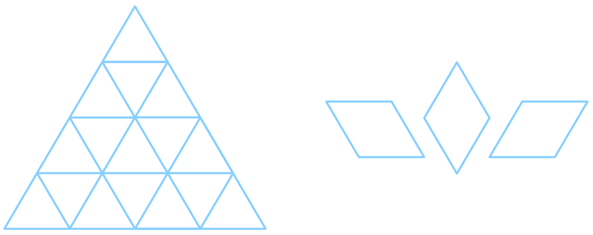
Problem A. Triangle Tiling

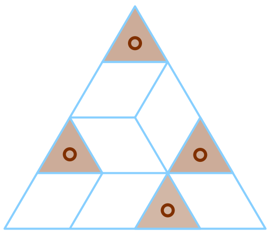
Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	1024 megabytes

Chiaki has an equilateral triangle with side length n. She would like to tile the triangle using unit rhombi with angles equal to $60°$and $120°$.



an equilateral triangle with side length 4
three types of rhombus numbered from 1 to 3 from left to right.

Chiaki soon figures out it's an impossible task. So she cuts out exactly n of the upward triangles from the equilateral triangle. Now, Chiaki would like to know whether it is possible to tile the remaining shape with rhombi. The figure below is an example tiling for $n = 4$.



Input

There are multiple test cases. The first line of input contains an integer T, indicating the number of test cases. For each test case:

The first line contains an integer n $(1 \le n \le 5000)$ – the side length of the equilateral triangle.

The i-th line the following n lines contains a binary string s with length i, where $s_j = 0$ means the j-th upward triangle was cut out.

It is guaranteed that the sum of n over all test cases does not exceed 5000.

Output

For each test case, output a valid tiling. A valid tiling consists of n lines and the i-th line contains i characters. The j-th character should be:

- '-': if the j-th upward triangle was cut out.
- '1': if the j-th upward triangle was coverd by the first type rhombus.
- '2': if the j-th upward triangle was coverd by the second type rhombus.
- '3': if the j-th upward triangle was coverd by the third type rhombus.

If there is no solution, output "Impossible!" (without the quotes) instead.

Example

standard input	standard output
1 4 0 11 010 1101	- 21 -3- 33-1

Problem B. Balanced Binary String

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	512 mebibytes

Chiaki has a string s of length n, consisting of '0', '1' and '?'.

A circular substring $s(i,l)$ of $s = s_1s_2\dots s_n$ is string $\begin{cases} s_is_{i+1}\dots s_{i+l-1} & i+l-1 \le n \\ s_is_{i+1}\dots s_ns_1s_2\dots s_{i+l-1-n} & i+l-1 > n \end{cases}$.

A binary string s of length n is *balanced* if for every two circular substrings $s(i,l)$ and $s(j,l)$ $(1 \le i,j,l \le n)$, the number of 1's in $s(i,l)$ and $s(j,l)$ differ at most by one. For example, 101 and 11010110 are balanced, while 1100 and 1010110110 are not balanced.

Chiaki would like to know the number of ways to replace every '?' to '0' or '1' in her string to make it balanced. Since this number may be very large, you are only asked to calculate it modulo 10^9+7.

Input

There are multiple test cases. The first line of input contains an integer T, indicating the number of test cases. For each test case:

The first line contains a nonempty string s $(1 \le |s| \le 1024)$ consisting of '0', '1' and '?'.

It is guaranteed that the sum of $|s|$ over all test cases does not exceed 1024.

Output

For each test case, output an integer denoting the number of ways.

Example

standard input	standard output
10	2
?	4
??	4
??1	6
???0	11
????1	11
?????0	22
??????1	22
???????0	31
????????1	32
?????????0	

Problem C. Digital Root

Input file:	*standard input*
Output file:	*standard output*
Time limit:	12 seconds
Memory limit:	512 mebibytes

Chiaki has a B-based digital string s of length n. She has prepared m queries for the string.

In the i-th query, she would like to know the number of substring $s_{l..r}$ $(1 \le l \le r \le n)$ of s such that after changing at most one digit in $s_{l..r}$ to some digit in the set A_i, the digital root of $s_{l..r}$ equals to x_i.

