Codeforces Round #322 (Div. 2)

A. Vasya the Hipster

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

One day Vasya the Hipster decided to count how many socks he had. It turned out that he had a red socks and b blue socks.

According to the latest fashion, hipsters should wear the socks of different colors: a red one on the left foot, a blue one on the right foot.

Every day Vasya puts on new socks in the morning and throws them away before going to bed as he doesn't want to wash them.

Vasya wonders, what is the maximum number of days when he can dress fashionable and wear different socks, and after that, for how many days he can then wear the same socks until he either runs out of socks or cannot make a single pair from the socks he's got.

Can you help him?

Input
The single line of the input contains two positive integers a and b $(1 \leq a, b \leq 100)$ — the number of red and blue socks that Vasya's got.

Output
Print two space-separated integers — the maximum number of days when Vasya can wear different socks and the number of days when he can wear the same socks until he either runs out of socks or cannot make a single pair from the socks he's got.

Keep in mind that at the end of the day Vasya throws away the socks that he's been wearing on that day.

Sample test(s)

input
3 1
output
1 1

input
2 3
output
2 0

input
7 3
output
3 2

Note
In the first sample Vasya can first put on one pair of different socks, after that he has two red socks left to wear on the second day.

B. Luxurious Houses

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

The capital of Berland has n multifloor buildings. The architect who built up the capital was very creative, so all the houses were built in one row.

Let's enumerate all the houses from left to right, starting with one. A house is considered to be *luxurious* if the number of floors in it is strictly greater than in all the houses with larger numbers. In other words, a house is luxurious if the number of floors in it is strictly greater than in all the houses, which are located to the right from it. In this task it is assumed that the heights of floors in the houses are the same.

The new architect is interested in n questions, i-th of them is about the following: "how many floors should be added to the i-th house to make it luxurious?" (for all i from 1 to n, inclusive). You need to help him cope with this task.

Note that all these questions are independent from each other — the answer to the question for house i does not affect other answers (i.e., the floors to the houses are not actually added).

Input

The first line of the input contains a single number n ($1 \le n \le 10^5$) — the number of houses in the capital of Berland.

The second line contains n space-separated positive integers h_i ($1 \le h_i \le 10^9$), where h_i equals the number of floors in the i-th house.

Output

Print n integers $a_1, a_2, ..., a_n$, where number a_i is the number of floors that need to be added to the house number i to make it luxurious. If the house is already luxurious and nothing needs to be added to it, then a_i should be equal to zero.

All houses are numbered from left to right, starting from one.

Sample test(s)

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

input
4 3 2 1 4
output
2 3 4 0

C. Developing Skills

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

Petya loves computer games. Finally a game that he's been waiting for so long came out!

The main character of this game has n different skills, each of which is characterized by an integer a_i from 0 to 100. The higher the number a_i is, the higher is the i-th skill of the character. The total rating of the character is calculated as the sum of the values of $\lfloor \frac{a_i}{10} \rfloor$ for all i from 1 to n. The expression $\lfloor x \rfloor$ denotes the result of rounding the number x *down* to the nearest integer.

At the beginning of the game Petya got k improvement units as a bonus that he can use to increase the skills of his character and his total rating. One improvement unit can increase any skill of Petya's character by exactly one. For example, if $a_4 = 46$, after using one imporvement unit to this skill, it becomes equal to 47. A hero's skill cannot rise higher more than 100. Thus, it is permissible that some of the units will remain unused.

Your task is to determine the optimal way of using the improvement units so as to maximize the overall rating of the character. It is not necessary to use all the improvement units.

Input

The first line of the input contains two positive integers n and k ($1 \le n \le 10^5$, $0 \le k \le 10^7$) — the number of skills of the character and the number of units of improvements at Petya's disposal.

The second line of the input contains a sequence of n integers a_i ($0 \le a_i \le 100$), where a_i characterizes the level of the i-th skill of the character.

Output

The first line of the output should contain a single non-negative integer — the maximum total rating of the character that Petya can get using k or less improvement units.

