Codeforces Round #191 (Div. 2)

A. Flipping Game

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

Iahub got bored, so he invented a game to be played on paper.

He writes n integers $a_1, a_2, ..., a_n$. Each of those integers can be either 0 or 1. He's allowed to do exactly one move: he chooses two indices i and j ($1 \le i \le j \le n$) and flips all values a_k for which their positions are in range $[i, j]$ (that is $i \le k \le j$). Flip the value of x means to apply operation $x = 1 - x$.

The goal of the game is that after **exactly** one move to obtain the maximum number of ones. Write a program to solve the little game of Iahub.

Input
The first line of the input contains an integer n ($1 \le n \le 100$). In the second line of the input there are n integers: $a_1, a_2, ..., a_n$. It is guaranteed that each of those n values is either 0 or 1.

Output
Print an integer — the maximal number of 1s that can be obtained after exactly one move.

Examples

input
5 1 0 0 1 0
output
4

input
4 1 0 0 1
output
4

Note
In the first case, flip the segment from 2 to 5 ($i = 2, j = 5$). That flip changes the sequence, it becomes: [1 1 1 0 1]. So, it contains four ones. There is no way to make the whole sequence equal to [1 1 1 1 1].

In the second case, flipping only the second and the third element ($i = 2, j = 3$) will turn all numbers into 1.

B. Hungry Sequence

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

Iahub and Iahubina went to a date at a luxury restaurant. Everything went fine until paying for the food. Instead of money, the waiter wants Iahub to write a Hungry sequence consisting of n integers.

A sequence a_1, a_2, ..., a_n, consisting of n integers, is *Hungry* if and only if:

- Its elements are in increasing order. That is an inequality $a_i < a_j$ holds for any two indices i, j $(i < j)$.
- For any two indices i and j $(i < j)$, a_j must **not** be divisible by a_i.

Iahub is in trouble, so he asks you for help. Find a Hungry sequence with n elements.

Input

The input contains a single integer: n $(1 \le n \le 10^5)$.

Output

Output a line that contains n space-separated integers a_1 a_2, ..., a_n $(1 \le a_i \le 10^7)$, representing a possible Hungry sequence. Note, that each a_i must not be greater than 10000000 (10^7) and less than 1.

If there are multiple solutions you can output any one.

Examples

input
3
output
2 9 15

input
5
output
11 14 20 27 31

C. Magic Five

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

There is a long plate s containing n digits. Iahub wants to delete some digits (possibly none, but he is not allowed to delete all the digits) to form his "magic number" on the plate, a number that is divisible by 5. Note that, the resulting number may contain leading zeros.

Now Iahub wants to count the number of ways he can obtain magic number, modulo 1000000007 ($10^9 + 7$). Two ways are different, if the set of deleted positions in s differs.

Look at the input part of the statement, s is given in a special form.

Input

In the first line you're given a string a ($1 \le |a| \le 10^5$), containing digits only. In the second line you're given an integer k ($1 \le k \le 10^9$). The plate s is formed by concatenating k copies of a together. That is $n = |a| \cdot k$.

Output

Print a single integer — the required number of ways modulo 1000000007 ($10^9 + 7$).

Examples

input
1256 1
output
4

input
13990 2
output
528

input
555 2
output
63

Note

In the first case, there are four possible ways to make a number that is divisible by 5: 5, 15, 25 and 125.

In the second case, remember to concatenate the copies of a. The actual plate is 1399013990.

In the third case, except deleting all digits, any choice will do. Therefore there are $2^6 - 1 = 63$ possible ways to delete digits.

D. Block Tower

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

After too much playing on paper, Iahub has switched to computer games. The game he plays is called "Block Towers". It is played in a rectangular grid with n rows and m columns (it contains $n \times m$ cells). The goal of the game is to build your own city. Some cells in the grid are big holes, where Iahub can't build any building. The rest of cells are empty. In some empty cell Iahub can build exactly one tower of two following types:

1. Blue towers. Each has population limit equal to 100.
2. Red towers. Each has population limit equal to 200. However, it can be built in some cell only if in that moment at least one of the neighbouring cells has a Blue Tower. Two cells are neighbours is they share a side.

