Codeforces Round #225 (Div. 1)

A. Milking cows

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

Iahub helps his grandfather at the farm. Today he must milk the cows. There are n cows sitting in a row, numbered from 1 to n from left to right. Each cow is either facing to the left or facing to the right. When Iahub milks a cow, all the cows that see the current cow get scared and lose one unit of the quantity of milk that they can give. A cow facing left sees all the cows with lower indices than her index, and a cow facing right sees all the cows with higher indices than her index. A cow that got scared once can get scared again (and lose one more unit of milk). A cow that has been milked once cannot get scared and lose any more milk. You can assume that a cow never loses all the milk she can give (a cow gives an infinitely amount of milk).

Iahub can decide the order in which he milks the cows. But he must milk each cow exactly once. Iahub wants to lose as little milk as possible. Print the minimum amount of milk that is lost.

Input

The first line contains an integer n $(1 \le n \le 200000)$. The second line contains n integers $a_1, a_2, ..., a_n$, where a_i is 0 if the cow number i is facing left, and 1 if it is facing right.

Output

Print a single integer, the minimum amount of lost milk.

Please, do not write the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use the `cin`, `cout` streams or the `%I64d` specifier.

Sample test(s)

input
4 0 0 1 0
output
1

input
5 1 0 1 0 1
output
3

Note

In the first sample Iahub milks the cows in the following order: cow 3, cow 4, cow 2, cow 1. When he milks cow 3, cow 4 loses 1 unit of milk. After that, no more milk is lost.

B. Volcanoes

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

Iahub got lost in a very big desert. The desert can be represented as a $n \times n$ square matrix, where each cell is a zone of the desert. The cell (i, j) represents the cell at row i and column j $(1 \le i, j \le n)$. Iahub can go from one cell (i, j) only down or right, that is to cells $(i + 1, j)$ or $(i, j + 1)$.

Also, there are m cells that are occupied by volcanoes, which Iahub cannot enter.

Iahub is initially at cell $(1, 1)$ and he needs to travel to cell (n, n). Knowing that Iahub needs 1 second to travel from one cell to another, find the minimum time in which he can arrive in cell (n, n).

Input

The first line contains two integers n $(1 \le n \le 10^9)$ and m $(1 \le m \le 10^5)$. Each of the next m lines contains a pair of integers, x and y $(1 \le x, y \le n)$, representing the coordinates of the volcanoes.

Consider matrix rows are numbered from 1 to n from top to bottom, and matrix columns are numbered from 1 to n from left to right. There is no volcano in cell $(1, 1)$. No two volcanoes occupy the same location.

Output

Print one integer, the minimum time in which Iahub can arrive at cell (n, n). If no solution exists (there is no path to the final cell), print -1.

Sample test(s)

input
4 2 1 3 1 4

output
6

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

output
12

input
2 2 1 2 2 1

output
-1

Note

Consider the first sample. A possible road is: $(1, 1) \rightarrow (1, 2) \rightarrow (2, 2) \rightarrow (2, 3) \rightarrow (3, 3) \rightarrow (3, 4) \rightarrow (4, 4)$.

C. Propagating tree

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

Iahub likes trees very much. Recently he discovered an interesting tree named propagating tree. The tree consists of n nodes numbered from 1 to n, each node i having an initial value a_i. The root of the tree is node 1.

This tree has a special property: when a value val is added to a value of node i, the value -val is added to values of all the children of node i. Note that when you add value -val to a child of node i, you also add -(-val) to all children of the child of node i and so on. Look an example explanation to understand better how it works.

This tree supports two types of queries:

- "$1\ x\ val$" — val is added to the value of node x;
- "$2\ x$" — print the current value of node x.

In order to help Iahub understand the tree better, you must answer m queries of the preceding type.

Input

The first line contains two integers n and m $(1 \le n, m \le 200000)$. The second line contains n integers a_1, a_2, ..., a_n $(1 \le a_i \le 1000)$. Each of the next $n-1$ lines contains two integers v_i and u_i $(1 \le v_i, u_i \le n)$, meaning that there is an edge between nodes v_i and u_i.

