A. AquaMoon and Two Arrays

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

AquaMoon and Cirno are playing an interesting game with arrays. Cirno has prepared two arrays a and b, both consist of n non-negative integers. AquaMoon can perform the following operation an arbitrary number of times (possibly zero):

- She chooses two indices i and j ($1 \leq i, j \leq n$), then decreases the i-th element of array a by 1, and increases the j-th element of array a by 1. The resulting values at i-th and j-th index of array a are $a_i - 1$ and $a_j + 1$, respectively. Each element of array a **must be non-negative after each operation**. If $i = j$ this operation doesn't change the array a.

AquaMoon wants to make some operations to make arrays a and b equal. Two arrays a and b are considered equal if and only if $a_i = b_i$ for all $1 \leq i \leq n$.

Help AquaMoon to find a sequence of operations that will solve her problem or find, that it is impossible to make arrays a and b equal.

Please note, that you **don't have to minimize** the number of operations.

Input

The input consists of multiple test cases. The first line contains a single integer t ($1 \leq t \leq 100$) — the number of test cases.

The first line of each test case contains a single integer n ($1 \leq n \leq 100$).

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($0 \leq a_i \leq 100$). **The sum of all a_i does not exceed 100.**

The third line of each test case contains n integers $b_1, b_2, \ldots, b_n$ ($0 \leq b_i \leq 100$). **The sum of all b_i does not exceed 100.**

Output

For each test case print "-1" on the only line if it is impossible to make two arrays equal with some sequence of operations.

Otherwise, print an integer m ($0 \leq m \leq 100$) in the first line — the number of operations. Then print m lines, each line consists of two integers i and j — the indices you choose for the operation.

It can be proven that if it is possible to make two arrays equal with some sequence of operations, there exists a sequence with $m \leq 100$.

If there are multiple possible solutions, you can print any.

Example

input
4 4 1 2 3 4 3 1 2 4 2 1 3 2 1 1 0 0 5 4 3 2 1 0 0 1 2 3 4
output
2 2 1 3 1 -1 0 6 1 4 1 4 1 5 1 5 2 5 2 5

Note

In the first example, we do the following operations:

- $i = 2$, $j = 1$: $[1, 2, 3, 4] \to [2, 1, 3, 4]$;
- $i = 3$, $j = 1$: $[2, 1, 3, 4] \to [3, 1, 2, 4]$;

In the second example, it's impossible to make two arrays equal.

B. AquaMoon and Stolen String

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

AquaMoon had n strings of length m each. n is an **odd** number.

When AquaMoon was gone, Cirno tried to pair these n strings together. After making $\frac{n-1}{2}$ pairs, she found out that there was exactly one string without the pair!

In her rage, she disrupted each pair of strings. For each pair, she selected some positions (at least 1 and at most m) and swapped the letters in the two strings of this pair at the selected positions.

For example, if $m = 6$ and two strings "abcdef" and "xyzklm" are in one pair and Cirno selected positions 2, 3 and 6 she will swap 'b' with 'y', 'c' with 'z' and 'f' with 'm'. The resulting strings will be "ayzdem" and "xbcklf".

Cirno then stole away the string without pair and shuffled all remaining strings in arbitrary order.

AquaMoon found the remaining $n - 1$ strings in complete disarray. Also, she remembers the initial n strings. She wants to know which string was stolen, but she is not good at programming. Can you help her?

Input

This problem is made as interactive. It means, that your solution will read the input, given by the interactor. But the interactor will give you the full input at the beginning and after that, you should print the answer. So you should solve the problem, like as you solve the usual, non-interactive problem because you won't have any interaction process. The only thing you should not forget is to flush the output buffer, after printing the answer. Otherwise, you can get an "Idleness limit exceeded" verdict. Refer to the interactive problems guide for the detailed information about flushing the output buffer.

The input consists of multiple test cases. The first line contains a single integer t ($1 \le t \le 100$) — the number of test cases.

