Codeforces Round #206 (Div. 1)

A. Vasya and Robot

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

Vasya has n items lying in a line. The items are consecutively numbered by numbers from 1 to n in such a way that the leftmost item has number 1, the rightmost item has number n. Each item has a weight, the i-th item weights w_i kilograms.

Vasya needs to collect all these items, however he won't do it by himself. He uses his brand new robot. The robot has two different arms — the left one and the right one. The robot can consecutively perform the following actions:

1. Take the leftmost item with the left hand and spend $w_i \cdot l$ energy units (w_i is a weight of the leftmost item, l is some parameter). If the previous action was the same (left-hand), then the robot spends extra Q_l energy units;
2. Take the rightmost item with the right hand and spend $w_j \cdot r$ energy units (w_j is a weight of the rightmost item, r is some parameter). If the previous action was the same (right-hand), then the robot spends extra Q_r energy units;

Naturally, Vasya wants to program the robot in a way that the robot spends as little energy as possible. He asked you to solve this problem. Your task is to find the minimum number of energy units robot spends to collect all items.

Input

The first line contains five integers n, l, r, Q_l, Q_r $(1 \le n \le 10^5; 1 \le l, r \le 100; 1 \le Q_l, Q_r \le 10^4)$.

The second line contains n integers $w_1, w_2, ..., w_n$ $(1 \le w_i \le 100)$.

Output

In the single line print a single number — the answer to the problem.

Sample test(s)

input
3 4 4 19 1 42 3 99
output
576

input
4 7 2 3 9 1 2 3 4
output
34

Note

Consider the first sample. As $l = r$, we can take an item in turns: first from the left side, then from the right one and last item from the left. In total the robot spends $4 \cdot 42 + 4 \cdot 99 + 4 \cdot 3 = 576$ energy units.

The second sample. The optimal solution is to take one item from the right, then one item from the left and two items from the right. In total the robot spends $(2 \cdot 4) + (7 \cdot 1) + (2 \cdot 3) + (2 \cdot 2 + 9) = 34$ energy units.

B. Game with Strings

time limit per test: 1 second
memory limit per test: 512 megabytes
input: standard input
output: standard output

Given an $n \times n$ table T consisting of lowercase English letters. We'll consider some string s *good* if the table contains a correct path corresponding to the given string. In other words, good strings are all strings we can obtain by moving from the left upper cell of the table only to the right and down. Here's the formal definition of correct paths:

Consider rows of the table are numbered from 1 to n from top to bottom, and columns of the table are numbered from 1 to n from left to the right. Cell (r, c) is a cell of table T on the r-th row and in the c-th column. This cell corresponds to letter $T_{r,c}$.

A path of length k is a sequence of table cells $[(r_1, c_1), (r_2, c_2), ..., (r_k, c_k)]$. The following paths are correct:

1. There is only one correct path of length 1, that is, consisting of a single cell: $[(1, 1)]$;
2. Let's assume that $[(r_1, c_1), ..., (r_m, c_m)]$ is a correct path of length m, then paths $[(r_1, c_1), ..., (r_m, c_m), (r_m + 1, c_m)]$ and $[(r_1, c_1), ..., (r_m, c_m), (r_m, c_m + 1)]$ are correct paths of length $m + 1$.

We should assume that a path $[(r_1, c_1), (r_2, c_2), ..., (r_k, c_k)]$ corresponds to a string of length k: $T_{r_1,c_1} + T_{r_2,c_2} + ... + T_{r_k,c_k}$.

Two players play the following game: initially they have an empty string. Then the players take turns to add a letter to the end of the string. After each move (adding a new letter) the resulting string must be good. The game ends after $2n$ - 1 turns. A player wins by the following scenario:

1. If the resulting string has strictly more letters "a" than letters "b", then the first player wins;
2. If the resulting string has strictly more letters "b" than letters "a", then the second player wins;
3. If the resulting string has the same number of letters "a" and "b", then the players end the game with a draw.

Your task is to determine the result of the game provided that both players played optimally well.

Input

The first line contains a single number n $(1 \le n \le 20)$.

Next n lines contain n lowercase English letters each — table T.

Output

In a single line print string "FIRST", if the first player wins, "SECOND", if the second player wins and "DRAW", if the game ends with a draw.

Sample test(s)

input
2 ab cd
output
DRAW

input
2 xa ay
output
FIRST

input
3 aab bcb bac
output
DRAW

Note

Consider the first sample:

Good strings are strings: a, ab, ac, abd, acd.

The first player moves first and adds letter a to the string, as there is only one good string of length 1. Then the second player can add b or c and the game will end with strings abd or acd, correspondingly. In the first case it will be a draw (the string has one a and one b), in the second case the first player wins. Naturally, in this case the second player prefers to choose letter b and end the game with a draw.

Consider the second sample:

Good strings are: `x`, `xa`, `xay`.

We can see that the game will end with string `xay` and the first player wins.

C. Vasya and Beautiful Arrays

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

Vasya's got a birthday coming up and his mom decided to give him an array of positive integers a of length n.

Vasya thinks that an array's beauty is the greatest common divisor of all its elements. His mom, of course, wants to give him as beautiful an array as possible (with largest possible beauty). Unfortunately, the shop has only one array a left. On the plus side, the seller said that he could decrease some numbers in the array (no more than by k for each number).