Each of the next m lines contains a query in the format described above. It is guaranteed that the following constraints hold for all queries: $1 \le x \le n, 1 \le val \le 1000$.

Output

For each query of type two (print the value of node x) you must print the answer to the query on a separate line. The queries must be answered in the order given in the input.

Sample test(s)

input

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

output

```
3
3
0
```

Note

The values of the nodes are $[1, 2, 1, 1, 2]$ at the beginning.

Then value 3 is added to node 2. It propagates and value -3 is added to it's sons, node 4 and node 5. Then it cannot propagate any more. So the values of the nodes are $[1, 5, 1, -2, -1]$.

Then value 2 is added to node 1. It propagates and value -2 is added to it's sons, node 2 and node 3. From node 2 it propagates again, adding value 2 to it's sons, node 4 and node 5. Node 3 has no sons, so it cannot propagate from there. The values of the nodes are $[3, 3, -1, 0, 1]$.

You can see all the definitions about the tree at the following link: http://en.wikipedia.org/wiki/Tree_(graph_theory)

D. Antimatter

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

Iahub accidentally discovered a secret lab. He found there n devices ordered in a line, numbered from 1 to n from left to right. Each device i $(1 \le i \le n)$ can create either a_i units of matter or a_i units of antimatter.

Iahub wants to choose some contiguous subarray of devices in the lab, specify the production mode for each of them (produce matter or antimatter) and finally take a photo of it. However he will be successful only if the amounts of matter and antimatter produced in the selected subarray will be the same (otherwise there would be overflowing matter or antimatter in the photo).

You are requested to compute the number of different ways Iahub can successful take a photo. A photo is different than another if it represents another subarray, or if at least one device of the subarray is set to produce matter in one of the photos and antimatter in the other one.

Input

The first line contains an integer n $(1 \le n \le 1000)$. The second line contains n integers a_1, a_2, ..., a_n $(1 \le a_i \le 1000)$.

The sum $a_1 + a_2 + ... + a_n$ will be less than or equal to 10000.

Output

Output a single integer, the number of ways Iahub can take a photo, modulo 1000000007 $(10^9 + 7)$.

Sample test(s)

input
4 1 1 1 1
output
12

Note

The possible photos are [1+, 2-], [1-, 2+], [2+, 3-], [2-, 3+], [3+, 4-], [3-, 4+], [1+, 2+, 3-, 4-], [1+, 2-, 3+, 4-], [1+, 2-, 3-, 4+], [1-, 2+, 3+, 4-], [1-, 2+, 3-, 4+] and [1-, 2-, 3+, 4+], where "i+" means that the i-th element produces matter, and "i-" means that the i-th element produces antimatter.

E. Vowels

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Iahubina is tired of so many complicated languages, so she decided to invent a new, simple language. She already made a dictionary consisting of n 3-words. A 3-word is a sequence of exactly 3 lowercase letters of the first 24 letters of the English alphabet (a to x). She decided that some of the letters are vowels, and all the others are consonants. The whole language is based on a simple rule: any word that contains at least one vowel is *correct*.

Iahubina forgot which letters are the vowels, and wants to find some possible correct sets of vowels. She asks Iahub questions. In each question, she will give Iahub a set of letters considered vowels (in this question). For each question she wants to know how many words of the dictionary are correct, considering the given set of vowels.

Iahubina wants to know the xor of the squared answers to all the possible questions. There are 2^{24} different questions, they are all subsets of the set of the first 24 letters of the English alphabet. Help Iahub find that number.

Input

The first line contains one integer, n ($1 \leq n \leq 10^4$). Each of the next n lines contains a 3-word consisting of 3 lowercase letters. There will be no two identical 3-words.

Output

Print one number, the xor of the squared answers to the queries.

Sample test(s)

input
5 abc aaa ada bcd def
output
0