A. Keyboard

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

Our good friend Mole is trying to code a big message. He is typing on an unusual keyboard with characters arranged in following way:

```
qwertyuiop
asdfghjkl;
zxcvbnm,./
```

Unfortunately Mole is blind, so sometimes it is problem for him to put his hands accurately. He accidentally moved both his hands with one position to the left or to the right. That means that now he presses not a button he wants, but one neighboring button (left or right, as specified in input).

We have a sequence of characters he has typed and we want to find the original message.

Input

First line of the input contains one letter describing direction of shifting ('L' or 'R' respectively for left or right).

Second line contains a sequence of characters written by Mole. The size of this sequence will be no more than 100. Sequence contains only symbols that appear on Mole's keyboard. It doesn't contain spaces as there is no space on Mole's keyboard.

It is guaranteed that even though Mole hands are moved, he is still pressing buttons on keyboard and not hitting outside it.

Output

Print a line that contains the original message.

Examples

input

```
R
s;;upimrrfod;pbr
```

output

```
allyouneedislove
```

B. Worms

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

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 $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 10^3$, $a_1 + a_2 + \dots + a_n \le 10^6$), 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 $q_1, q_2, \dots, q_m$ ($1 \le q_i \le a_1 + a_2 + \dots + a_n$), 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.

Examples

input

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

output

```
1
5
3
```

Note

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.

C. Captain Marmot

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

Captain Marmot wants to prepare a huge and important battle against his enemy, Captain Snake. For this battle he has n regiments, each consisting of 4 moles.

Initially, each mole i ($1 \le i \le 4n$) is placed at some position (x_i, y_i) in the Cartesian plane. Captain Marmot wants to move some moles to make the regiments **compact**, if it's possible.

Each mole i has a home placed at the position (a_i, b_i). Moving this mole one time means rotating his position point (x_i, y_i) 90 degrees counter-clockwise around it's home point (a_i, b_i).

A regiment is **compact** only if the position points of the 4 moles form a square with non-zero area.

Help Captain Marmot to find out for each regiment the minimal number of moves required to make that regiment compact, if it's possible.

Input

The first line contains one integer n ($1 \le n \le 100$), the number of regiments.

The next $4n$ lines contain 4 integers x_i, y_i, a_i, b_i ($-10^4 \le x_i, y_i, a_i, b_i \le 10^4$).

Output

Print n lines to the standard output. If the regiment i can be made compact, the i-th line should contain one integer, the minimal number of required moves. Otherwise, on the i-th line print "-1" (without quotes).

Examples

input

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

output

```
1
-1
3
3
```

Note

In the first regiment we can move once the second or the third mole.

We can't make the second regiment compact.

In the third regiment, from the last 3 moles we can move once one and twice another one.

In the fourth regiment, we can move **twice** the first mole and **once** the third mole.

D. Flowers

time limit per test: 1.5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

We saw the little game Marmot made for Mole's lunch. Now it's Marmot's dinner time and, as we all know, Marmot eats flowers. At every dinner he eats some red and white flowers. Therefore a dinner can be represented as a sequence of several flowers, some of them white and some of them red.

But, for a dinner to be tasty, there is a rule: Marmot wants to eat white flowers only in groups of size k.

Now Marmot wonders in how many ways he can eat between a and b flowers. As the number of ways could be very large, print it modulo 1000000007 ($10^9 + 7$).

Input

Input contains several test cases.

The first line contains two integers t and k ($1 \le t, k \le 10^5$), where t represents the number of test cases.

The next t lines contain two integers a_i and b_i ($1 \le a_i \le b_i \le 10^5$), describing the i-th test.

Output

Print t lines to the standard output. The i-th line should contain the number of ways in which Marmot can eat between a_i and b_i flowers at dinner modulo 1000000007 ($10^9 + 7$).

Examples

input

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

output

```
6
5
5
```

Note

- For $K = 2$ and length 1 Marmot can eat (R).
- For $K = 2$ and length 2 Marmot can eat (RR) and (WW).
- For $K = 2$ and length 3 Marmot can eat (RRR), (RWW) and (WWR).
- For $K = 2$ and length 4 Marmot can eat, for example, ($WWWW$) or ($RWWR$), but for example he can't eat ($WWWR$).

E. Pillars

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

Marmot found a row with n pillars. The i-th pillar has the height of h_i meters. Starting from one pillar i_1, Marmot wants to jump on the pillars i_2, ..., i_k. ($1 \le i_1 < i_2 < \ldots < i_k \le n$). From a pillar i Marmot can jump on a pillar j only if $i < j$ and $|h_i - h_j| \ge d$, where $|x|$ is the absolute value of the number x.

Now Marmot is asking you find out a jump sequence with maximal length and print it.

Input

The first line contains two integers n and d ($1 \le n \le 10^5$, $0 \le d \le 10^9$).

The second line contains n numbers $h_1, h_2, \ldots, h_n$ ($1 \le h_i \le 10^{15}$).

Output

The first line should contain one integer k, the maximal length of a jump sequence.

The second line should contain k integers $i_1, i_2, \ldots, i_k$ ($1 \le i_1 < i_2 < \ldots < i_k \le n$), representing the pillars' indices from the maximal length jump sequence.

If there is more than one maximal length jump sequence, print any.

Examples

input

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

output

```
4
1 2 3 5
```

input

```
10 3
2 1 3 6 9 11 7 3 20 18
```

output

```
6
1 4 6 7 8 9
```

Note

In the first example Marmot chooses the pillars $1, 2, 3, 5$ with the heights $1, 3, 6, 4$. Another jump sequence of length 4 is $1, 2, 4, 5$.

F. Ant colony

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

Mole is hungry again. He found one ant colony, consisting of n ants, ordered in a row. Each ant i ($1 \le i \le n$) has a strength s_i.

In order to make his dinner more interesting, Mole organizes a version of «Hunger Games» for the ants. He chooses two numbers l and r ($1 \le l \le r \le n$) and each pair of ants with indices between l and r (inclusively) will fight. When two ants i and j fight, ant i gets one battle point only if s_i divides s_j (also, ant j gets one battle point only if s_j divides s_i).

After all fights have been finished, Mole makes the ranking. An ant i, with v_i battle points obtained, is going to be freed only if $v_i = r - l$, or in other words only if it took a point in every fight it participated. After that, Mole eats the rest of the ants. Note that there can be many ants freed or even none.

In order to choose the best sequence, Mole gives you t segments $[l_i, r_i]$ and asks for each of them how many ants is he going to eat if those ants fight.

Input

The first line contains one integer n ($1 \le n \le 10^5$), the size of the ant colony.

The second line contains n integers $s_1, s_2, \ldots, s_n$ ($1 \le s_i \le 10^9$), the strengths of the ants.

The third line contains one integer t ($1 \le t \le 10^5$), the number of test cases.

Each of the next t lines contains two integers l_i and r_i ($1 \le l_i \le r_i \le n$), describing one query.

Output

Print to the standard output t lines. The i-th line contains number of ants that Mole eats from the segment $[l_i, r_i]$.

Examples

input

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

output

```
4
4
1
1
```

Note

In the first test battle points for each ant are $v = [4, 0, 2, 0, 2]$, so ant number 1 is freed. Mole eats the ants $2, 3, 4, 5$.

In the second test case battle points are $v = [0, 2, 0, 2]$, so no ant is freed and all of them are eaten by Mole.

In the third test case battle points are $v = [2, 0, 2]$, so ants number 3 and 5 are freed. Mole eats **only the ant 4**.

In the fourth test case battle points are $v = [0, 1]$, so ant number 5 is freed. Mole eats the ant 4.