Problem A. Subarrays Beauty

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

You are given an array a consisting of n integers. A subarray (l, r) from array a is defined as non-empty sequence of consecutive elements $a_l, a_{l+1}, \ldots, a_r$.

The beauty of a subarray (l, r) is calculated as the `bitwise AND` for all elements in the subarray:

$$Beauty(l, r) = a_l \ \& \ a_{l+1} \ \& \ a_{l+2} \ \& \ \ldots \ \& \ a_r$$

Your task is to calculate the summation of the beauty of all subarrays (l, r) $(1 \le l \le r \le n)$:

$$\sum_{l=1}^{n} \sum_{r=l}^{n} Beauty(l, r)$$

Input

The first line contains an integer T, where T is the number of test cases.

The first line of each test case contains an integer n $(1 \le n \le 10^5)$, where n is the size of the array a.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ $(1 \le a_i \le 10^6)$, giving the array a.

Output

For each test case, print a single line containing the summation of the beauty of all subarrays in the given array.

Example

standard input	standard output

2
3
7 11 9
4
11 9 6 11
	40
	48

Note

As input/output can reach huge size it is recommended to use fast input/output methods: for example, prefer to use `scanf/printf` instead of `cin/cout` in C++, prefer to use `BufferedReader/PrintWriter` instead of `Scanner/System.out` in Java.

A `bitwise AND` takes two equal-length binary representations and performs the logical `AND` operation on each pair of the corresponding bits, by multiplying them. Thus, if both bits in the compared position are 1, the bit in the resulting binary representation is 1 $(1 \times 1 = 1)$; otherwise, the result is 0 $(1 \times 0 = 0$ and $0 \times 0 = 0)$. This operation exists in all modern programming languages, for example in language C++ and Java it is marked as **&**.

In the first test case, the answer is calculated as summation of 6 subarrays as follow:

$$Beauty(1, 1) + Beauty(l, 2) + Beauty(1, 3) + Beauty(2, 2) + Beauty(2, 3) + Beauty(3, 3)$$

$$(7) + (7 \ \& \ 11) + (7 \ \& \ 11 \ \& \ 9) + (11) + (11 \ \& \ 9) + (9) = 40$$

Problem B. Array Reconstructing

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

You are given an array a consisting of n elements $a_1, a_2, \ldots, a_n$. Array a has a special property, which is:

$$a_i = (a_{i-1} + 1) \% m, \text{ for each } i \ (1 < i \le n)$$

You are given the array a with some lost elements from it, each lost element is replaced by -1. Your task is to find all the lost elements again, can you?

Input

The first line contains an integer T, where T is the number of test cases.

The first line of each test case contains two integers n and m $(1 \le n \le 1000)$ $(1 \le m \le 10^9)$, where n is the size of the array, and m is the described modulus in the problem statement.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ $(-1 \le a_i < m)$, giving the array a. If the i^{th} element is lost, then a_i will be -1. Otherwise, a_i will be a non-negative integer less than m.

It is guaranteed that the input is correct, and there is at least one non-lost element in the given array.

Output

For each test case, print a single line containing n integers $a_1, a_2, \ldots, a_n$, giving the array a after finding all the lost elements.

It is guaranteed that an answer exists for the given input.

Example

standard input	standard output

4
5 10
1 2 3 4 5
4 10
7 -1 9 -1
6 7
5 -1 -1 1 2 3
6 10
5 -1 7 -1 9 0
	1 2 3 4 5
	7 8 9 0
	5 6 0 1 2 3
	5 6 7 8 9 0

Note

As input/output can reach huge size it is recommended to use fast input/output methods: for example, prefer to use `scanf/printf` instead of `cin/cout` in C++, prefer to use `BufferedReader/PrintWriter` instead of `Scanner/System.out` in Java.

Problem C. Large Summation

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

You are given an array a consisting of n element $a_1, a_2, \ldots, a_n$. For each element a_i you must find another element a_j $(i \neq j)$, such that the summation of a_i and a_j mod $(10^9 + 7)$ (i.e. $(a_i + a_j)$ % $(10^9 + 7)$) is as maximal as possible.

Can you help judges by solving this hard problem?

Input

The first line contains an integer T, where T is the number of test cases.

