A. Calculating Function

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

For a positive integer n let's define a function f:

$$f(n) = -1 + 2 - 3 + .. + (-1)^n n$$

Your task is to calculate $f(n)$ for a given integer n.

Input

The single line contains the positive integer n ($1 \leq n \leq 10^{15}$).

Output

Print $f(n)$ in a single line.

Examples

input

```
4
```

output

```
2
```

input

```
5
```

output

```
-3
```

Note

$f(4) = -1 + 2 - 3 + 4 = 2$

$f(5) = -1 + 2 - 3 + 4 - 5 = -3$

B. OR in Matrix

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

Let's define logical OR as an operation on two logical values (i. e. values that belong to the set $\{0, 1\}$) that is equal to 1 if either or both of the logical values is set to 1, otherwise it is 0. We can define logical OR of three or more logical values in the same manner:

$a_1 \text{ OR } a_2 \text{ OR} \ldots \text{OR } a_k$ where $a_i \in \{0, 1\}$ is equal to 1 if some $a_i = 1$, otherwise it is equal to 0.

Nam has a matrix A consisting of m rows and n columns. The rows are numbered from 1 to m, columns are numbered from 1 to n. Element at row i ($1 \le i \le m$) and column j ($1 \le j \le n$) is denoted as A_{ij}. All elements of A are either 0 or 1. From matrix A, Nam creates another matrix B of the same size using formula:

$B_{ij} = A_{i1} \text{ OR } A_{i2} \text{ OR} \ldots \text{OR } A_{in} \text{ OR } A_{1j} \text{ OR } A_{2j} \text{ OR} \ldots \text{OR } A_{mj}$.

(B_{ij} is OR of all elements in row i and column j of matrix A)

Nam gives you matrix B and challenges you to guess matrix A. Although Nam is smart, he could probably make a mistake while calculating matrix B, since size of A can be large.

Input

The first line contains two integer m and n ($1 \le m, n \le 100$), number of rows and number of columns of matrices respectively.

The next m lines each contain n integers separated by spaces describing rows of matrix B (each element of B is either 0 or 1).

Output

In the first line, print "NO" if Nam has made a mistake when calculating B, otherwise print "YES". If the first line is "YES", then also print m rows consisting of n integers representing matrix A that can produce given matrix B. If there are several solutions print any one.

Examples

input

```
2 2
1 0
0 0
```

output

```
NO
```

input

```
2 3
1 1 1
1 1 1
```

output

```
YES
1 1 1
1 1 1
```

input

```
2 3
0 1 0
1 1 1
```

output

```
YES
0 0 0
0 1 0
```

C. Palindrome Transformation

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

Nam is playing with a string on his computer. The string consists of n lowercase English letters. It is meaningless, so Nam decided to make the string more beautiful, that is to make it be a palindrome by using 4 arrow keys: left, right, up, down.

There is a cursor pointing at some symbol of the string. Suppose that cursor is at position i ($1 \le i \le n$, the string uses 1-based indexing) now. Left and right arrow keys are used to move cursor around the string. The string is cyclic, that means that when Nam presses left arrow key, the cursor will move to position $i - 1$ if $i > 1$ or to the end of the string (i. e. position n) otherwise. The same holds when he presses the right arrow key (if $i = n$, the cursor appears at the beginning of the string).

When Nam presses up arrow key, the letter which the text cursor is pointing to will change to the next letter in English alphabet (assuming that alphabet is also cyclic, i. e. after 'z' follows 'a'). The same holds when he presses the down arrow key.

Initially, the text cursor is at position p.

Because Nam has a lot homework to do, he wants to complete this as fast as possible. Can you help him by calculating the minimum number of arrow keys presses to make the string to be a palindrome?

Input

The first line contains two space-separated integers n ($1 \le n \le 10^5$) and p ($1 \le p \le n$), the length of Nam's string and the initial position of the text cursor.

The next line contains n lowercase characters of Nam's string.

Output

Print the minimum number of presses needed to change string into a palindrome.

Examples

input

```
8 3
aeabcaez
```

output

```
6
```

Note

A string is a palindrome if it reads the same forward or reversed.

In the sample test, initial Nam's string is: $\text{ae}\mathbf{a}\text{bcaez}$ (cursor position is shown bold).

