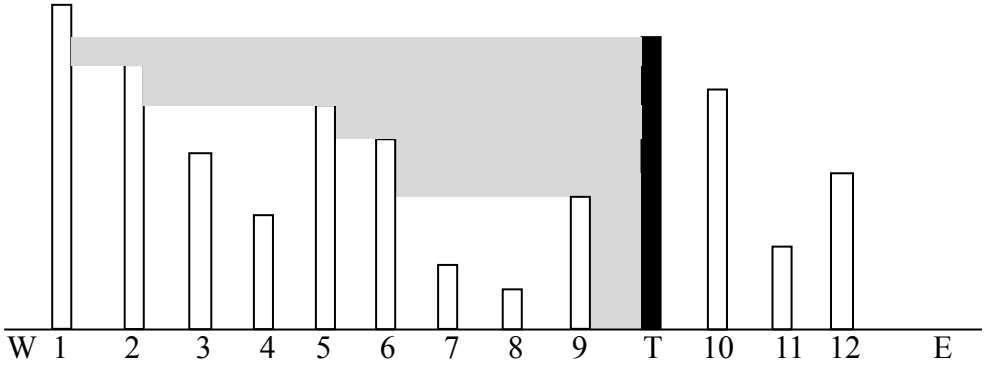
INTERNATIONAL TOURNAMENT IN INFORMATICS
26 November, 2016, Shumen, Bulgaria
Senior Group

Task A1. TOWERS

City X consists of N buildings, ordered in a row from west to east and numbered from 1 to N. **Each building has a different height – an integer number, respectively h_1, h_2, …, h_N**. The city government plans to build a tower, which will be in the same row as the buildings (it can be before the first building, between any two of the buildings or after the last building). The tower will broadcast messages to the citizens. **The tower must have height H, which should be different from all other buildings' heights**.

Due to some strange engineering ideas, the tower will be able to broadcast signals only to the west (to the beginning of the buildings' row). The signals are also strange – they are rays which travel horizontally (parallel to the ground, which we consider as a straight line) and are emitted out of the whole body of the tower (from the top to the bottom). So we can imagine that the tower radiates a continuous band of signals with width equal to the tower's height. When a ray hits a building, it stops. **Each building receives the signals using a receiver located on its top**. A building receives a message if at least one ray reaches its receiver.

In other words, a building numbered i will receive messages from the tower only when: the building i is to the west of the tower; i is not higher than the tower; and there is no other building j between them ($j>i$), which is higher than building i.



Take a look at the example in the figure above: the buildings, which are able to receive messages are with numbers 2, 5, 6, and 9.

Only one tower will be built, however the city government has received offers for K tower variants, each of different height (and having different building cost). The offered towers are numbered from 1 to K. Each of these towers has its height, which is also different from all the heights of buildings in the town. The city leaders would like to know the maximal number of buildings, which would receive messages, for each of the offered K towers, before they make their decision which one to accept. Of course, calculations should be made assuming optimal placement of each tower.

Write a program **`towers`** to determine the maximum number of buildings, which would receive messages for each of the K offers. You will be given the row of buildings in the town (actually, their heights) and the heights of all offered towers. Certainly, you have to consider the optimal placement for each tower.

Input

Two space separated positive integers are given on the first row of the standard input: N and K – the number of buildings and the number of offered towers.
N space separated positive integers are input from the second row – the heights of the buildings in the town, ordered by the building numbers (from the first to the N-th).

The third row consists of K space separated positive integers – the heights of the offered towers.

Output

The program should write on a single row of the standard output K space separated non-negative integers: for each offer in the third input row – the maximal number of buildings which would receive messages, if the tower were built, assuming optimal placement.

