Codeforces Round #802 (Div. 2)

A. Optimal Path

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

You are given a table a of size $n \times m$. We will consider the table rows numbered from top to bottom from 1 to n, and the columns numbered from left to right from 1 to m. We will denote a cell that is in the i-th row and in the j-th column as (i, j). In the cell (i, j) there is written a number $(i - 1) \cdot m + j$, that is $a_{ij} = (i - 1) \cdot m + j$.

A turtle initially stands in the cell $(1, 1)$ and it wants to come to the cell (n, m). From the cell (i, j) it can in one step go to one of the cells $(i + 1, j)$ or $(i, j + 1)$, if it exists. A path is a sequence of cells in which for every two adjacent in the sequence cells the following satisfies: the turtle can reach from the first cell to the second cell in one step. A cost of a path is the sum of numbers that are written in the cells of the path.

1	2	3
4	5	6

For example, with $n = 2$ and $m = 3$ the table will look as shown above. The turtle can take the following path: $(1, 1) \to (1, 2) \to (1, 3) \to (2, 3)$. The cost of such way is equal to $a_{11} + a_{12} + a_{13} + a_{23} = 12$. On the other hand, the paths $(1, 1) \to (1, 2) \to (2, 2) \to (2, 1)$ and $(1, 1) \to (1, 3)$ are incorrect, because in the first path the turtle can't make a step $(2, 2) \to (2, 1)$, and in the second path it can't make a step $(1, 1) \to (1, 3)$.

You are asked to tell the turtle a minimal possible cost of a path from the cell $(1, 1)$ to the cell (n, m). Please note that the cells $(1, 1)$ and (n, m) are a part of the way.

Input

The first line contains a single integer t ($1 \le t \le 1000$) — the number of test cases. The description of test cases follows.

A single line of each test case contains two integers n and m ($1 \le n, m \le 10^4$) — the number of rows and columns of the table a respectively.

Output

For each test case output a single integer — a minimal possible cost of a path from the cell $(1, 1)$ to the cell (n, m).

Example

input
7 1 1 2 3 3 2 7 1 1 10 5 5 10000 10000
output
1 12 13 28 55 85 500099995000

Note

In the first test case the only possible path consists of a single cell $(1, 1)$.

The path with the minimal cost in the second test case is shown in the statement.

In the fourth and the fifth test cases there is only one path from $(1, 1)$ to (n, m). Both paths visit every cell in the table.

B. Palindromic Numbers

time limit per test: 2 seconds

memory limit per test: 256 megabytes
input: standard input
output: standard output

During a daily walk Alina noticed a long number written on the ground. Now Alina wants to find some positive number of same length without leading zeroes, such that the sum of these two numbers is a palindrome.

Recall that a number is called a palindrome, if it reads the same right to left and left to right. For example, numbers $121, 66, 98989$ are palindromes, and $103, 239, 1241$ are not palindromes.

Alina understands that a valid number always exist. Help her find one!

Input
The first line of input data contains an integer t ($1 \le t \le 100$) — the number of test cases. Next, descriptions of t test cases follow.

The first line of each test case contains a single integer n ($2 \le n \le 100\,000$) — the length of the number that is written on the ground.

The second line of contains the positive n-digit integer without leading zeroes — the number itself.

It is guaranteed that the sum of the values n over all test cases does not exceed $100\,000$.

Output
For each of t test cases print an answer — a positive n-digit integer without leading zeros, such that the sum of the input integer and this number is a palindrome.

We can show that at least one number satisfying the constraints exists. If there are multiple solutions, you can output any of them.

Example

input
3 2 99 4 1023 3 385
output
32 8646 604

Note
In the first test case $99 + 32 = 131$ is a palindrome. Note that another answer is 12, because $99 + 12 = 111$ is also a palindrome.

In the second test case $1023 + 8646 = 9669$.

In the third test case $385 + 604 = 989$.

C. Helping the Nature

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

Little Leon lives in the forest. He has recently noticed that some trees near his favourite path are withering, while the other ones are overhydrated so he decided to learn how to control the level of the soil moisture to save the trees.

There are n trees growing near the path, the current levels of moisture of each tree are denoted by the array $a_1, a_2, \ldots, a_n$. Leon has learned three abilities which will help him to dry and water the soil.

- Choose a position i and decrease the level of moisture of the trees $1, 2, \ldots, i$ by 1.
- Choose a position i and decrease the level of moisture of the trees $i, i + 1, \ldots, n$ by 1.
- Increase the level of moisture of all trees by 1.

Leon wants to know the minimum number of actions he needs to perform to make the moisture of each tree equal to 0.

