Information

Memory limit

Here is the translation:

- In problem A: 256 megabytes
- In problem B: 1024 megabytes
- In problem C: 256 megabytes
- In problem D: 256 megabytes
- In problem E: 1024 megabytes

Source code limit

The size of each solution source code can't exceed 256 KiB.

Submissions limit

You can submit at most 50 solutions for each problem.

You can submit a solution to each task at most once per 30 seconds. This restriction does not apply in the last 15 minutes of the contest round.

Scoring

Each problem consists of several subtasks. The subtask score is awarded if all tests in the subtask are passed.

The number of points scored for the problem is the total number of points scored on each of its subtasks. The score for the subtask is the maximum number of points earned for this subtask among all the solutions submitted.

Feedback

To get feedback for your solution, go to "Runs" tab in PCMS2 Web Client and use "View Feedback" link. In each problem of the contest you will see the score for each subtask, or the verdict for the first failed test.

Scoreboard

The contestants' scoreboard is available during the contest. Use "Monitor" link in PCMS2 Web Client to access the scoreboard. The standings provided in PCMS2 Web Client are not final.

Problem A. OGD

Time limit: 1 second

Cat Miston has grown tired of powers of two and the number 2 in general. However, he still likes GCD (greatest common divisor). So he came up with OGD — the odd greatest divisor.

The cat knows a thing or two about math. So he can find the OGD of a number. Therefore, help him answer a more complex question: what is the sum of OGD of numbers from l to r? To simplify your task, calculate this sum modulo $10^9 + 9$.

Input

The first line contains an integer t — the number of questions ($1 \le t \le 10^5$).

In the i-th of the following t lines, two integers l_i and r_i are given ($1 \le l_i \le r_i \le 10^{18}$).

Output

Output t numbers (one per line) — the answers to the questions.

Scoring

Subtask	Points	Additional constraints		Required subgroups
		t	l_i and r_i	
1	10	$t = 1$	$l_i = r_i \le 10^6$	
2	10	$t = 1$	$l_i = r_i \le 10^{12}$	1
3	20	$t = 1$	$r_i - l_i \le 5 \cdot 10^6$	1, 2
4	20	$t = 1$	$r_i - l_i \le 10^8$	1, 2, 3
5	10	–	$r_i \le 10^5$	
6	5	–	$r_i \le 10^9$	1, 5
7	25	–	–	1–6

Example

standard input	standard output
2 1 13 1000001 1001499	63 639972

Explanation

The answer to the first question is OGD(1) + OGD(2) + OGD(3) + ... + OGD(13) = 1 + 1 + 3 + 1 + 5 + 3 + 7 + 1 + 9 + 5 + 11 + 3 + 13 = 63

Problem B. Total Ambiguity

Time limit: 2 seconds

We call an array `ambiguous` if there exists **strictly more** than one way to split it into two non-empty subarrays such that both of these subarrays are permutations. Note that each element of the array must belong to exactly one subarray of the split, and each subarray must consist of consecutive elements of the original array.

For example, the array $[1, 2, 1]$ is `ambiguous`; it can be split into $[1] + [2, 1]$ as well as $[1, 2] + [1]$.

You are given an array $a_1, a_2, \ldots, a_n$. In one operation, you can change any element of the array to any integer from 1 to n. What is the minimum number of operations required to make the array $a_1, a_2, \ldots, a_n$ ambiguous?

Recall that an array $p_1, p_2, \ldots, p_n$ is called a permutation if for each x from 1 to n, there exists exactly one i such that $p_i = x$.

Input

The first line of input contains an integer n ($3 \le n \le 10^7$) — the length of the array a and an integer f, which describes the format in which the array a will be given.

If $f = 1$, the format is standard, and then:

The second line of input contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le n$) — the elements of the array a.

If $f = 2$, the format is as follows:

The second line of input contains an integer m ($3 \le m \le \min(n, 2 \cdot 10^5)$) and three integers x, y, z ($0 \le x, y, z < n$).

The third line of input contains m integers $b_1, b_2, \ldots, b_m$ ($1 \le b_i \le n$).

The values $a_1, a_2, \ldots, a_n$ — the array for which you need to find the answer to the problem, are calculated as follows:

- For $1 \le i \le m$: $a_i = b_i$
- For $m < i \le n$: $a_i = 1 + (x \cdot a_{i-2} + y \cdot a_{i-1} + z) \bmod n$

Output

In a single line, output one number — the answer to the problem.

