Codeforces Round #369 (Div. 2)

A. Bus to Udayland

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

ZS the Coder and Chris the Baboon are travelling to Udayland! To get there, they have to get on the special IOI bus. The IOI bus has n rows of seats. There are 4 seats in each row, and the seats are separated into pairs by a walkway. When ZS and Chris came, some places in the bus was already occupied.

ZS and Chris are good friends. They insist to get **a pair of neighbouring empty seats**. Two seats are considered neighbouring if they are in the same row and in the same pair. Given the configuration of the bus, can you help ZS and Chris determine where they should sit?

Input

The first line of the input contains a single integer n $(1 \le n \le 1000)$ — the number of rows of seats in the bus.

Then, n lines follow. Each line contains exactly 5 characters, the first two of them denote the first pair of seats in the row, the third character denotes the walkway (it always equals '|') and the last two of them denote the second pair of seats in the row.

Each character, except the walkway, equals to 'O' or to 'X'. 'O' denotes an empty seat, 'X' denotes an occupied seat. See the sample cases for more details.

Output

If it is possible for Chris and ZS to sit at neighbouring empty seats, print "YES" (without quotes) in the first line. In the next n lines print the bus configuration, where the characters in the pair of seats for Chris and ZS is changed with characters '+'. Thus the configuration should differ from the input one by exactly two charaters (they should be equal to 'O' in the input and to '+' in the output).

If there is no pair of seats for Chris and ZS, print "NO" (without quotes) in a single line.

If there are multiple solutions, you may print any of them.

Examples

input
6 OO|OX XO|XX OX|OO XX|OX OO|OO OO|XX
output
YES ++|OX XO|XX OX|OO XX|OX OO|OO OO|XX

input
4 XO|OX XO|XX OX|OX XX|OX
output
NO

input
5 XX|XX XX|XX XO|OX XO|OO OX|XO
output

```
YES
XX|XX
XX|XX
X0|0X
X0|++
0X|X0
```

Note

Note that the following is an incorrect configuration for the first sample case because the seats must be in the same pair.

O+|+X

XO|XX

OX|OO

XX|OX

OO|OO

OO|XX

B. Chris and Magic Square

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

ZS the Coder and Chris the Baboon arrived at the entrance of Udayland. There is a $n \times n$ magic grid on the entrance which is filled with integers. Chris noticed that exactly one of the cells in the grid is empty, and to enter Udayland, they need to fill a **positive integer** into the empty cell.

Chris tried filling in random numbers but it didn't work. ZS the Coder realizes that they need to fill in a positive integer such that the numbers in the grid form *a magic square*. This means that he has to fill in a positive integer so that the sum of the numbers in each row of the grid (), each column of the grid (), and the two long diagonals of the grid (the main diagonal — and the secondary diagonal —) are equal.

Chris doesn't know what number to fill in. Can you help Chris find the correct positive integer to fill in or determine that it is impossible?

Input

The first line of the input contains a single integer n ($1 \le n \le 500$) — the number of rows and columns of the magic grid.

n lines follow, each of them contains n integers. The j-th number in the i-th of them denotes $a_{i,j}$ ($1 \le a_{i,j} \le 10^9$ or $a_{i,j} = 0$), the number in the i-th row and j-th column of the magic grid. If the corresponding cell is empty, $a_{i,j}$ will be equal to 0. Otherwise, $a_{i,j}$ is **positive**.

It is guaranteed that there is exactly one pair of integers i, j ($1 \le i, j \le n$) such that $a_{i,j} = 0$.

Output

Output a single integer, the positive integer x ($1 \le x \le 10^{18}$) that should be filled in the empty cell so that the whole grid becomes a magic square. If such positive integer x does not exist, output -1 instead.

If there are multiple solutions, you may print any of them.

Examples

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

input
4 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1
output
1

input
4 1 1 1 1 1 1 0 1 1 1 2 1 1 1 1 1
output
-1

Note

In the first sample case, we can fill in 9 into the empty cell to make the resulting grid a magic square. Indeed,

The sum of numbers in each row is:

$4 + 9 + 2 = 3 + 5 + 7 = 8 + 1 + 6 = 15$.

The sum of numbers in each column is:

$4 + 3 + 8 = 9 + 5 + 1 = 2 + 7 + 6 = 15$.

The sum of numbers in the two diagonals is:

$4 + 5 + 6 = 2 + 5 + 8 = 15$.

In the third sample case, it is impossible to fill a number in the empty square such that the resulting grid is a magic square.

C. Coloring Trees

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

ZS the Coder and Chris the Baboon has arrived at Udayland! They walked in the park where n trees grow. They decided to be naughty and color the trees in the park. The trees are numbered with integers from 1 to n from left to right.

Initially, tree i has color c_i. ZS the Coder and Chris the Baboon recognizes only m different colors, so $0 \le c_i \le m$, where $c_i = 0$ means that tree i is *uncolored*.

ZS the Coder and Chris the Baboon decides to color only the uncolored trees, i.e. the trees with $c_i = 0$. They can color each of them them in any of the m colors from 1 to m. Coloring the i-th tree with color j requires exactly $p_{i,j}$ litres of paint.

The two friends define the *beauty* of a coloring of the trees as the **minimum** number of contiguous groups (each group contains some subsegment of trees) you can split all the n trees into so that each group contains trees of the same color. For example, if the colors of the trees from left to right are $2, 1, 1, 1, 3, 2, 2, 3, 1, 3$, the beauty of the coloring is 7, since we can partition the trees into 7 contiguous groups of the same color : $\{2\}, \{1, 1, 1\}, \{3\}, \{2, 2\}, \{3\}, \{1\}, \{3\}$.