The first line of each test case contains two integers n, m ($1 \le n \le 10^5$, $1 \le m \le 10^5$) — the number of strings and the length of each string, respectively.

The next n lines each contain a string with length m, describing the original n strings. All string consists of lowercase Latin letters.

The next $n - 1$ lines each contain a string with length m, describing the strings after Cirno exchanged and reordered them.

It is guaranteed that n is odd and that the sum of $n \cdot m$ over all test cases does not exceed 10^5.

Hack format:

The first line should contain a single integer t. After that t test cases should follow in the following format:

The first line should contain two integers n and m.

The following n lines should contain n strings of length m, describing the original strings.

The following $\frac{n-1}{2}$ lines should describe the pairs. They should contain, in the following order: the index of the first string i ($1 \le i \le n$), the index of the second string j ($1 \le j \le n$, $i \ne j$), the number of exchanged positions k ($1 \le k \le m$), and the list of k positions that are exchanged (k distinct indices from 1 to m in any order).

The final line should contain a permutation of integers from 1 to n, describing the way the strings should be reordered. The strings will be placed in the order indices placed in this permutation, the stolen string index will be ignored.

Output

For each test case print a single line with the stolen string.

Example

input
3 3 5 aaaaa bbbbb ccccc aaaaa bbbbb 3 4 aaaa bbbb cccc aabb bbaa

5 6
abcdef
uuuuuu
kekeke
ekekek
xyzklm
xbcklf
eueueu
ayzdem
ukukuk

output

ccccc
cccc
kekeke

Note

In the first test case, "aaaaa" and "bbbbb" exchanged all positions, and "ccccc" is the stolen string.

In the second test case, "aaaa" and "bbbb" exchanged two first positions, and "cccc" is the stolen string.

This is the first test in the hack format:

3
3 5
aaaaa
bbbbb
ccccc
1 2 5 1 2 3 4 5
2 1 3
3 4
aaaa
bbbb
cccc
1 2 2 1 2
2 1 3
5 6
abcdef
uuuuuu
kekeke
ekekek
xyzklm
1 5 3 2 3 6
2 4 3 2 4 6
5 4 1 2 3

C. AquaMoon and Strange Sort

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

AquaMoon has n friends. They stand in a row from left to right, and the i-th friend from the left wears a T-shirt with a number a_i written on it. Each friend has a direction (left or right). In the beginning, the direction of each friend is **right**.

AquaMoon can make some operations on friends. On each operation, AquaMoon can choose two **adjacent** friends and swap their positions. After each operation, the direction of both chosen friends will also be flipped: left to right and vice versa.

AquaMoon hopes that after some operations, the numbers written on the T-shirt of n friends in the row, read from left to right, become **non-decreasing**. Also she wants, that all friends will have a direction of **right** at the end. Please find if it is possible.

Input

The input consists of multiple test cases. The first line contains a single integer t ($1 \leq t \leq 50$) — the number of test cases.

The first line of each test case contains a single integer n ($1 \leq n \leq 10^5$) — the number of Aquamoon's friends.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 10^5$) — the numbers, written on the T-shirts.

It is guaranteed that the sum of n for all test cases does not exceed 10^5.

Output

For each test case, if there exists a possible sequence of operations, print "YES" (without quotes); otherwise, print "NO" (without quotes).

You can print each letter in any case (upper or lower).

Example

input
3 4 4 3 2 5 4 3 3 2 2 5 1 2 3 5 4
output
YES YES NO

Note
The possible list of operations in the first test case:

1. Swap a_1 and a_2. The resulting sequence is $3, 4, 2, 5$. The directions are: left, left, right, right.
2. Swap a_2 and a_3. The resulting sequence is $3, 2, 4, 5$. The directions are: left, left, right, right.
3. Swap a_1 and a_2. The resulting sequence is $2, 3, 4, 5$. The directions are: right, right, right, right.