Sample test(s)

input
2 4 7 9
output
2

input
3 8 17 15 19
output
5

input
2 2 99 100
output
20

Note

In the first test case the optimal strategy is as follows. Petya has to improve the first skill to 10 by spending 3 improvement units, and the second skill to 10, by spending one improvement unit. Thus, Petya spends all his improvement units and the total rating of the character becomes equal to $lfloor\ frac\{100\}\{10\}\ rfloor + \ lfloor\ frac\{100\}\{10\}\ rfloor = 10 + 10 =$ 20.

In the second test the optimal strategy for Petya is to improve the first skill to 20 (by spending 3 improvement units) and to improve the third skill to 20 (in this case by spending 1 improvement units). Thus, Petya is left with 4 improvement units and he will be able to increase the second skill to 19 (which does not change the overall rating, so Petya does not necessarily have to do it). Therefore, the highest possible total rating in this example is $\lfloor \frac{20}{10} \rfloor + \lfloor \frac{19}{10} \rfloor + \lfloor \frac{20}{10} \rfloor = 2 + 1 + 2 = 5$.

In the third test case the optimal strategy for Petya is to increase the first skill to 100 by spending 1 improvement unit. Thereafter, both skills of the character will be equal to 100, so Petya will not be able to spend the remaining improvement unit. So the answer is equal to $\lfloor \frac{100}{10} \rfloor + \lfloor \frac{100}{10} \rfloor = 10 + 10 = 20$.

D. Three Logos

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

Three companies decided to order a billboard with pictures of their logos. A billboard is a big *square* board. A logo of each company is a rectangle of a non-zero area.

Advertisers will put up the ad only if it is possible to place all three logos on the billboard so that they do not overlap and the billboard has no empty space left. When you put a logo on the billboard, you should rotate it so that the sides were parallel to the sides of the billboard.

Your task is to determine if it is possible to put the logos of all the three companies on some square billboard without breaking any of the described rules.

Input

The first line of the input contains six positive integers $x_1, y_1, x_2, y_2, x_3, y_3$ ($1 \le x_1, y_1, x_2, y_2, x_3, y_3 \le 100$), where x_i and y_i determine the length and width of the logo of the i-th company respectively.

Output

If it is impossible to place all the three logos on a square shield, print a single integer "`-1`" (without the quotes).

If it is possible, print in the first line the length of a side of square n, where you can place all the three logos. Each of the next n lines should contain n uppercase English letters "`A`", "`B`" or "`C`". The sets of the same letters should form solid rectangles, provided that:

- the sizes of the rectangle composed from letters "`A`" should be equal to the sizes of the logo of the first company,
- the sizes of the rectangle composed from letters "`B`" should be equal to the sizes of the logo of the second company,
- the sizes of the rectangle composed from letters "`C`" should be equal to the sizes of the logo of the third company,

Note that the logos of the companies can be rotated for printing on the billboard. The billboard mustn't have any empty space. If a square billboard can be filled with the logos in multiple ways, you are allowed to print any of them.

See the samples to better understand the statement.

Sample test(s)

input
```
5 1 2 5 5 2
```
output
```
5
AAAAA
BBBBB
BBBBB
CCCCC
CCCCC
```

input
```
4 4 2 6 4 2
```
output
```
6
BBBBBB
BBBBBB
AAAACC
AAAACC
AAAACC
AAAACC
```

E. Kojiro and Furrari

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

Motorist Kojiro spent 10 years saving up for his favorite car brand, Furrari. Finally Kojiro's dream came true! Kojiro now wants to get to his girlfriend Johanna to show off his car to her.

Kojiro wants to get to his girlfriend, so he will go to her along a coordinate line. For simplicity, we can assume that Kojiro is at the point f of a coordinate line, and Johanna is at point e. Some points of the coordinate line have gas stations. Every gas station fills with only one type of fuel: *Regular-92*, *Premium-95* or *Super-98*. Thus, each gas station is characterized by a pair of integers t_i and x_i — the number of the gas type and its position.