The seller can obtain array b from array a if the following conditions hold: $b_i > 0$; $0 \le a_i - b_i \le k$ for all $1 \le i \le n$.

Help mom find the maximum possible beauty of the array she will give to Vasya (that seller can obtain).

Input

The first line contains two integers n and k ($1 \le n \le 3 \cdot 10^5$; $1 \le k \le 10^6$). The second line contains n integers a_i ($1 \le a_i \le 10^6$) — array a.

Output

In the single line print a single number — the maximum possible beauty of the resulting array.

Sample test(s)

input
6 1 3 6 10 12 13 16
output
3

input
5 3 8 21 52 15 77
output
7

Note

In the first sample we can obtain the array:

3 6 9 12 12 15

In the second sample we can obtain the next array:

7 21 49 14 77

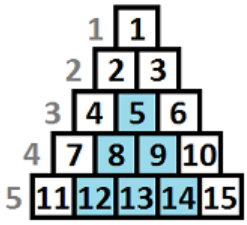
D. Transferring Pyramid

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Vasya and Petya are using an interesting data storing structure: a pyramid.

The pyramid consists of n rows, the i-th row contains i cells. Each row is shifted half a cell to the left relative to the previous row. The cells are numbered by integers from 1 to $\frac{n(n+1)}{2}$ as shown on the picture below.

An example of a pyramid at $n = 5$ is:



This data structure can perform operations of two types:

1. Change the value of a specific cell. It is described by three integers: "$t\ \ i\ \ v$", where $t = 1$ (the type of operation), i — the number of the cell to change and v the value to assign to the cell.
2. Change the value of some subpyramid. The picture shows a highlighted subpyramid with the top in cell 5. It is described by $s + 2$ numbers: "$t\ \ i\ \ v_1\ \ v_2\ \ \ldots\ \ v_s$", where $t = 2$, i — the number of the top cell of the pyramid, s — the size of the subpyramid (the number of cells it has), v_j — the value you should assign to the j-th cell of the subpyramid.

Formally: a subpyramid with top at the i-th cell of the k-th row (the 5-th cell is the second cell of the third row) will contain cells from rows from k to n, the $(k + p)$-th row contains cells from the i-th to the $(i + p)$-th $(0 \le p \le n - k)$.

Vasya and Petya had two identical pyramids. Vasya changed some cells in his pyramid and he now wants to send his changes to Petya. For that, he wants to find a sequence of operations at which Petya can repeat all Vasya's changes. Among all possible sequences, Vasya has to pick the minimum one (the one that contains the fewest numbers).

You have a pyramid of n rows with k changed cells. Find the sequence of operations which result in **each of the k changed cells being changed by at least one operation**. Among all the possible sequences pick **the one that contains the fewest numbers**.

Input

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

The next k lines contain the coordinates of the modified cells r_i and c_i $(1 \le c_i \le r_i \le n)$ — the row and the cell's number in the row. All cells are distinct.

Output

Print a single number showing how many numbers the final sequence has.

Sample test(s)

input
4 5 3 1 3 3 4 1 4 3 4 4
output
10

input
7 11 2 2 3 1 4 3 5 1 5 2 5 5 6 4 7 2 7 3 7 4 7 5
output
26

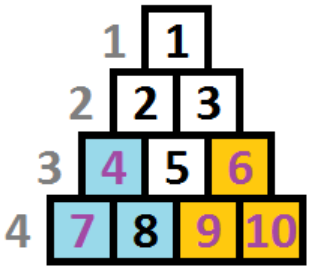
Note

One of the possible solutions of the first sample consists of two operations:

2 4 v_4 v_7 v_8

2 6 v_6 v_9 v_{10}

The picture shows the changed cells color-highlighted. The subpyramid used by the first operation is highlighted blue and the subpyramid used by the first operation is highlighted yellow:



E. Lucky Number Representation

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

We know that lucky digits are digits 4 and 7, however Vasya's got another favorite digit 0 and he assumes it also is lucky! Lucky numbers are such **non-negative** integers whose decimal record only contains lucky digits. For example, numbers $0, 47, 7074$ are lucky, but $1, 7377, 895, -7$ are not.

Vasya has t important positive integers he needs to remember. Vasya is quite superstitious and he wants to remember lucky numbers only, so he is asking you for each important number to represent it as a sum of exactly six lucky numbers (Vasya just can't remember more numbers). Then Vasya can just remember these six numbers and calculate the important number at any moment.

For each of t important integers represent it as the sum of six lucky numbers or state that this is impossible.

Input

The first line contains a single integer t $(1 \le t \le 5000)$.

Next t lines contain a single positive integer n_i $(1 \le n_i \le 10^{18})$ — the list of important numbers.

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

Output

Print t lines. The i-th line must contain the answer for the i-th important number: if the solution exists, the line must contain exactly six lucky numbers the sum of which equals n_i, if the solution doesn't exist the string must contain a single integer `-1`.

If there are multiple answers print any of them.

Sample test(s)

input
5 42 17 444 7 51

output
7 7 7 7 7 7 -1 400 0 40 0 4 0 7 0 0 0 0 0 47 4 0 0 0 0