In optimal solution, Nam may do 6 following steps:

$$\text{ae}\mathbf{a}\text{bcaez} \xrightarrow{right} \text{aea}\mathbf{b}\text{caez} \xrightarrow{up} \text{aea}\mathbf{c}\text{caez} \xrightarrow{left} \\ \text{ae}\mathbf{a}\text{ccaez} \xrightarrow{left} \text{a}\mathbf{e}\text{accaez} \xrightarrow{left} \mathbf{a}\text{eaccaez} \xrightarrow{down} \mathbf{z}\text{eaccaez}$$

The result, zeaccaez, is now a palindrome.

D. Valid Sets

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

As you know, an undirected connected graph with n nodes and $n - 1$ edges is called a *tree*. You are given an integer d and a tree consisting of n nodes. Each node i has a value a_i associated with it.

We call a set S of tree nodes *valid* if following conditions are satisfied:

1. S is non-empty.
2. S is connected. In other words, if nodes u and v are in S, then all nodes lying on the simple path between u and v should also be presented in S.
3. $\max_{u \in S} a_u - \min_{v \in S} a_v \le d$.

Your task is to count the number of valid sets. Since the result can be very large, you must print its remainder modulo 1000000007 ($10^9 + 7$).

Input

The first line contains two space-separated integers d ($0 \le d \le 2000$) and n ($1 \le n \le 2000$).

The second line contains n space-separated positive integers $a_1, a_2, \ldots, a_n$($1 \le a_i \le 2000$).

Then the next $n - 1$ line each contain pair of integers u and v ($1 \le u, v \le n$) denoting that there is an edge between u and v. It is guaranteed that these edges form a tree.

Output

Print the number of valid sets modulo 1000000007.

Examples

input

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

output

```
8
```

input

```
0 3
1 2 3
1 2
2 3
```

output

```
3
```

input

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

output

```
41
```

Note

In the first sample, there are exactly 8 valid sets: $\{1\}, \{2\}, \{3\}, \{4\}, \{1, 2\}, \{1, 3\}, \{3, 4\}$ and $\{1, 3, 4\}$. Set $\{1, 2, 3, 4\}$ is not valid, because the third condition isn't satisfied. Set $\{1, 4\}$ satisfies the third condition, but conflicts with the second condition.

E. LIS of Sequence

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

The next "Data Structures and Algorithms" lesson will be about Longest Increasing Subsequence (LIS for short) of a sequence. For better understanding, Nam decided to learn it a few days before the lesson.

Nam created a sequence a consisting of n ($1 \le n \le 10^5$) elements $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^5$). A subsequence $a_{i_1}, a_{i_2}, \ldots, a_{i_k}$ where $1 \le i_1 < i_2 < \ldots < i_k \le n$ is called increasing if $a_{i_1} < a_{i_2} < a_{i_3} < \ldots < a_{i_k}$. An increasing subsequence is called longest if it has maximum length among all increasing subsequences.

Nam realizes that a sequence may have several longest increasing subsequences. Hence, he divides all indexes i ($1 \le i \le n$), into three groups:

1. group of all i such that a_i belongs to no longest increasing subsequences.
2. group of all i such that a_i belongs to at least one **but not every** longest increasing subsequence.
3. group of all i such that a_i belongs to every longest increasing subsequence.

Since the number of longest increasing subsequences of a may be very large, categorizing process is very difficult. Your task is to help him finish this job.

Input

The first line contains the single integer n ($1 \le n \le 10^5$) denoting the number of elements of sequence a.

The second line contains n space-separated integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^5$).

Output

Print a string consisting of n characters. i-th character should be '1', '2' or '3' depending on which group among listed above index i belongs to.

Examples

input
```
1
4
```
output
```
3
```

input
```
4
1 3 2 5
```
output
```
3223
```

input
```
4
1 5 2 3
```
output
```
3133
```

Note

In the second sample, sequence a consists of 4 elements: $\{a_1, a_2, a_3, a_4\} = \{1, 3, 2, 5\}$. Sequence a has exactly 2 longest increasing subsequences of length 3, they are $\{a_1, a_2, a_4\} = \{1, 3, 5\}$ and $\{a_1, a_3, a_4\} = \{1, 2, 5\}$.

In the third sample, sequence a consists of 4 elements: $\{a_1, a_2, a_3, a_4\} = \{1, 5, 2, 3\}$. Sequence a have exactly 1 longest increasing subsequence of length 3, that is $\{a_1, a_3, a_4\} = \{1, 2, 3\}$.