D. AquaMoon and Chess

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

Cirno gave AquaMoon a chessboard of size $1 \times n$. Its cells are numbered with integers from 1 to n from left to right. In the beginning, some of the cells are occupied with at most one pawn, and other cells are unoccupied.

In each operation, AquaMoon can choose a cell i with a pawn, and do **either** of the following (if possible):

- Move pawn from it to the $(i+2)$-th cell, if $i+2 \leq n$ and the $(i+1)$-th cell is occupied and the $(i+2)$-th cell is unoccupied.
- Move pawn from it to the $(i-2)$-th cell, if $i-2 \geq 1$ and the $(i-1)$-th cell is occupied and the $(i-2)$-th cell is unoccupied.

You are given an initial state of the chessboard. AquaMoon wants to count the number of states reachable from the initial state with some sequence of operations. But she is not good at programming. Can you help her? As the answer can be large find it modulo $998\,244\,353$.

Input
The input consists of multiple test cases. The first line contains a single integer t ($1 \leq t \leq 10\,000$) — the number of test cases.

The first line contains a single integer n ($1 \leq n \leq 10^5$) — the size of the chessboard.

The second line contains a string of n characters, consists of characters "0" and "1". If the i-th character is "1", the i-th cell is initially occupied; otherwise, the i-th cell is initially unoccupied.

It is guaranteed that the sum of n over all test cases does not exceed 10^5.

Output
For each test case, print the number of states that reachable from the initial state with some sequence of operations modulo $998\,244\,353$.

Example

input
6 4 0110 6 011011 5 01010 20 10001111110110111000 20 00110110100110111101 20 11101111011000100010
output
3 6 1 1287 1287 715

Note
In the first test case the strings "1100", "0110" and "0011" are reachable from the initial state with some sequence of operations.

E. AquaMoon and Permutations

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

Cirno has prepared n arrays of length n each. Each array is a permutation of n integers from 1 to n. These arrays are special: for all $1 \leq i \leq n$, if we take the i-th element of each array and form another array of length n with these elements, the resultant array is also a permutation of n integers from 1 to n. In the other words, if you put these n arrays under each other to form a matrix with n rows and n columns, this matrix is a Latin square.

Afterwards, Cirno added additional n arrays, each array is a permutation of n integers from 1 to n. For all $1 \leq i \leq n$, there exists **at least one** position $1 \leq k \leq n$, such that for the i-th array and the $(n+i)$-th array, the k-th element of both arrays is the same. Notice that the arrays indexed from $n+1$ to $2n$ **don't have to** form a Latin square.

Also, Cirno made sure that for all $2n$ arrays, no two arrays are completely equal, i. e. for all pair of indices $1 \leq i < j \leq 2n$, there exists **at least one** position $1 \leq k \leq n$, such that the k-th elements of the i-th and j-th array are **different**.

Finally, Cirno arbitrarily changed the order of $2n$ arrays.

AquaMoon calls a subset of all $2n$ arrays of size n **good** if these arrays from a Latin square.

AquaMoon wants to know how many good subsets exist. Because this number may be particularly large, find it modulo $998\,244\,353$. Also, she wants to find any good subset. Can you help her?

Input
The input consists of multiple test cases. The first line contains a single integer t ($1 \leq t \leq 100$) — the number of test cases.

The first line of each test case contains a single integer n ($5 \leq n \leq 500$).

Then $2n$ lines followed. The i-th of these lines contains n integers, representing the i-th array.

It is guaranteed, that the sum of n over all test cases does not exceed 500.

Output
For each test case print two lines.

In the first line, print the number of good subsets by modulo $998\,244\,353$.

In the second line, print n indices from 1 to $2n$ — indices of the n arrays that form a good subset (you can print them in any order). If there are several possible answers — print any of them.

Example

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

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

output

```
1
1 2 3 4 5 6 7
2
1 3 5 6 10
4
1 3 6 7 8 9
```

Note

In the first test case, the number of good subsets is 1. The only such subset is the set of arrays with indices 1, 2, 3, 4, 5, 6, 7.