The first line of each test contains an integer n $(2 \leq n \leq 10^5)$, where n is the size of the array a.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ $(0 \leq a_i < 10^9 + 7)$, giving the array a.

Output

For each test case, print a single line containing n space separated elements, such that the i^{th} element is the answer to the i^{th} element in the array a (i.e. the maximum value of $(a_i + a_j)$ % $(10^9 + 7)$).

Example

standard input	standard output

2
3
1 2 3
2
1000000000 1000000000
	4 5 5
	999999993 999999993

Note

As input/output can reach huge size it is recommended to use fast input/output methods: for example, prefer to use `scanf/printf` instead of `cin/cout` in C++, prefer to use `BufferedReader/PrintWriter` instead of `Scanner/System.out` in Java.

Problem D. Counting Test

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

Yousef has a string s that is used to build a magical string w by repeating the string s infinitely many times. For example, if $s = aabbb$, then $w = aabbbaabbbaabbbaabbb....$

Mohammad always claims that his memory is strong, and that his ability to count is very high, so Yousef decided to hold a test for Mohammad, in order to know the truth of his claims.

Yousef will give Mohammad q queries, such that each query consisting of two integers l and r, and a lowercase English letter c. The answer of each query is how many times the letter c appears between the l^{th} and r^{th} letters in **string** w.

Mohammad must answer all the queries correctly, in order to proof his claims, but currently he is busy finishing his meal. Can you help Mohammad by answering all queries?

Input

The first line contains an integer T, where T is the number of test cases.

The first line of each test case contains two integers n and q $(1 \le n \le 10^4)$ $(1 \le q \le 10^5)$, where n is the length of string s, and q is the number of queries.

The second line of each test case contains the string s of length n consisting only of lowercase English letters.

Then q lines follow, each line contains two integers l and r and a lowercase English letter c $(1 \le l \le r \le 10^9)$, giving the queries.

Output

For each query, print a single line containing how many times the letter c appears between the l^{th} and r^{th} letters in **string** w.

Example

standard input	standard output
1 8 5 abcabdca 1 8 c 1 15 b 4 9 a 5 25 d 2 7 c	2 4 3 3 2

Note

As input/output can reach huge size it is recommended to use fast input/output methods: for example, prefer to use `scanf/printf` instead of `cin/cout` in C++, prefer to use `BufferedReader/PrintWriter` instead of `Scanner/System.out` in Java.

Problem E. Game of Dice

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

It is always fun to play with dice, here is one interesting game:

You are given n fair dice with 6 faces, the faces does not necessarily have values from 1 to 6 written on them. instead, they can have any values from 1 to 100 written on them.

Suppose you threw the n dice one after another, let us define the result of the throws as the multiplication of the numbers you get from each dice mod $10^9 + 7$.

You task is to find how many different ways you can get the result of the dice throws equal to x.

Input

The first line contains an integer T, where T is the number of test cases.

The first line of each test case contains two integers n and x $(1 \le n \le 14)$ $(0 \le x < 10^9 + 7)$, where n is the number of the dice and x is the required result from the throws.

Then n lines follow, the i^{th} line contains 6 integers $f_{i1}, f_{i2}, \ldots, f_{i6}$ $(1 \le f_{ij} \le 100)$, giving the values of i^{th} dice's faces.

Output

For each test case, print a single line containing how many different ways you can get the result of the dice throws equal to x

Example

standard input	standard output

3
3 9
1 2 3 4 5 6
1 2 3 4 5 6
1 2 3 4 5 6
5 6
1 2 9 9 9 9
1 2 9 9 9 9
1 3 9 9 9 9
1 3 9 9 9 9
1 6 9 9 9 9
5 999999937
100 1 1 1 1 1
100 1 1 1 1 1
100 1 1 1 1 1
100 1 1 1 1 1
100 1 1 1 1 1
	3
	5
	1

Problem F. Strings and Queries

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

You are given a set of n strings such that all characters in the strings are '`a`', '`b`', or '`c`'.

Also, you are given q queries, such that each query consisting of two strings a and b. The answer of each query is the index of the string with the most number of palindrome substrings between strings a and b from the given set.