ZS the Coder and Chris the Baboon wants to color all uncolored trees so that the beauty of the coloring is **exactly** k. They need your help to determine the minimum amount of paint (in litres) needed to finish the job.

Please note that the friends can't color the trees that are already colored.

Input

The first line contains three integers, n, m and k ($1 \le k \le n \le 100$, $1 \le m \le 100$) — the number of trees, number of colors and beauty of the resulting coloring respectively.

The second line contains n integers $c_1, c_2, \ldots, c_n$ ($0 \le c_i \le m$), the initial colors of the trees. c_i equals to 0 if the tree number i is uncolored, otherwise the i-th tree has color c_i.

Then n lines follow. Each of them contains m integers. The j-th number on the i-th of them line denotes $p_{i,j}$ ($1 \le p_{i,j} \le 10^9$) — the amount of litres the friends need to color i-th tree with color j. $p_{i,j}$'s are specified even for the initially colored trees, but such trees still can't be colored.

Output

Print a single integer, the minimum amount of paint needed to color the trees. If there are no valid tree colorings of beauty k, print -1.

Examples

input

```
3 2 2
0 0 0
1 2
3 4
5 6
```

output

```
10
```

input

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

output

```
-1
```

input

```
3 2 2
2 0 0
1 3
2 4
3 5
```

output

```
5
```

input

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

output
0

Note

In the first sample case, coloring the trees with colors $2, 1, 1$ minimizes the amount of paint used, which equals to $2 + 3 + 5 = 10$. Note that $1, 1, 1$ would not be valid because the beauty of such coloring equals to 1 ($\{1, 1, 1\}$ is a way to group the trees into a single group of the same color).

In the second sample case, all the trees are colored, but the beauty of the coloring is 3, so there is no valid coloring, and the answer is -1.

In the last sample case, all the trees are colored and the beauty of the coloring matches k, so no paint is used and the answer is 0.

D. Directed Roads

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

ZS the Coder and Chris the Baboon has explored Udayland for quite some time. They realize that it consists of n towns numbered from 1 to n.

There are n directed roads in the Udayland. i-th of them goes from town i to some other town a_i ($a_i \neq i$). ZS the Coder can flip the direction of any road in Udayland, i.e. if it goes from town A to town B before the flip, it will go from town B to town A after.

ZS the Coder considers the roads in the Udayland *confusing*, if there is a sequence of distinct towns $A_1, A_2, ..., A_k$ ($k > 1$) such that for every $1 \leq i < k$ there is a road from town A_i to town A_{i+1} and another road from town A_k to town A_1. In other words, the roads are confusing if **some of them** form a directed cycle of some towns.

Now ZS the Coder wonders how many sets of roads (there are 2^n variants) in initial configuration can he choose to flip such that after flipping each road in the set exactly once, the resulting network will **not** be confusing.

Note that it is allowed that after the flipping there are more than one directed road from some town and possibly some towns with no roads leading out of it, or multiple roads between any pair of cities.

Input

The first line of the input contains single integer n ($2 \leq n \leq 2 \cdot 10^5$) — the number of towns in Udayland.

The next line contains n integers $a_1, a_2, ..., a_n$ ($1 \leq a_i \leq n, a_i \neq i$), a_i denotes a road going from town i to town a_i.

Output

Print a single integer — the number of ways to flip some set of the roads so that the resulting whole set of all roads is not confusing. Since this number may be too large, print the answer modulo $10^9 + 7$.

Examples

input
3 2 3 1
output
6

input
4 2 1 1 1
output
8

input
5 2 4 2 5 3
output
28

Note

Consider the first sample case. There are 3 towns and 3 roads. The towns are numbered from 1 to 3 and the roads are , , initially. Number the roads 1 to 3 in this order.

The sets of roads that ZS the Coder can flip (to make them not confusing) are $\{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}$. Note that the empty set is invalid because if no roads are flipped, then towns $1, 2, 3$ is form a directed cycle, so it is confusing. Similarly, flipping all roads is confusing too. Thus, there are a total of 6 possible sets ZS the Coder can flip.

The sample image shows all possible ways of orienting the roads from the first sample such that the network is **not confusing**.

E. ZS and The Birthday Paradox

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

ZS the Coder has recently found an interesting concept called the Birthday Paradox. It states that given a random set of 23 people, there is around 50% chance that some two of them share the same birthday. ZS the Coder finds this very interesting, and decides to test this with the inhabitants of Udayland.

In Udayland, there are 2^n days in a year. ZS the Coder wants to interview k people from Udayland, each of them has birthday in one of 2^n days (each day with equal probability). He is interested in the probability of at least two of them have the birthday at the same day.

ZS the Coder knows that the answer can be written as an irreducible fraction . He wants to find the values of A and B (he does not like to deal with floating point numbers). Can you help him?

Input

The first and only line of the input contains two integers n and k ($1 \le n \le 10^{18}, 2 \le k \le 10^{18}$), meaning that there are 2^n days in a year and that ZS the Coder wants to interview exactly k people.

Output

If the probability of at least two k people having the same birthday in 2^n days long year equals ($A \ge 0, B \ge 1,$), print the A and B in a single line.

Since these numbers may be too large, print them modulo $10^6 + 3$. Note that A and B must be coprime **before** their remainders modulo $10^6 + 3$ are taken.

Examples

input
3 2
output
1 8

input
1 3
output
1 1

input
4 3
output
23 128

Note

In the first sample case, there are $2^3 = 8$ days in Udayland. The probability that 2 people have the same birthday among 2 people is clearly , so $A = 1, B = 8$.

In the second sample case, there are only $2^1 = 2$ days in Udayland, but there are 3 people, so it is guaranteed that two of them have the same birthday. Thus, the probability is 1 and $A = B = 1$.