We should remind you that a digital root $d(x)$ of the B-based digital string x (x may have some leading zeros) is the sum $s(x)$ of all the digits of this number, if $s(x) \le B - 1$, otherwise it is $d(s(x))$. For example, a digital root of the number 6543_{10} is calculated as follows: $d(6543_{10}) = d(6_{10} + 5_{10} + 4_{10} + 3_{10}) = d(18_{10}) = 9_{10}$, $d(abcd_{16}) = d(2e_{16}) = d(10_{16}) = 1_{16}$.

Note that in this problem we will use the lowercase English letters from 'a' to 'f' to represent the digits with values from 10 to 15.

Input

The first line contains three integers n, m and B $(1 \le n, m \le 2^{20}, 2 \le B \le 16)$ – the length of the string, the number of queries and the base of the number.

The second line contains a B-based digital string s of length n.

Each of the following m lines contains a character x_i and a B-based string a_i $(1 \le |a_i| \le B)$– the expected value of digital root and the set A_i. All characters in a_i are distinct.

Output

For each query, output an integer denoting the number of substrings.

Examples

standard input	standard output

9 2 10
123456789
9 12
8 123456789
	24
	45

5 10 5
01234
0 1
1 1
2 1
3 1
4 1
0 1
1 0
2 0
3 0
4 0
	1
	13
	9
	9
	9
	1
	10
	9
	10
	6

Problem D. Ternary String Revolution

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

A ternary string is a sequence of digits, where each digit is either 0, 1, or 2.

For a ternary string, Chiaki can perform any of the following operations:

- Replace one occurrence of 00 to 12 or vice versa. For example, 120011 can change to 121211 or 000011.
- Replace one occurrence of 111 to 20 or vice versa. For example, 1112011 can change to 202011 or 11111111.
- Remove one occurrence of 22 or put string 22 on any position (you can also put it at the beginning or the ending of the string). For example, 1221 can change to 11, 221221, 122221 or 122122.
- Remove one occurrence of 012 or put string 012 on any position (you can also put it at the beginning or the ending of the string). For example, 10121 can change to 11, 01210121, 10120121, 10012121, 10101221, 10120121 or 10121012.

Chiaki has a ternary string s of length n and m other ternary strings $t_1, t_2, \ldots, t_m$. For each ternary string t_i, she would like to know the number of pairs (l, r) $(1 \le l \le r \le n)$ such that the substring $s_{l..r}$ can become t_i after performing several above operations.

Input

There are multiple test cases. The first line of input contains an integer T, indicating the number of test cases. For each test case:

The first line contains two integers n and m $(1 \le n, m \le 10^6)$ – the length of s and the number of other ternary strings.

The second line contains a ternary string s of length n.

Each of the next m lines contains a ternary string t_i $(1 \le |t_i| \le 10^6)$.

It is guaranteed that the sum of the length of all strings over all test cases does not exceed 2×10^6.

Output

For each test cases, output m lines, where the i-th line contains an integer denoting the answer for ternary string t_i.

Example

standard input	standard output

2
11 4
01021001020
0
1
2
012
6 3
012210
0
1
2
	6
	3
	4
	0
	2
	4
	4

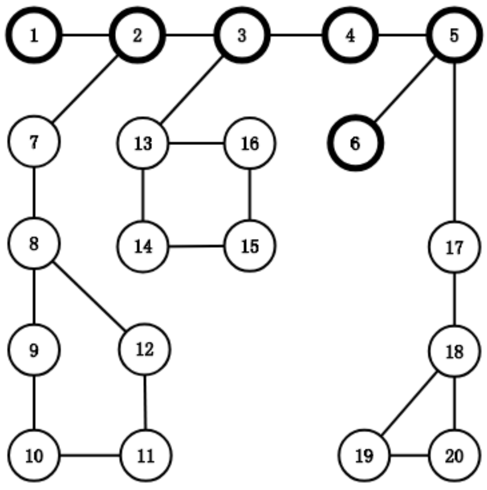
Problem E. Chiaki Chain Counting

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1.5 seconds
Memory limit:	256 mebibytes

An ordinary chain is a graph consisting of sequential (at least two) vertices. Every two adjacent vertices are connected by an edge. The k-th order Chiaki Chain looks slightly different from a chain. There are k sub-chains of various lengths extended from k different vertices on the main chain. At the other side of each sub-chain, there is a simple cycle of length $3, 4, \ldots, k+2$ respectively. There is no useless vertices or edges on the k-th order Chiaki Chain.