Iahub is also allowed to destroy a building from any cell. He can do this operation as much as he wants. After destroying a building, the other buildings are not influenced, and the destroyed cell becomes empty (so Iahub can build a tower in this cell if needed, see the second example for such a case).

Iahub can convince as many population as he wants to come into his city. So he needs to configure his city to allow maximum population possible. Therefore he should find a sequence of operations that builds the city in an optimal way, so that total population limit is as large as possible.

He says he's the best at this game, but he doesn't have the optimal solution. Write a program that calculates the optimal one, to show him that he's not as good as he thinks.

Input

The first line of the input contains two integers n and m ($1 \le n, m \le 500$). Each of the next n lines contains m characters, describing the grid. The j-th character in the i-th line is '.' if you're allowed to build at the cell with coordinates (i, j) a tower (empty cell) or '#' if there is a big hole there.

Output

Print an integer k in the first line ($0 \le k \le 10^6$) — the number of operations Iahub should perform to obtain optimal result.

Each of the following k lines must contain a single operation in the following format:

1. «B x y» ($1 \le x \le n, 1 \le y \le m$) — building a blue tower at the cell (x, y);
2. «R x y» ($1 \le x \le n, 1 \le y \le m$) — building a red tower at the cell (x, y);
3. «D x y» ($1 \le x \le n, 1 \le y \le m$) — destroying a tower at the cell (x, y).

If there are multiple solutions you can output any of them. Note, that you shouldn't minimize the number of operations.

Examples

input
2 3 ..# .#.
output
4 B 1 1 R 1 2 R 2 1 B 2 3

input
1 3 ...
output
5 B 1 1 B 1 2 R 1 3 D 1 2 R 1 2

E. Axis Walking

time limit per test: 3 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Iahub wants to meet his girlfriend Iahubina. They both live in Ox axis (the horizontal axis). Iahub lives at point 0 and Iahubina at point d.

Iahub has n positive integers a_1, a_2, ..., a_n. The sum of those numbers is d. Suppose p_1, p_2, ..., p_n is a permutation of $\{1, 2, ..., n\}$. Then, let $b_1 = a_{p_1}$, $b_2 = a_{p_2}$ and so on. The array b is called a "route". There are $n!$ different routes, one for each permutation p.

Iahub's travel schedule is: he walks b_1 steps on Ox axis, then he makes a break in point b_1. Then, he walks b_2 more steps on Ox axis and makes a break in point $b_1 + b_2$. Similarly, at j-th $(1 \le j \le n)$ time he walks b_j more steps on Ox axis and makes a break in point $b_1 + b_2 + ... + b_j$.

Iahub is very superstitious and has k integers which give him bad luck. He calls a route "good" if he never makes a break in a point corresponding to one of those k numbers. For his own curiosity, answer how many good routes he can make, modulo 1000000007 $(10^9 + 7)$.

Input

The first line contains an integer n $(1 \le n \le 24)$. The following line contains n integers: $a_1, a_2, ..., a_n$ $(1 \le a_i \le 10^9)$.

The third line contains integer k $(0 \le k \le 2)$. The fourth line contains k positive integers, representing the numbers that give Iahub bad luck. Each of these numbers does not exceed 10^9.

Output

Output a single integer — the answer of Iahub's dilemma modulo 1000000007 $(10^9 + 7)$.

Examples

input
3 2 3 5 2 5 7
output
1

input
3 2 2 2 2 1 3
output
6

Note

In the first case consider six possible orderings:

- [2, 3, 5]. Iahub will stop at position 2, 5 and 10. Among them, 5 is bad luck for him.
- [2, 5, 3]. Iahub will stop at position 2, 7 and 10. Among them, 7 is bad luck for him.
- [3, 2, 5]. He will stop at the unlucky 5.
- [3, 5, 2]. This is a valid ordering.
- [5, 2, 3]. He got unlucky twice (5 and 7).
- [5, 3, 2]. Iahub would reject, as it sends him to position 5.

In the second case, note that it is possible that two different ways have the identical set of stopping. In fact, all six possible ways have the same stops: [2, 4, 6], so there's no bad luck for Iahub.