Scoring

Subtask	Points	Additional Constraints				Required Subgroups
		n	a	f	Additional	
1	15	$n \le 7$	–	$f = 1$	–	
2	7	$n \le 2000$	–	$f = 1$	The answer does not exceed 1	
3	21	$n \le 2000$	–	$f = 1$	–	1,2
4	4	$n \le 2 \cdot 10^5$	$a_i = i$	$f = 1$	–	
5	4	$n \le 2 \cdot 10^5$	$a_i = 1$	$f = 1$	–	
6	15	$n \le 2 \cdot 10^5$	–	$f = 1$	The answer does not exceed 1	2
7	18	$n \le 2 \cdot 10^5$	–	$f = 1$	–	1-6
8	16	–	–	$f = 2$	–	

Examples

standard input	standard output
7 1 2 2 1 4 3 7 7	3
4 1 1 2 2 1	1
3 1 1 1 1	1
3 1 1 2 3	1
15 2 6 12 5 11 10 9 8 7 1 1	6

Note

In the first test case, the following three operations can be performed:

- Replace the first element of the array with 3
- Replace the sixth element of the array with 2
- Replace the seventh element of the array with 1

After the operations, the array looks like this: $[3, 2, 1, 4, 3, 2, 1]$, and it is easy to see that this array is ambiguous.

Problem C. Elephant Filimon and the Very Important Message

Time limit: 2 seconds

Once, Elephant Filimon wrote a Very Important Message consisting of letters from the Russian alphabet on a piece of paper. After that, he encrypted it as follows: instead of each letter, he wrote its ordinal number in the alphabet, thus obtaining an array of numbers from 1 to 33.

Many years later, Elephant Filimon found the encrypted Very Important Message again. Possessing a magnificent memory, he recalled what was what. Deciding that his old cipher was too weak, he wanted to encrypt the message even more strongly, and for this, he wanted to calculate the following tricky characteristic, and he needs your help with this!

Given an array $a_1, a_2, \ldots, a_n$ of numbers from 1 to 33.

In one operation, you can combine two adjacent elements of the array into one element equal to the sum of the combined elements. However, the operation can only be performed if the new element is a **prime** number.

For example, from the array $[1, 2, 3, 4, 5]$, you can obtain the arrays $[3, 3, 4, 5], [1, 5, 4, 5], [1, 2, 7, 5]$ in one operation. However, it is **not possible** to obtain the array $[1, 2, 3, 9]$, since 9 is not a prime number.

How many different arrays can be obtained after applying some number of operations (possibly zero)? If one array can be obtained in several ways, it is still counted only once. Since the answer can be quite large, find it modulo 333 333 333.

Input

The first line of input contains one integer n ($1 \le n \le 333\,333$) — the length of the array a.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 33$) — the elements of the array a.

Output

In a single line, output the answer to the problem — the number of arrays that can be reached through operations, modulo 333 333 333.

Scoring

Subtask	Points	Additional constraints		Necessary Groups
		n	a	
1	12	$n \le 10$	–	–
2	12	$n \le 18$	–	1
3	13	$n \le 500$	–	1-2
4	11	$n \le 5000$	$a_i > 1$	–
5	10	$n \le 5000$	–	1-4
6	11	$n \le 10^5$	$a_i > 1$	4
7	10	$n \le 10^5$	–	1-6
8	12	–	$a_i = 1$	–
9	9	–	–	1-8

Examples

standard input	standard output
1 1	1
5 1 2 3 4 5	5
3 33 33 33	1
9 1 1 1 1 1 1 1 1 1	180
11 19 13 16 15 22 10 13 10 14 16 15	80

Problem D. Caesar's Palindrome

Time limit: 1.5 seconds

Percy, a shark in the world of programming, became interested in ciphers. Recently, he read an article about the Caesar cipher, where all letters are cyclically shifted backward by a fixed step d. For example, with $d = 1$, the letter "b" becomes "a", and the letter "a" becomes "z". If the alphabet consists of the first k letters (for example, $k = 26$ for the Latin alphabet), the shift occurs within these k symbols. For instance, with $k = 4$, the alphabet consists of the characters `a`, `b`, `c`, `d`. In this case:

- With $d = 1$: `b` $\to$ `a`, `a` $\to$ `d`.
- With $d = 2$: `c` $\to$ `a`, `a` $\to$ `c`.

