A. Rating Increase

```
Input file:     standard input
Output file:    standard output
Time limit:     2 seconds
Memory limit:   256 megabytes
```

Monocarp is a great solver of adhoc problems. Recently, he participated in an Educational Codeforces Round, and gained rating!

Monocarp knew that, before the round, his rating was a. After the round, it increased to b ($b > a$). He wrote both values one after another to not forget them.

However, he wrote them so close to each other, that he can't tell now where the first value ends and the second value starts.

Please, help him find some values a and b such that:

- neither of them has a leading zero;
- both of them are strictly greater than 0;
- $b > a$;
- they produce the given value ab when written one after another.

If there are multiple answers, you can print any of them.

Input

The first line contains a single integer t ($1 \le t \le 10^4$) — the number of testcases.

The only line of each testcase consists of a single string ab of length from 2 to 8 that:

- consists only of digits;
- doesn't start with a zero.

Output

For each testcase, determine if such values a and b exist. If they don't, print `-1`. Otherwise, print two integers a and b.

If there are multiple answers, you can print any of them.

Standard Input	Standard Output

5
20002001
391125
200200
2001000
12
	2000 2001
	39 1125
	-1
	200 1000
	1 2

Note

In the second testcase, printing 3 and 91125 is also valid.

In the third testcase, 20 and 0200 is not valid, because b has a leading zero. 200 and 200 is not valid, because 200 is not strictly greater than 200.

,

B. Swap and Delete

```
Input file:     standard input
Output file:    standard output
Time limit:     1 second
Memory limit:   256 megabytes
```

You are given a binary string s (a string consisting only of `0`-s and `1`-s).

You can perform two types of operations on s:

1. delete one character from s. This operation costs 1 coin;
2. swap any pair of characters in s. This operation is free (costs 0 coins).

You can perform these operations any number of times and in any order.

Let's name a string you've got after performing operations above as t. The string t is *good* if for each i from 1 **to** $|t|$ $t_i \neq s_i$ ($|t|$ is the length of the string t). The empty string is *always good*. Note that you are comparing the resulting string t with the *initial string* s.

What is the minimum total cost to make the string t good?

Input

The first line contains a single integer t ($1 \le t \le 10^4$) — the number of test cases. Then t test cases follow.

The only line of each test case contains a binary string s ($1 \le |s| \le 2 \cdot 10^5$; $s_i \in \{$`0`, `1`$\}$) — the initial string, consisting of characters `0` and/or `1`.

Additional constraint on the input: the total length of all strings s doesn't exceed $2 \cdot 10^5$.

Output

For each test case, print one integer — the minimum total cost to make string t good.

Standard Input	Standard Output
4 0 011 0101110001 111100	1 1 0 4

Note

In the first test case, you have to delete a character from s to get the empty string t. Only then t becomes good. One deletion costs 1 coin.

In the second test case, you can, for example, delete the second character from s to get the string `01`, and then swap the first and second characters to get the string $t =$ `10`. String t is good, since $t_1 \neq s_1$ and $t_2 \neq s_2$. The total cost is 1 coin.

In the third test case, you can, for example, swap s_1 with s_2, swap s_3 with s_4, swap s_5 with s_7, s_6 with s_8 and s_9 with s_{10}. You'll get $t =$ `1010001110`. All swap operations are free, so the total cost is 0.

,

C. Game with Multiset

```
Input file:     standard input
Output file:    standard output
Time limit:     1.5 seconds
Memory limit:   256 megabytes
```

In this problem, you are initially given an empty multiset. You have to process two types of queries:

1. ADD x — add an element equal to 2^x to the multiset;
2. GET w — say whether it is possible to take the sum of some subset of the current multiset and get a value equal to w.

Input

The first line contains one integer m $(1 \le m \le 10^5)$ — the number of queries.

Then m lines follow, each of which contains two integers t_i, v_i, denoting the i-th query. If $t_i = 1$, then the i-th query is ADD v_i $(0 \le v_i \le 29)$. If $t_i = 2$, then the i-th query is GET v_i $(0 \le v_i \le 10^9)$.

Output

For each GET query, print YES if it is possible to choose a subset with sum equal to w, or NO if it is impossible.

