Codeforces Round #FF (Div. 2)

A. DZY Loves Hash

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

DZY has a hash table with p buckets, numbered from 0 to $p - 1$. He wants to insert n numbers, in the order they are given, into the hash table. For the i-th number x_i, DZY will put it into the bucket numbered $h(x_i)$, where $h(x)$ is the hash function. In this problem we will assume, that $h(x) = x \ mod \ p$. Operation $a \ mod \ b$ denotes taking a remainder after division a by b.

However, each bucket can contain no more than one element. If DZY wants to insert an number into a bucket which is already filled, we say a "conflict" happens. Suppose the first conflict happens right after the i-th insertion, you should output i. If no conflict happens, just output `-1`.

Input

The first line contains two integers, p and n $(2 \le p, n \le 300)$. Then n lines follow. The i-th of them contains an integer x_i $(0 \le x_i \le 10^9)$.

Output

Output a single integer — the answer to the problem.

Sample test(s)

input

```
10 5
0
21
53
41
53
```

output

```
4
```

input

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

output

```
-1
```

B. DZY Loves Strings

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

DZY loves collecting special strings which only contain lowercase letters. For each lowercase letter c DZY knows its value w_c. For each special string $s = s_1 s_2 \ldots s_{|s|}$ ($|s|$ is the length of the string) he represents its value with a function $f(s)$, where

$$f(s) = \sum_{i=1}^{|s|} (w_{s_i} \cdot i).$$

Now DZY has a string s. He wants to insert k lowercase letters into this string in order to get the largest possible value of the resulting string. Can you help him calculate the largest possible value he could get?

Input

The first line contains a single string s $(1 \le |s| \le 10^3)$.

The second line contains a single integer k $(0 \le k \le 10^3)$.

The third line contains twenty-six integers from w_a to w_z. Each such number is non-negative and doesn't exceed 1000.

Output

Print a single integer — the largest possible value of the resulting string DZY could get.

Sample test(s)

input
abc 3 1 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
output
41

Note

In the test sample DZY can obtain "abcbbc", $value = 1 \cdot 1 + 2 \cdot 2 + 3 \cdot 2 + 4 \cdot 2 + 5 \cdot 2 + 6 \cdot 2 = 41$.

C. DZY Loves Sequences

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

DZY has a sequence a, consisting of n integers.

We'll call a sequence $a_i, a_{i+1}, ..., a_j$ $(1 \le i \le j \le n)$ a subsegment of the sequence a. The value $(j - i + 1)$ denotes the length of the subsegment.

Your task is to find the longest subsegment of a, such that it is possible to change at most one number (change one number to any integer you want) from the subsegment to make the subsegment strictly increasing.

You only need to output the length of the subsegment you find.

Input

The first line contains integer n $(1 \le n \le 10^5)$. The next line contains n integers $a_1, a_2, ..., a_n$ $(1 \le a_i \le 10^9)$.

Output

In a single line print the answer to the problem — the maximum length of the required subsegment.

Sample test(s)

input
6 7 2 3 1 5 6
output
5

Note

You can choose subsegment a_2, a_3, a_4, a_5, a_6 and change its 3rd element (that is a_4) to 4.

D. DZY Loves Modification

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

As we know, DZY loves playing games. One day DZY decided to play with a $n \times m$ matrix. To be more precise, he decided to modify the matrix with exactly k operations.

Each modification is one of the following:

1. Pick some row of the matrix and decrease each element of the row by p. This operation brings to DZY the value of pleasure equal to the sum of elements of the row before the decreasing.
2. Pick some column of the matrix and decrease each element of the column by p. This operation brings to DZY the value of pleasure equal to the sum of elements of the column before the decreasing.

DZY wants to know: what is the largest total value of pleasure he could get after performing exactly k modifications? Please, help him to calculate this value.

Input

The first line contains four space-separated integers n, m, k and p $(1 \le n, m \le 10^3; 1 \le k \le 10^6; 1 \le p \le 100)$.

Then n lines follow. Each of them contains m integers representing a_{ij} $(1 \le a_{ij} \le 10^3)$ — the elements of the current row of the matrix.

Output

Output a single integer — the maximum possible total pleasure value DZY could get.

Sample test(s)

input
2 2 2 2 1 3 2 4

output
11

input
2 2 5 2 1 3 2 4

output
11

Note

For the first sample test, we can modify: column 2, row 2. After that the matrix becomes:

```
1 1
0 0
```

For the second sample test, we can modify: column 2, row 2, row 1, column 1, column 2. After that the matrix becomes:

```
-3 -3
-2 -2
```

E. DZY Loves Fibonacci Numbers

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

In mathematical terms, the sequence F_n of Fibonacci numbers is defined by the recurrence relation

$$F_1 = 1;\ F_2 = 1;\ F_n = F_{n-1} + F_{n-2}\ (n > 2).$$

DZY loves Fibonacci numbers very much. Today DZY gives you an array consisting of n integers: $a_1, a_2, ..., a_n$. Moreover, there are m queries, each query has one of the two types:

1. Format of the query "$1\ l\ r$". In reply to the query, you need to add F_{i-l+1} to each element a_i, where $l \le i \le r$.
2. Format of the query "$2\ l\ r$". In reply to the query you should output the value of $\sum_{i=l}^{r} a_i$ modulo $1000000009\ (10^9 + 9)$.

Help DZY reply to all the queries.

Input

The first line of the input contains two integers n and m ($1 \le n, m \le 300000$). The second line contains n integers $a_1, a_2, ..., a_n$ ($1 \le a_i \le 10^9$) – initial array a.

Then, m lines follow. A single line describes a single query in the format given in the statement. It is guaranteed that for each query inequality $1 \le l \le r \le n$ holds.

Output

For each query of the second type, print the value of the sum on a single line.

Sample test(s)

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

output
17 12

Note

After the first query, $a = [2, 3, 5, 7]$.

For the second query, $sum = 2 + 3 + 5 + 7 = 17$.

After the third query, $a = [2, 4, 6, 9]$.

For the fourth query, $sum = 2 + 4 + 6 = 12$.