Input
The first line contains a single integer t ($1 \le t \le 2 \cdot 10^4$) — the number of test cases. The description of t test cases follows.

The first line of each test case contains a single integer n ($1 \le n \le 200\,000$).

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($-10^9 \le a_i \le 10^9$) — the initial levels of trees moisture.

It is guaranteed that the sum of n over all test cases doesn't exceed $200\,000$.

Output

For each test case output a single integer — the minimum number of actions. It can be shown that the answer exists.

Example

input
4 3 -2 -2 -2 3 10 4 7 4 4 -4 4 -4 5 1 -2 3 -4 5
output
2 13 36 33

Note

In the first test case it's enough to apply the operation of adding 1 to the whole array 2 times.

In the second test case you can apply the operation of decreasing 4 times on the prefix of length 3 and get an array $6, 0, 3$.

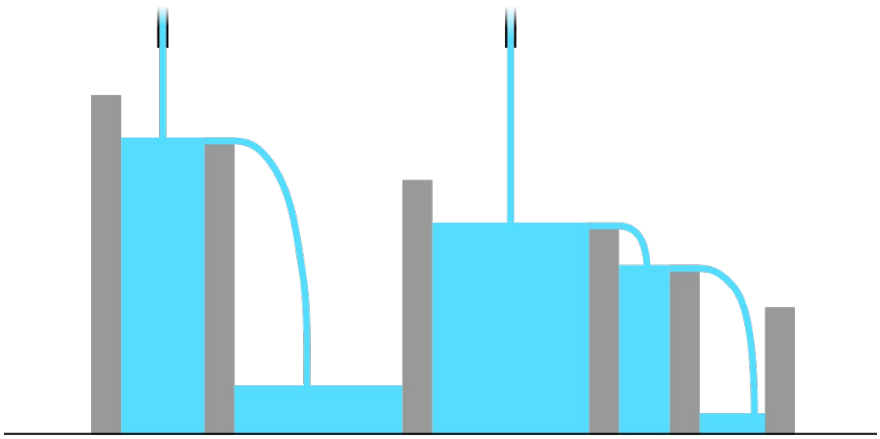
After that apply the operation of decreasing 6 times on the prefix of length 1 and 3 times on the suffix of length 1. In total, the number of actions will be $4 + 6 + 3 = 13$. It can be shown that it's impossible to perform less actions to get the required array, so the answer is 13.

D. River Locks

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

Recently in Divanovo, a huge river locks system was built. There are now n locks, the i-th of them has the volume of v_i liters, so that it can contain any amount of water between 0 and v_i liters. Each lock has a pipe attached to it. When the pipe is open, 1 liter of water enters the lock every second.

The locks system is built in a way to immediately transfer all water exceeding the volume of the lock i to the lock $i + 1$. If the lock $i + 1$ is also full, water will be transferred further. Water exceeding the volume of the last lock pours out to the river.



The picture illustrates 5 locks with two open pipes at locks 1 and 3. Because locks 1, 3, and 4 are already filled, effectively the water goes to locks 2 and 5.

Note that the volume of the i-th lock may be greater than the volume of the $i + 1$-th lock.

To make all locks work, you need to completely fill each one of them. The mayor of Divanovo is interested in q independent queries. For each query, suppose that initially all locks are empty and all pipes are closed. Then, some pipes are opened simultaneously. For the j-th query the mayor asks you to calculate the minimum number of pipes to open so that all locks are filled no later than after t_j seconds.

Please help the mayor to solve this tricky problem and answer his queries.

Input

The first lines contains one integer n ($1 \le n \le 200\,000$) — the number of locks.

The second lines contains n integers $v_1, v_2, \ldots, v_n$ ($1 \le v_i \le 10^9$)) — volumes of the locks.

The third line contains one integer q ($1 \le q \le 200\,000$) — the number of queries.

Each of the next q lines contains one integer t_j ($1 \le t_j \le 10^9$) — the number of seconds you have to fill all the locks in the query j.

Output

Print q integers. The j-th of them should be equal to the minimum number of pipes to turn on so that after t_j seconds all of the locks are filled. If it is impossible to fill all of the locks in given time, print -1.

Examples

input
5 4 1 5 4 1 6 1 6 2 3 4 5
output
-1 3 -1 -1 4 3

input
5 4 4 4 4 4 6 1 3 6 5 2 4
output
-1 -1 4 4 -1 5

Note

There are 6 queries in the first example test.

In the queries $1, 3, 4$ the answer is -1. We need to wait 4 seconds to fill the first lock even if we open all the pipes.