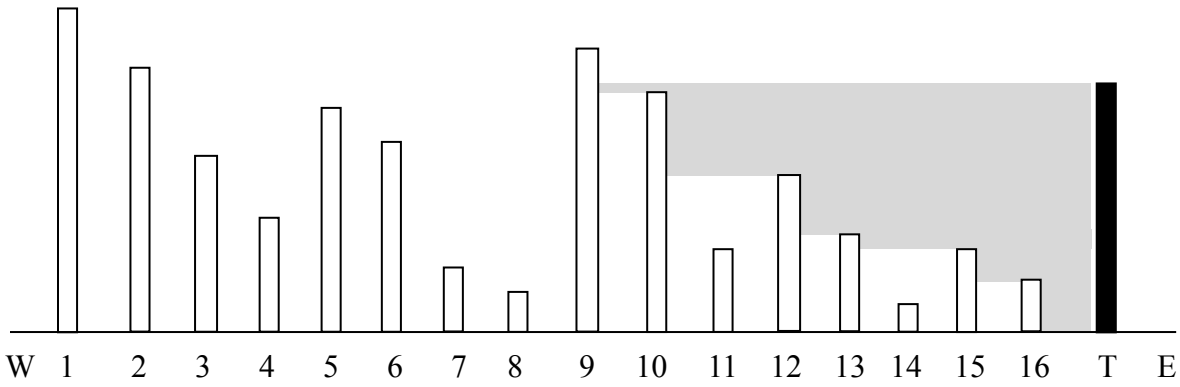
Constraints

$1 \le N \le 1\ 000\ 000$;
$1 \le K \le 100\ 000$;
$1 \le$ height of each building and offered tower $\le 10^9$
In 20% of the test cases $N \le 1000$, $K \le 20$
In another 30% of the test cases $N \le 1\ 000\ 000$, $K \le 20$

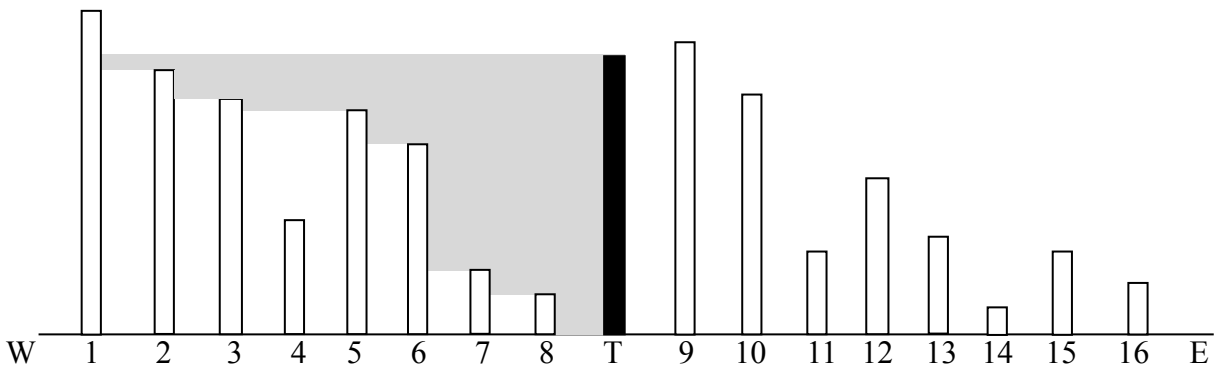
Example

Input	**Output**
16 3 200 170 155 90 150 140 40 30 185 160 50 110 80 15 70 35 165 180 120	5 6 4

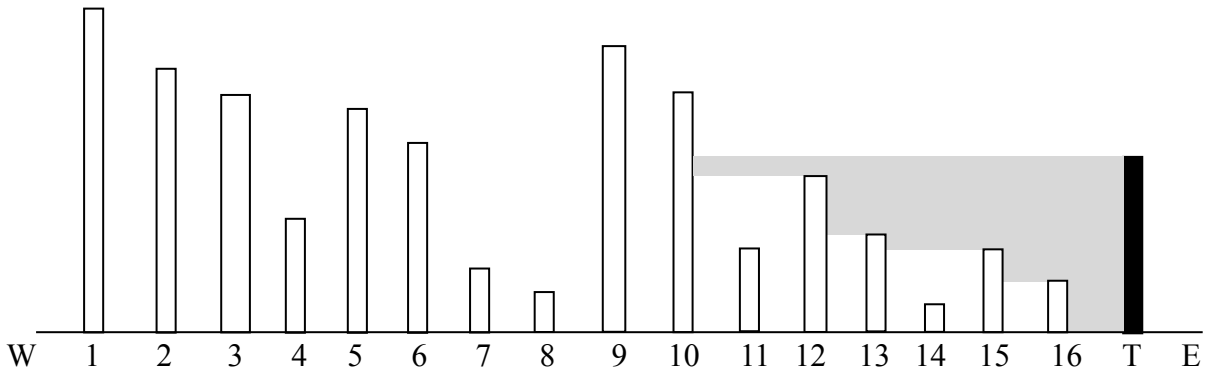
Explanation: Optimal locations of each tower are shown in the pictures below.