Standard Input	Standard Output

5
1 0
1 0
1 0
2 3
2 4
	YES
	NO

7
1 0
1 1
1 2
1 10
2 4
2 6
2 7
	YES
	YES
	YES

,

D. Array Collapse

```
Input file:     standard input
Output file:    standard output
Time limit:     2 seconds
Memory limit:   256 megabytes
```

You are given an array $[p_1, p_2, \dots, p_n]$, where all elements are distinct.

You can perform several (possibly zero) operations with it. In one operation, you can choose a **contiguous subsegment** of p and remove **all** elements from that subsegment, **except for** the minimum element on that subsegment. For example, if $p = [3, 1, 4, 7, 5, 2, 6]$ and you choose the subsegment from the 3-rd element to the 6-th element, the resulting array is $[3, 1, 2, 6]$.

An array a is called *reachable* if it can be obtained from p using several (maybe zero) aforementioned operations. Calculate the number of *reachable* arrays, and print it modulo 998244353.

Input

The first line of the input contains one integer t ($1 \le t \le 10^4$) — the number of test cases.

Each test case consists of two lines. The first line contains one integer n ($1 \le n \le 3 \cdot 10^5$). The second line contains n **distinct** integers $p_1, p_2, \dots, p_n$ ($1 \le p_i \le 10^9$).

Additional constraint on the input: the sum of n over all test cases does not exceed $3 \cdot 10^5$.

Output

For each test case, print one integer — the number of *reachable* arrays, taken modulo 998244353.

Standard Input	Standard Output

3
2
2 1
4
2 4 1 3
5
10 2 6 3 4
	2
	6
	12

E. Matrix Problem

```
Input file:     standard input
Output file:    standard output
Time limit:     1 second
Memory limit:   256 megabytes
```

You are given a matrix a, consisting of n rows by m columns. Each element of the matrix is equal to 0 or 1.

You can perform the following operation any number of times (possibly zero): choose an element of the matrix and replace it with either 0 or 1.

You are also given two arrays A and B (of length n and m respectively). After you perform the operations, the matrix should satisfy the following conditions:

1. the number of ones in the i-th row of the matrix should be exactly A_i for every $i \in [1, n]$.
2. the number of ones in the j-th column of the matrix should be exactly B_j for every $j \in [1, m]$.

Calculate the minimum number of operations you have to perform.

Input

The first line contains two integers n and m ($2 \le n, m \le 50$).

Then n lines follow. The i-th of them contains m integers $a_{i,1}, a_{i,2}, \dots, a_{i,m}$ ($0 \le a_{i,j} \le 1$).

The next line contains n integers $A_1, A_2, \dots, A_n$ ($0 \le A_i \le m$).

The next line contains m integers $B_1, B_2, \dots, B_m$ ($0 \le B_i \le n$).

Output

Print one integer — the minimum number of operations you have to perform, or `-1` if it is impossible.

Standard Input	Standard Output

3 3
0 0 0
0 0 0
0 0 0
1 1 1
1 1 1
	3

3 3
1 1 1
1 1 1
1 1 1
3 2 1
1 2 3
	3

2 2
0 0
0 0
1 2
0 1
	-1

,

F. Palindromic Problem

```
Input file:     standard input
Output file:    standard output
Time limit:     5 seconds
Memory limit:   512 megabytes
```

You are given a string s of length n, consisting of lowercase Latin letters.

You are allowed to replace at most one character in the string with an arbitrary lowercase Latin letter.

Print the lexicographically minimal string that can be obtained from the original string and contains the maximum number of palindromes as substrings. Note that if a palindrome appears more than once as a substring, it is counted the same number of times it appears.

The string a is lexicographically smaller than the string b if and only if one of the following holds:

- a is a prefix of b, but $a \neq b$;
- in the first position where a and b are different, the string a contains a letter that appears earlier in the alphabet than the corresponding letter in b.

Input

The first line contains one integer n ($1 \leq n \leq 3 \cdot 10^5$) — the number of characters in the string.

The second line contains the string s itself, consisting of exactly n lowercase Latin letters.

Output

In the first line, print one integer — the maximum number of palindromic substrings that can be obtained using the operation described in the statement at most once.

In the second line, print the string that can be obtained from s and has the maximum possible number of palindromic substrings. If there are multiple answers, print the lexicographically smallest one.

Standard Input	Standard Output

`5`
`aabaa`
	`15`
	`aaaaa`

`5`
`aaaaa`
	`15`
	`aaaaa`

`4`
`awoo`
	`7`
	`aooo`