In the sixth query we can open pipes in locks 1, 3, and 4. After 4 seconds the locks 1 and 4 are full. In the following 1 second 1 liter of water is transferred to the locks 2 and 5. The lock 3 is filled by its own pipe.

Similarly, in the second query one can open pipes in locks 1, 3, and 4.

In the fifth query one can open pipes $1, 2, 3, 4$.

E. Serega the Pirate

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

Little pirate Serega robbed a ship with puzzles of different kinds. Among all kinds, he liked only one, the hardest.

A puzzle is a table of n rows and m columns, whose cells contain each number from 1 to $n \cdot m$ exactly once.

To solve a puzzle, you have to find a sequence of cells in the table, such that any two consecutive cells are adjacent by the side in the table. The sequence can have arbitrary length and should visit each cell one or more times. For a cell containing the number i, denote the position of the first occurrence of this cell in the sequence as t_i. The sequence solves the puzzle, if $t_1 < t_2 < \cdots < t_{nm}$. In other words, the cell with number x should be first visited before the cell with number $x + 1$ for each x.

Let's call a puzzle solvable, if there exists at least one suitable sequence.

In one move Serega can choose two arbitrary cells in the table (not necessarily adjacent by the side) and swap their numbers. He would like to know the minimum number of moves to make his puzzle solvable, but he is too impatient. Thus, please tell if the minimum number of moves is 0, 1, or at least 2. In the case, where 1 move is required, please also find the number of suitable cell pairs to swap.

Input

In the first line there are two whole positive numbers n, m ($1 \le n \cdot m \le 400\,000$) — table dimensions.

In the next n lines there are m integer numbers $a_{i1}, a_{i2}, \ldots, a_{im}$ ($1 \le a_{ij} \le nm$).

It is guaranteed that every number from 1 to nm occurs exactly once in the table.

Output
Let a be the minimum number of moves to make the puzzle solvable.

If $a = 0$, print 0.

If $a = 1$, print 1 and the number of valid swaps.

If $a \geq 2$, print 2.

Examples

input
3 3 2 1 3 6 7 4 9 8 5
output
0

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

input
1 6 1 6 5 4 3 2
output
2

Note
In the first example the sequence $(1, 2), (1, 1), (1, 2), (1, 3), (2, 3), (3, 3), (2, 3), (1, 3), (1, 2), (1, 1), (2, 1), (2, 2), (3, 2), (3, 1)$ solves the puzzle, so the answer is 0.

The puzzle in the second example can't be solved, but it's solvable after any of three swaps of cells with values $(1, 5), (1, 6), (2, 6)$.

The puzzle from the third example requires at least two swaps, so the answer is 2.

F. Puzzle

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

Pupils Alice and Ibragim are best friends. It's Ibragim's birthday soon, so Alice decided to gift him a new puzzle. The puzzle can be represented as a matrix with 2 rows and n columns, every element of which is either 0 or 1. In one move you can swap two values in neighboring cells.

More formally, let's number rows 1 to 2 from top to bottom, and columns 1 to n from left to right. Also, let's denote a cell in row x and column y as (x, y). We consider cells (x_1, y_1) and (x_2, y_2) neighboring if $|x_1 - x_2| + |y_1 - y_2| = 1$.

Alice doesn't like the way in which the cells are currently arranged, so she came up with her own arrangement, with which she wants to gift the puzzle to Ibragim. Since you are her smartest friend, she asked you to help her find the minimal possible number of operations in which she can get the desired arrangement. Find this number, or determine that it's not possible to get the new arrangement.

Input
The first line contains an integer n ($1 \leq n \leq 200\,000$) — the number of columns in the puzzle.

Following two lines describe the current arrangement on the puzzle. Each line contains n integers, every one of which is either 0 or 1.

The last two lines describe Alice's desired arrangement in the same format.

Output
If it is possible to get the desired arrangement, print the minimal possible number of steps, otherwise print -1.

Examples

input
5

```
0 1 0 1 0
1 1 0 0 1
1 0 1 0 1
0 0 1 1 0
```

output

```
5
```

input

```
3
1 0 0
0 0 0
0 0 0
0 0 0
```

output

```
-1
```

Note

In the first example the following sequence of swaps will suffice:

- $(2, 1), (1, 1)$,
- $(1, 2), (1, 3)$,
- $(2, 2), (2, 3)$,
- $(1, 4), (1, 5)$,
- $(2, 5), (2, 4)$.

It can be shown that 5 is the minimal possible answer in this case.

In the second example no matter what swaps you do, you won't get the desired arrangement, so the answer is -1.