At this time, Spotty gifted Percy a string s consisting of lowercase Latin letters. Percy became curious: if within the segment $[l, r]$ he chooses any substring and shifts all characters in it by step d (as in the Caesar cipher), what is the maximum length of a palindrome substring that can be obtained in the entire string? The step d is chosen by Percy himself. Spotty has q such questions, and an answer is required for each of them.

A palindrome is a string that reads the same forwards and backwards. For example, the strings `abacaba`, `aaaa`, `abba`, `racecar` are palindromes.

Input

Each test consists of several sets of input data. The first line contains a single integer t ($1 \le t \le 10^5$), the number of input data sets. The description of the input data sets follows.

The first line of each input data set contains the string s ($1 \le |s| \le 10^5$) — the string gifted to Percy.

The second line of each input data set contains an integer k ($1 \le k \le 26$) — the size of the alphabet. It is guaranteed that the string s consists only of the first k letters of the Latin alphabet.

The third line of each input data set contains an integer q ($1 \le q \le 10^5$) — the number of queries.

In the next q lines of each input data set, there are two integers l and r ($1 \le l \le r \le |s|$) describing the segment.

It is guaranteed that the sum of $|s|$ and the sum of q across all input data sets does not exceed 10^5.

Output

For each query, output the length of the longest palindrome.

Scoring

Subtask	Points	Additional Constraints					Required Subgroups
		$\|s\|$	q	$\sum \|s\|$	$\sum q$	l and r	
1	5	$\|s\| \le 10$	$q \le 10$	$\sum \|s\| \le 30$	$\sum q \le 30$	–	–
2	8	$\|s\| \le 50$	$q \le 50$	$\sum \|s\| \le 100$	$\sum q \le 100$	–	1
3	13	$\|s\| \le 100$	$q \le 100$	$\sum \|s\| \le 200$	$\sum q \le 200$	–	1, 2
4	23	$\|s\| \le 10^3$	$q \le 10^3$	–	–	–	1–3
5	16	–	–	$\sum \|s\| \le 10^4$	–	$l = r$	–
6	35	–	–	–	–	–	1–5

Example

standard input	standard output
1 abbccacccc 3 10 6 8 7 10 1 5 3 10 5 6 1 5 3 10 5 9 4 7 3 7	7 9 9 9 7 9 9 7 7 7

Problem E. Coins on a Tree

Time limit: 8 seconds

You are given a tree with n vertices, initially, there is exactly one coin in each vertex of the tree.

A total of $n-2$ operations will be performed. In each operation, one of the coins is removed from the tree.

After each operation, find the minimum distance between any two different coins on the tree and the number of pairs of coins that achieve this minimum distance. The distance between coins is defined as the number of edges on the simple path between the vertices containing these coins.

Input

The first line of input contains two integers n and t ($3 \le n \le 5 \cdot 10^5$, $0 \le t \le 1$) — the number of vertices in the tree, and a special parameter that determines how the queries are specified.

Each of the following $n-1$ lines of the input file contains two integers u and v, representing the next edge (u, v) in the tree ($1 \le u, v \le n$).

Each of the following $n-2$ lines contains an integer x'. The vertex number from which the coin is removed in this query is calculated using the formula: $x = x' \oplus (lastans \cdot t)$, where $\oplus$ denotes the bitwise XOR operation. Initially, $lastans = 0$, and after each query, $lastans$ equals the number of pairs from the answer to the previous query ($1 \le x \le n$, it is guaranteed that all values of x are distinct).

Output

Output $n-2$ lines, each containing two integers: the minimum distance and the number of pairs of vertices that achieve this minimum distance.

Scoring

Group	Points	Additional constraints			Required subgroups
		n	t	u and v	
1	11	$n \le 100$	$t = 0$	–	
2	13	$n \le 1000$	$t = 0$	–	1
3	28	$n \le 2 \cdot 10^5$	$t = 0$	–	1,2
4	15	$n \le 2 \cdot 10^5$	–	$u + 1 = v$	
5	16	$n \le 2 \cdot 10^5$	–	–	1-4
6	17	$n \le 5 \cdot 10^5$	–	–	1-5

Example

standard input	standard output
5 0 2 1 2 3 3 4 5 4 2 4 3	1 2 2 2 4 1