A `substring` of the string s is a sequence $s_l, s_{l+1}, \ldots, s_r$ for some integers (l, r) such that $(1 \le l \le r \le n)$, where n is the length of the string s.

A `palindrome` is a word, phrase, number, or other sequence of characters which reads the same backward as forward, such as "`madam`" or "`racecar`".

Input

The first line contains an integer T, where T is the number of test cases.

The first line of each test case contains two integers n and q $(1 \le n \le 10^4)$ $(1 \le q \le 10^5)$, where n is the number of strings, and q is the number of queries.

Then n lines follow, each line contains a non-empty string with length no more than 30, giving the strings. All characters in the strings are '`a`', '`b`', or '`c`'. It is guaranteed that all strings are unique. The given strings are numbered from 1 to n.

Then q lines follow, each line contains two strings a and b, giving the queries. It is guaranteed that strings a and b exist in the given set of strings.

Output

For each query, print a single line containing the index of the string with the most number of palindrome substrings between the given strings a and b. If there are more than one answer, print the lowest index.

Example

standard input	standard output
1 5 5 aaaaa abaabc cbbaca abccba abca aaaaa abca cbbaca abccba abaabc abccba abccba abca abaabc abccba	1 4 2 4 2

Note

As input/output can reach huge size it is recommended to use fast input/output methods: for example, prefer to use `scanf/printf` instead of `cin/cout` in C++, prefer to use `BufferedReader/PrintWriter` instead of `Scanner/System.out` in Java.

Problem G. Magical Indices

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

Alaa sometimes feels bored at work, so at such times she starts playing with a beautiful array a consisting of n integers $a_1, a_2, \ldots, a_n$.

Alaa starts counting the number of magical indices in the array a. An index x is said to be magical if it satisfying the following rules:

1. $1 < x < n$
2. $a_y \leq a_x$, for each y $(1 \leq y < x)$.
3. $a_x \leq a_z$, for each z $(x < z \leq n)$.

Can you help Alaa by counting the number of magical indices in the array a.

Input

The first line contains an integer T, where T is the number of test cases.

The first line of each test case contains an integer n $(1 \leq n \leq 10^6)$, where n is the size of the array a.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ $(1 \leq a_i \leq 10^6)$, giving the array a.

Output

For each test case, print a single line containing the number of magical indices in the array a.

Example

standard input	standard output

2
8
2 1 3 4 6 5 7 9
6
4 2 7 9 8 10
	3
	1

Note

As input/output can reach huge size it is recommended to use fast input/output methods: for example, prefer to use `scanf/printf` instead of `cin/cout` in C++, prefer to use `BufferedReader/PrintWriter` instead of `Scanner/System.out` in Java.

Problem H. Corrupted Images

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

Husam has a lot of free time, so he is using this time in repairing corrupted binary images.

A Binary image can be represented as 2-dimensional array consisting of n rows, each of which is divided into m columns. The rows are numbered from 1 to n from top to bottom, and the columns are numbered from 1 to m from left to right. Each cell is identified by a pair (x, y), which means that the cell is located at row x and column y. The possible values in the binary images are either zero or one.

A binary image is good if all cells in the first row, last row, first column, and last column are ones. Otherwise, the binary image is corrupted. If the binary image is corrupted, Husam will fix it.

Husam wants to fix the image with the minimum number of moves, such that in each move he can swap the values at two different cells.

Can you help Husam by calculating the minimum number of required moves to fix the image?

Input

The first line contains an integer T, where T is the number of test cases.

The first line of each test case contains two integers n and m ($3 \leq n, m \leq 50$), where n is the number of rows in the binary image, and m is the number of columns in each row.

Then n lines follow, each line contains m characters, giving the binary image. All values in the binary image are either zero or one.

Output

For each test case, print a single line containing -1 if Husam cannot fix the binary image. Otherwise, print the minimum number of required moves to fix the image.

Example

standard input	standard output

3
3 3
111
101
111
4 4
1111
0111
1000
1111
4 5
10101
01010
10101
01010
	0
	2
	-1

Note

In the first test case, the image is good. In the second test case, Husam needs to swap the values of cells (2, 1) and (2, 2), and swap the values of cells (2, 3) and (3, 4), in order to fix the image.

