Codeforces Round #363 (Div. 2)

A. Launch of Collider

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

There will be a launch of a new, powerful and unusual collider very soon, which located along a straight line. n particles will be launched inside it. All of them are located in a straight line and there can not be two or more particles located in the same point. The coordinates of the particles coincide with the distance in meters from the center of the collider, x_i is the coordinate of the i-th particle and its position in the collider at the same time. All coordinates of particle positions are **even integers**.

You know the direction of each particle movement — it will move to the right or to the left after the collider's launch start. All particles begin to move simultaneously at the time of the collider's launch start. Each particle will move straight to the left or straight to the right with the constant speed of 1 meter per microsecond. The collider is big enough so particles can not leave it in the foreseeable time.

Write the program which finds the moment of the first collision of any two particles of the collider. In other words, find the number of microseconds before the first moment when any two particles are at the same point.

Input

The first line contains the positive integer n $(1 \le n \le 200\,000)$ — the number of particles.

The second line contains n symbols "L" and "R". If the i-th symbol equals "L", then the i-th particle will move to the left, otherwise the i-th symbol equals "R" and the i-th particle will move to the right.

The third line contains the sequence of pairwise distinct **even** integers $x_1, x_2, ..., x_n$ $(0 \le x_i \le 10^9)$ — the coordinates of particles in the order from the left to the right. It is guaranteed that the coordinates of particles are given in the increasing order.

Output

In the first line print the only integer — the first moment (in microseconds) when two particles are at the same point and there will be an explosion.

Print the only integer `-1`, if the collision of particles doesn't happen.

Examples

input
4 RLRL 2 4 6 10
output
1

input
3 LLR 40 50 60
output
-1

Note

In the first sample case the first explosion will happen in 1 microsecond because the particles number 1 and 2 will simultaneously be at the same point with the coordinate 3.

In the second sample case there will be no explosion because there are no particles which will simultaneously be at the same point.

B. One Bomb

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

You are given a description of a depot. It is a rectangular checkered field of $n \times m$ size. Each cell in a field can be empty ("`.`") or it can be occupied by a wall ("`*`").

You have one bomb. If you lay the bomb at the cell (x, y), then after triggering it will wipe out all walls in the row x and all walls in the column y.

You are to determine if it is possible to wipe out all walls in the depot by placing and triggering **exactly one bomb**. The bomb can be laid both in an empty cell or in a cell occupied by a wall.

Input

The first line contains two positive integers n and m $(1 \le n, m \le 1000)$ — the number of rows and columns in the depot field.

The next n lines contain m symbols "`.`" and "`*`" each — the description of the field. j-th symbol in i-th of them stands for cell (i, j). If the symbol is equal to "`.`", then the corresponding cell is empty, otherwise it equals "`*`" and the corresponding cell is occupied by a wall.

Output

If it is impossible to wipe out all walls by placing and triggering exactly one bomb, then print "`NO`" in the first line (without quotes).

Otherwise print "`YES`" (without quotes) in the first line and two integers in the second line — the coordinates of the cell at which the bomb should be laid. If there are multiple answers, print any of them.

Examples

input
3 4 .*.. .... .*..
output
YES 1 2

input
3 3 ..* .*. *..
output
NO

input
6 5 ..*.. ..*.. ***** ..*.. ..*.. ..*..
output
YES 3 3

C. Vacations

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

Vasya has n days of vacations! So he decided to improve his IT skills and do sport. Vasya knows the following information about each of this n days: whether that gym opened and whether a contest was carried out in the Internet on that day. For the i-th day there are four options:

1. on this day the gym is closed and the contest is not carried out;
2. on this day the gym is closed and the contest is carried out;
3. on this day the gym is open and the contest is not carried out;
4. on this day the gym is open and the contest is carried out.

On each of days Vasya can either have a rest or write the contest (if it is carried out on this day), or do sport (if the gym is open on this day).

Find the minimum number of days on which Vasya will have a rest (it means, he will not do sport and write the contest at the same time). The only limitation that Vasya has — *he does not want to do the same activity on two consecutive days: it means, he will not do sport on two consecutive days, and write the contest on two consecutive days.*

Input

The first line contains a positive integer n $(1 \le n \le 100)$ — the number of days of Vasya's vacations.

The second line contains the sequence of integers $a_1, a_2, ..., a_n$ $(0 \le a_i \le 3)$ separated by space, where:

- a_i equals 0, if on the i-th day of vacations the gym is closed and the contest is not carried out;
- a_i equals 1, if on the i-th day of vacations the gym is closed, but the contest is carried out;
- a_i equals 2, if on the i-th day of vacations the gym is open and the contest is not carried out;
- a_i equals 3, if on the i-th day of vacations the gym is open and the contest is carried out.

Output

Print the minimum possible number of days on which Vasya will have a rest. Remember that Vasya refuses:

- to do sport on any two consecutive days,
- to write the contest on any two consecutive days.

Examples

input
4 1 3 2 0
output
2

input
7 1 3 3 2 1 2 3
output
0

input
2 2 2
output
1

Note

In the first test Vasya can write the contest on the day number 1 and do sport on the day number 3. Thus, he will have a rest for only 2 days.

In the second test Vasya should write contests on days number 1, 3, 5 and 7, in other days do sport. Thus, he will not have a rest for a single day.

In the third test Vasya can do sport either on a day number 1 or number 2. He can not do sport in two days, because it will be contrary to the his limitation. Thus, he will have a rest for only one day.

D. Fix a Tree

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

A tree is an undirected connected graph without cycles.

Let's consider a rooted undirected tree with n vertices, numbered 1 through n. There are many ways to represent such a tree. One way is to create an array with n integers $p_1, p_2, ..., p_n$, where p_i denotes a parent of vertex i (here, for convenience a root is considered its own parent).

