an unprinted page from this task. You can assume that a task goes to the queue immediately, that's why if a task has just arrived by time t, the printer can already choose it for printing.

You are given full information about all tasks except for one: you don't know this task's priority. However, we know the time when the last page from this task was finished printing. Given this information, find the unknown priority value and determine the moments of time when the printer finished printing each task.

Input

The first line contains integer n ($1 \le n \le 50000$). Next n lines describe the tasks. The i-th of these lines contains three integers t_i, s_i and p_i, separated by single spaces ($0 \le t_i \le 10^9$, $1 \le s_i, p_i \le 10^9$). Exactly one task (let's assume that his number is x) has number -1 written instead of the priority. All priorities are different. The last line contains integer T — the time when the printer finished printing the last page of task x ($1 \le T \le 10^{15}$). Numbers t_i are not necessarily distinct. The tasks in the input are written in the arbitrary order.

Output

In the first line print integer p_x — the priority of the task number x ($1 \le p_x \le 10^9$, remember that all priorities should be distinct). Then print n integers, the i-th of them represents the moment of time when the last page of the task number i finished printing.

It is guaranteed that at least one solution exists. If there are multiple solutions, print any of them.

Examples

input

```
3
4 3 -1
0 2 2
1 3 3
7
```

output

```
4
7 8 4
```

input

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

output

```
4
7 6 4
```

Note

Let's consider the first test case. Let's assume that the unknown priority equals 4, then the printer's actions for each second are as follows:

- the beginning of the 1-st second (time 0). The queue has task 2. The printer prints the first page of this task;
- the beginning of the 2-nd second (time 1). The queue has tasks 2 and 3. The printer prints the first page of task 3;
- the beginning of the 3-rd second (time 2). The queue has tasks 2 and 3. The printer prints the second page of task 3;
- the beginning of the 4-th second (time 3). The queue has tasks 2 and 3. The printer prints the third (last) page of task 3. Thus, by the end of the 4-th second this task will have been printed;
- the beginning of the 5-th second (time 4). The queue has tasks 2 and 1. The printer prints the first page of task 1;
- the beginning of the 6-th second (time 5). The queue has tasks 2 and 1. The printer prints the second page of task 1;
- the beginning of the 7-th second (time 6). The queue has tasks 2 and 1. The printer prints the third (last) page of task 1. Thus, by the end of the 7-th second this task will have been printed;

- the beginning of the 8-th second (time 7). The queue has task 2. The printer prints the second (last) page of task 2. Thus, by the end of the 8-th second this task will have been printed.

In the end, task number 1 will have been printed by the end of the 7-th second, as was required. And tasks 2 and 3 are printed by the end of the of the 8-th and the 4-th second correspondingly.