In the second test case, the number of good subsets is 2. They are 1, 3, 5, 6, 10 or 2, 4, 7, 8, 9.

F. AquaMoon and Wrong Coordinate

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

Cirno gives AquaMoon a problem. There are m people numbered from 0 to $m - 1$. They are standing on a coordinate axis in points with positive integer coordinates. They are facing right (i.e. in the direction of the coordinate increase). At this moment everyone will start running with the constant speed in the direction of coordinate increasing. The initial coordinate of the i-th person on the line is x_i, and the speed of the i-th person is v_i. So the coordinate of the i-th person at the moment t will be $x_i + t \cdot v_i$.

Cirno captured the coordinates of m people in k consecutive integer moments from 0 to $k - 1$. In every moment, the coordinates of m people were recorded in **arbitrary order**.

To make the problem more funny, Cirno modified one coordinate at the moment y ($0 < y < k - 1$) to a **different** integer.

AquaMoon wants to find the moment y and the original coordinate p before the modification. Actually, she is not a programmer at all. So she wasn't able to solve it. Can you help her?

Input

This problem is made as interactive. It means, that your solution will read the input, given by the interactor. But the interactor will give you the full input at the beginning and after that, you should print the answer. So you should solve the problem, like as you solve the usual, non-interactive problem because you won't have any interaction process. The only thing you should not forget is to flush the output buffer, after printing the answer. Otherwise, you can get an "Idleness limit exceeded" verdict. Refer to the interactive problems guide for the detailed information about flushing the output buffer.

The first line contains two integers m and k ($5 \le m \le 1000$, $7 \le k \le 1000$) — the number of people and the number of recorded moments.

The next k lines contain captured positions. i-th of these lines contains m integers between 1 and 10^6 (inclusive), representing positions captured by Cirno at the moment $i - 1$.

The input is guaranteed to be valid (i.e. only one integer was modified to a different value according to the problem statement). Also, it is guaranteed, that $1 \le v_i \le 1000$ for all $1 \le i \le m$.

Hack format:

The first line should contain two integers m and k ($5 \le m \le 1000$, $7 \le k \le 1000$) — the number of people and the number of moments.

In the second line, there should be m integers $x_0, x_1, \ldots, x_{m-1}$ ($1 \le x_i \le 10^6$), where x_i is the initial coordinate of the i-th person.

In the third line, there should be m integers $v_0, v_1, \ldots, v_{m-1}$ ($1 \le v_i \le 1000$), where v_i is the speed of the i-th person. It should be true that $x_i + (k - 1)v_i \le 10^6$ for each $0 \le i < m$.

In the next k lines, each line should contain m integers. i-th line should contain m distinct integers $p_0, p_1, \ldots, p_{m-1}$ ($0 \le p_j < m$). The meaning of these numbers: j-th integer in the input in the i-th moment is the coordinate of the p_j-th person.

In the last line, there should be three integers y, i, c. Cirno modified the coordinate of the i-th person at the moment y to c ($1 \le y \le k - 2$, $0 \le i \le m - 1$, $1 \le c \le 10^6$, $c \ne x_i + y \cdot v_i$).

Output

Print a single line with two integers y, p — the moment that contains the modified coordinate and the original coordinate.

Example

input

```
5 7
6 9 9 6 9
10 7 10 8 10
```

11 11 11 10 8
12 12 12 12 9
14 13 12 10 13
11 14 16 14 14
12 15 18 15 15

output

4 13

Note

In the first test the initial coordinates of people are $9, 6, 6, 9, 9$ and their speeds are $1, 2, 1, 1, 1$. So, it's easy to see, that at the moment 4 one coordinate was modified from 13 to 12.

This is the first test in the hack format:

```
5 7
9 6 6 9 9
1 2 1 1 1
2 3 4 1 0
0 2 3 1 4
4 3 0 1 2
1 3 4 0 2
1 4 0 2 3
2 4 1 3 0
2 4 1 3 0
4 0 12
```