For this rooted tree the array p is $[2, 3, 3, 2]$.

Given a sequence $p_1, p_2, ..., p_n$, one is able to restore a tree:

1. There must be exactly one index r that $p_r = r$. A vertex r is a root of the tree.
2. For all other $n - 1$ vertices i, there is an edge between vertex i and vertex p_i.

A sequence $p_1, p_2, ..., p_n$ is called valid if the described procedure generates some (any) rooted tree. For example, for $n = 3$ sequences `(1,2,2)`, `(2,3,1)` and `(2,1,3)` **are not** valid.

You are given a sequence $a_1, a_2, ..., a_n$, not necessarily valid. Your task is to change the minimum number of elements, in order to get a valid sequence. Print the minimum number of changes and an example of a valid sequence after that number of changes. If there are many valid sequences achievable in the minimum number of changes, print any of them.

Input

The first line of the input contains an integer n $(2 \le n \le 200\,000)$ — the number of vertices in the tree.

The second line contains n integers $a_1, a_2, ..., a_n$ $(1 \le a_i \le n)$.

Output

In the first line print the minimum number of elements to change, in order to get a valid sequence.

In the second line, print any valid sequence possible to get from $(a_1, a_2, ..., a_n)$ in the minimum number of changes. If there are many such sequences, any of them will be accepted.

Examples

input
4 2 3 3 4
output
1 2 3 4 4

input
5 3 2 2 5 3
output
0 3 2 2 5 3

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

Note

In the first sample, it's enough to change one element. In the provided output, a sequence represents a tree rooted in a vertex 4 (because $p_4 = 4$), which you can see on the left drawing below. One of other correct solutions would be a sequence `2 3 3 2`, representing a tree rooted in vertex 3 (right drawing below). On both drawings, roots are painted red.

In the second sample, the given sequence is already valid.

E. LRU

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

While creating high loaded systems one should pay a special attention to caching. This problem will be about one of the most popular caching algorithms called LRU (Least Recently Used).

Suppose the cache may store no more than k objects. At the beginning of the workflow the cache is empty. When some object is queried we check if it is present in the cache and move it here if it's not. If there are more than k objects in the cache after this, the least recently used one should be removed. In other words, we remove the object that has the smallest time of the last query.

Consider there are n videos being stored on the server, all of the same size. Cache can store no more than k videos and caching algorithm described above is applied. We know that any time a user enters the server he pick the video i with probability p_i. The choice of the video is independent to any events before.

The goal of this problem is to count for each of the videos the probability it will be present in the cache after 10^{100} queries.

Input

The first line of the input contains two integers n and k ($1 \le k \le n \le 20$) — the number of videos and the size of the cache respectively. Next line contains n real numbers p_i ($0 \le p_i \le 1$), each of them is given with no more than two digits after decimal point.

It's guaranteed that the sum of all p_i is equal to 1.

Output

Print n real numbers, the i-th of them should be equal to the probability that the i-th video will be present in the cache after 10^{100} queries. You answer will be considered correct if its absolute or relative error does not exceed 10^{-6}.

Namely: let's assume that your answer is a, and the answer of the jury is b. The checker program will consider your answer correct, if .

Examples

input
3 1 0.3 0.2 0.5
output
0.3 0.2 0.5

input
2 1 0.0 1.0
output
0.0 1.0

input
3 2 0.3 0.2 0.5
output
0.675 0.4857142857142857 0.8392857142857143

input
3 3 0.2 0.3 0.5
output
1.0 1.0 1.0

F. Limak and Shooting Points

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

Bearland is a dangerous place. Limak can't travel on foot. Instead, he has k magic teleportation stones. Each stone can be used **at most once**. The i-th stone allows to teleport to a point (ax_i, ay_i). Limak can use stones **in any order**.

There are n monsters in Bearland. The i-th of them stands at (mx_i, my_i).

The given $k + n$ points are pairwise distinct.

After each teleportation, Limak can shoot an arrow in some direction. An arrow will hit the first monster in the chosen direction. Then, both an arrow and a monster disappear. It's dangerous to stay in one place for long, so Limak can shoot only one arrow from one place.

A monster should be afraid if it's possible that Limak will hit it. How many monsters should be afraid of Limak?

Input

The first line of the input contains two integers k and n $(1 \le k \le 7, 1 \le n \le 1000)$ — the number of stones and the number of monsters.

The i-th of following k lines contains two integers ax_i and ay_i $(-10^9 \le ax_i, ay_i \le 10^9)$ — coordinates to which Limak can teleport using the i-th stone.

The i-th of last n lines contains two integers mx_i and my_i $(-10^9 \le mx_i, my_i \le 10^9)$ — coordinates of the i-th monster.

The given $k + n$ points are pairwise distinct.

Output

Print the number of monsters which should be afraid of Limak.

Examples

input
2 4 -2 -1 4 5 4 2 2 1 4 -1 1 -1

output
3

input
3 8 10 20 0 0 20 40 300 600 30 60 170 340 50 100 28 56 90 180 -4 -8 -1 -2

output
5

Note

In the first sample, there are two stones and four monsters. Stones allow to teleport to points $(-2, -1)$ and $(4, 5)$, marked blue in the drawing below. Monsters are at $(4, 2)$, $(2, 1)$, $(4, -1)$ and $(1, -1)$, marked red. A monster at $(4, -1)$ shouldn't be afraid because it's impossible that Limak will hit it with an arrow. Other three monsters can be hit and thus the answer is 3.

In the second sample, five monsters should be afraid. Safe monsters are those at $(300, 600)$, $(170, 340)$ and $(90, 180)$.