One liter of fuel is enough to drive for exactly 1 km (this value does not depend on the type of fuel). Fuels of three types differ only in quality, according to the research, that affects the lifetime of the vehicle motor. A Furrari tank holds exactly s liters of fuel (regardless of the type of fuel). At the moment of departure from point f Kojiro's tank is completely filled with fuel *Super-98*. At each gas station Kojiro can fill the tank with any amount of fuel, but of course, at no point in time, the amount of fuel in the tank can be more than s liters. Note that the tank **can** simultaneously have different types of fuel. The car can moves both left and right.

To extend the lifetime of the engine Kojiro seeks primarily to minimize the amount of fuel of type *Regular-92*. If there are several strategies to go from f to e, using the minimum amount of fuel of type *Regular-92*, it is necessary to travel so as to minimize the amount of used fuel of type *Premium-95*.

Write a program that can for the m possible positions of the start f_i minimize firstly, the amount of used fuel of type *Regular-92* and secondly, the amount of used fuel of type *Premium-95*.

Input

The first line of the input contains four positive integers e, s, n, m ($1 \le e, s \le 10^9$, $1 \le n, m \le 2 \cdot 10^5$) — the coordinate of the point where Johanna is, the capacity of a Furrari tank, the number of gas stations and the number of starting points.

Next n lines contain two integers each t_i, p_i ($1 \le t_i \le 3$, $-10^9 \le x_i \le 10^9$), representing the type of the i-th gas station (1 represents *Regular-92*, 2 — *Premium-95* and 3 — *Super-98*) and the position on a coordinate line of the i-th gas station. Gas stations don't necessarily follow in order from left to right.

The last line contains m integers f_i ($-10^9 \le f_i < e$). Start positions don't necessarily follow in order from left to right.

No point of the coordinate line contains more than one gas station. It is possible that some of points f_i or point e coincide with a gas station.

Output

Print exactly m lines. The i-th of them should contain two integers — the minimum amount of gas of type *Regular-92* and type *Premium-95*, if Kojiro starts at point f_i. First you need to minimize the first value. If there are multiple ways to do it, you need to also minimize the second value.

If there is no way to get to Johanna from point f_i, the i-th line should look like that "`-1 -1`" (two numbers minus one without the quotes).

Sample test(s)

input
```
8 4 1 1
2 4
0
```
output
```
0 4
```

input
```
9 3 2 3
2 3
1 6
-1 0 1
```
output
```
-1 -1
3 3
3 2
```

input
```
20 9 2 4
1 5
2 10
-1 0 1 2
```
output
```
-1 -1
-1 -1
-1 -1
```

```
-1 -1
```

F. Zublicanes and Mumocrates

time limit per test: 3 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

It's election time in Berland. The favorites are of course parties of zublicanes and mumocrates. The election campaigns of both parties include numerous demonstrations on n main squares of the capital of Berland. Each of the n squares certainly can have demonstrations of only one party, otherwise it could lead to riots. On the other hand, both parties have applied to host a huge number of demonstrations, so that on all squares demonstrations must be held. Now the capital management will distribute the area between the two parties.

Some pairs of squares are connected by $(n - 1)$ bidirectional roads such that between any pair of squares there is a unique way to get from one square to another. Some squares are on the outskirts of the capital meaning that they are connected by a road with only one other square, such squares are called *dead end* squares.

The mayor of the capital instructed to distribute all the squares between the parties so that the *dead end* squares had the same number of demonstrations of the first and the second party. It is guaranteed that the number of dead end squares of the city is even.

To prevent possible conflicts between the zublicanes and the mumocrates it was decided to minimize the number of roads connecting the squares with the distinct parties. You, as a developer of the department of distributing squares, should determine this smallest number.

Input

The first line of the input contains a single integer n $(2 \le n \le 5000)$ — the number of squares in the capital of Berland.

Next $n - 1$ lines contain the pairs of integers x, y $(1 \le x, y \le n, x \ne y)$ — the numbers of the squares connected by the road. All squares are numbered with integers from 1 to n. It is guaranteed that the number of dead end squares of the city is even.

Output

Print a single number — the minimum number of roads connecting the squares with demonstrations of different parties.

Sample test(s)

input
8 1 4 2 4 3 4 6 5 7 5 8 5 4 5
output
1

input
5 1 2 1 3 1 4 1 5
output
2