Problem I. The Crazy Jumper

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

Universal Studios Theme Parks announced a new game named `The Crazy Jumper`, that is targeted problem solving fans.

In this game, there are n large boxes numbered from 1 to n from left to right, such that the i^{th} box $(1 < i \leq n)$ is located to the right of the $(i-1)^{th}$ box. Each box has a color, such that c_i is the color of the i^{th} box.

The goal of the game is to reach the n^{th} box starting from the 1^{st} box, by jumping between the boxes. The player can do one of the following jumps:

1. Jump one box to the right.
2. If the player stands at a box of color x, he/she can jump to the closest box of color x that is to the right of him, if such box exist.

Ziad will be the first one to play the game, but he is not sure whether the game is amusing or not, so he wants to finish it with the minimum number of jumps. Can you help Ziad by telling him what is the minimum number of required jumps to reach the n^{th} box stating from the 1^{st} box?

Input

The first line contains an integer T, where T is the number of test cases.

The first line of each test case contains an integer n $(1 \leq n \leq 2 \times 10^5)$, where n is the number of the boxes in the game.

The second line of each test case contains n integers $c_1, c_2, \ldots, c_n$ $(1 \leq c_i \leq 2 \times 10^5)$, where c_i is the color of the i^{th} box.

Output

For each test case, print a single line containing the minimum number of required jumps to reach the n^{th} box starting from the 1^{st} box.

Example

standard input	standard output

3
6
9 2 4 7 1 5
5
1 2 1 1 4
6
1 2 3 1 3 2
	5
	3
	2

Note

As input/output can reach huge size it is recommended to use fast input/output methods: for example, prefer to use `scanf/printf` instead of `cin/cout` in C++, prefer to use `BufferedReader/PrintWriter` instead of `Scanner/System.out` in Java.

Problem J. The Hell Boy

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

Since the problem set was hard, here is an easy task for you to solve.

You are given an array a consisting of n integers, and your task is to calculate the summation of the multiplication of all subsets of array a. (See the note for more clarifications)

A subset of an array a is defined as a set of elements that can be obtained by deleting zero or more elements from the original array a.

Input

The first line contains an integer T, where T is the number of test cases.

The first line of each test case contains an integer n $(1 \leq n \leq 10^5)$, where n is the size of array a.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ $(1 \leq a_i \leq 10^6)$, giving the array a.

Output

For each test case, print a single line containing the summation of the multiplication of all subsets of array a. Since this number may be too large, print the answer modulo $10^9 + 7$.

Example

standard input	standard output

3
3
1 2 3
2
3 5
1
4512
	23
	23
	4512

Note

As input/output can reach huge size it is recommended to use fast input/output methods: for example, prefer to use `scanf/printf` instead of `cin/cout` in C++, prefer to use `BufferedReader/PrintWriter` instead of `Scanner/System.out` in Java.

In the first test case, the array a has 6 subsets, and the answer is calculated as follow:

$$(1) + (2) + (3) + (1 \times 2) + (1 \times 3) + (2 \times 3) + (1 \times 2 \times 3) = 23$$

.

Problem K. Palindromes Building

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

An `anagram` is a word or phrase formed by rearranging the letters of another word or phrase, using all the original letters exactly once, such as "`post`", "`stop`", and "`spot`".

You are given a string s consisting of lowercase English letters. Your task is to count how many distinct anagrams of the string s are palindromes.

A `palindrome` is a word, phrase, number, or other sequence of characters which reads the same backward as forward, such as "`madam`" or "`racecar`".

For example, "`aabb`" has 6 distinct anagrams, which are: "`aabb`", "`abab`", "`abba`", "`baab`", "`baba`", "`bbaa`". Two of the previous anagrams are palindromes, which are: "`abba`", "`baab`".

Input

The first line contains an integer T, where T is the number of test cases.

The first line of each test case contains an integer n ($1 \le n \le 20$), where n is the length of the string s.

The second line of each test contains a string s of length n consisting of lowercase English letters.

Output

For each test case, print a single line containing how many distinct anagrams of the string s are palindromes.

Example

standard input	standard output
5 4 aabb 6 ababbc 6 abaaba 12 babacbcbcaca 14 aaaabbcaaaabbc	2 0 3 90 105