Location of tower 1. Buildings that are able to receive messages are 10, 12, 13, 15, and 16



Location of tower 2. Buildings that are able to receive messages are 2, 3, 5, 6, 7, and 8



Location of tower 3. Buildings that are able to receive messages are 12, 13, 15 and, 16

Task A2. THE GAME "DIVISIBILITY"

Two players, X and Y, play the following game:

- They are given a positive integer P and a set **A** consisting of N different nonnegative integers, $\mathbf{A} = \{a_1, a_2, \ldots, a_N\}$, such that every a_i is less than P.
- Players play with alternating turns. Each player on his turn deletes a number from the set A.
- If after exactly K turns, the sum of the numbers remaining in **A** is divisible by P – Player X wins. Otherwise – Player Y wins.

Write a program **`div`**, which determines who wins if both players play optimally.

Input

On the first line of the standard input is the positive integer T – the number of games in this test case.

After that, for each $i = 0, 1, \ldots, T - 1$:

- on the $(3i + 2)$-nd line are the numbers N, K and P, separated by spaces;
- on the $(3i + 3)$-rd line is either the symbol X, or the symbol Y, denoting which of the players goes first;
- on the $(3i + 4)$-th line are the space separated numbers $a_1, a_2, \ldots, a_N$..

Output

The standard output should be one line with T symbols (without separators), one symbol for each game in the test case. The i-th symbol should be X, if X wins in the i-th game, no matter how Y plays; otherwise, this symbol should be Y.

Constraints

$1 \le K \le N \le 5000$;
$P \le 10^{18}$;
$0 \le a_i < P$ for each $0 \le i < N$ and $a_i \ne a_j$ for each $0 \le i < j < N$.
In 20% of the test cases $N \le 25$.
In other 20% of the test cases P is a prime number.

Example

Input	Output
3 5 3 7 X 1 2 3 4 6 8 4 13 Y 5 10 6 11 2 8 9 3 6 1 12 X 1 4 5 7 9 11	XYX

Task A3. BIATHLON

Piggy decided to organize a biathlon race, where the competitors would compete in two disciplines. She invited *N* competitors, who have the following characteristics:

- Each competitor has velocities V_1 and V_2, for the two disciplines, respectively.
- The competitors have constant velocities (V_1 and V_2) throughout the respective tracks.
- The distance, which a competitor covers for time t_1 in the first discipline is $S_1 = V_1 t_1$, and the distance for the second discipline for time t_2 is $S_2 = V_2 t_2$.
- A competitor wins, if the sum of his times is *uniquely* the smallest among these of all competitors (i.e. strictly less than all the others).

As an organizer, Piggy can choose whatever distances she likes (non-negative real numbers S_1 and S_2) for each of the two disciplines. Now she is wondering which competitors are potential winners, that's to say, whether there exist S_1 and S_2 to ensure them victory.

Write a program **`biathlon`** to determine which competitors can win.

Input

On the first line of the standard input, *N* is given. On the next *N* lines are given two positive integers V_1 and V_2, separated by a space: the velocities of the *i*-th competitor (for *i*=0, 1, …, *N*-1).

Output

On one line of the standard output, print the indexes of the competitors who can win. The indexes should be in increasing order, separated by spaces. Indexing starts from 0. This line should contain the number -1, when there is no competitor who can win.

Subtasks

Subtask 1 (*20 points*): $2 \le N \le 100$, $1 \le V_1, V_2 \le 100$
Subtask 2 (*40 points*): $2 \le N \le 5000$, $1 \le V_1, V_2 \le 10\,000$
Subtask 3 (*40 points*): $2 \le N \le 100\,000$, $1 \le V_1, V_2 \le 10\,000$

Examples

Input	Output	Notes
4 1 4 2 2 4 1 3 3	0 2 3	All competitors, who can win, have the indexes: 0, 2 and 3. The one with index 0 wins for distances, for example, S_1=0 and S_2=10; competitor with index 2 wins for distances S_1=10 and S_2=0; the one with index 3 wins for some distances S_1=10 and S_2=10. Competitor with index 1 cannot win: he is always being defeated by competitor with index 3.
3 3 3 3 3 2 2	-1	Only competitors 0 and 1 can have minimal times, but neither of them unique. That's why the correct output is an empty line.