Note that the main chain and the sub-chains should consist of at least two vertices.

The following image corresponds to the a 3-rd order Chiaki Chain with 20 vertices and 22 edges:



Given n, m and k, Chiaki would like to know the number of labelled k-th order Chiaki Chain with n vertices and m edges. Since this number may be very large, you are only asked to calculate it modulo $10^9 + 7$.

Input

There are multiple test cases. The first line of the input contains an integer T $(1 \le T \le 10^5)$, indicating the number of test cases. For each test case:

The first line contains three integers n, m and k $(1 \le n, m, k \le 10^6)$ — the number of vertices and the number of edges in the graph and the order of Chiaki Chain.

Output

For each test case, output an integer denoting the answer.

Example

standard input	standard output

4
1 1 1
3 3 1
4 4 1
5 5 1
	0
	0
	0
	60

Problem F. Subset Sum

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	512 mebibytes

Chiaki has n integers $a_1, a_2, \ldots, a_n$ and another integer c, and she would like to choose a subset of the n integers whose sum does not exceed c. Find the maximum possible sum of the chosen subset.

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \le T \le 2 \times 10^4$), indicating the number of test cases. For each test case:

The first line contains two integers n and c ($1 \le n \le 2 \times 10^4$, $1 \le c \le 10^9$). The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 2 \times 10^4$).

The sum of all n does not exceed 2×10^4.

Output

For each test case, output an integer denoting the answer.

Example

standard input	standard output

3
3 5
2 3 4
3 1
2 3 4
3 1000000000
2 3 4
	5
	0
	9

Problem G. Back and Forth

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

There are n stations and m *directed* roads between them.

One day, Chiaki is going from the s-th station to the t-th station, then back to the s-th station. Doing so, he needs to buy tickets for stations he passes. The price the tickets for the i-th station is p_i. If Chiaki buys a ticket for the i-th station, he can passes the station as many times as he wants. Find the minimum price of tickets to buy.

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \le T \le 200$) indicating the number of test cases. For each test case:

The first line of each test case contains four integers n, m, s and t ($1 \le n \le 200$, $0 \le m \le n \times (n-1)$, $1 \le s, t \le n$). The second line contains n integers $p_1, p_2, \ldots, p_n$ ($1 \le p_i \le 100$). The i-th of the following m lines contains two integers a_i and b_i, which denote a road from the a_i station to the b_i-th station ($1 \le a_i, b_i \le n$).

The sum of all n does not exceed 200.

Output

For each test case, output an integer denoting the answer. Print -1 for no solution.

Example

standard input	standard output

```
3
4 5 1 4
1 1 1 1
1 2
2 3
3 1
4 2
3 4
4 4 1 2
1 1 1 1
1 2
2 3
3 4
4 1
4 8 1 3
1 100 1 1
1 2
2 1
2 3
3 2
1 4
4 1
3 4
4 3
```
	```
	4
	4
	3
	```

Problem H. Stone Game

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

Chiaki finds the following interesting stone game: two players start with two non-empty piles of stones. In each turn, the player can choose a pile with an even number of stones and move half of the stones of this pile to the other pile. The game ends if a player cannot move, or if we reach a previously reached position. In the first case, the player who cannot move loses. In the second case, the game is declared a draw.

Given two positive integers n and m, Chiaki would like to know the number of pairs (a, b) $(1 \le a \le n, 1 \le b \le m)$ such that if initially the two piles have a and b stones respectively, then the first player has a winning strategy, or the game ends with a draw, or the second player has a winning strategy. Since this number may be very large, you are only asked to calculate it modulo $10^9 + 7$.

Input

There are multiple test cases. The first line of the input contains an integer T, indicating the number of test cases. For each test case:

The first line contains a binary string s $(1 \le |s| \le 10^6)$ – the binary representation of n without leading zeros.

The second line contains a binary string t $(1 \le |t| \le 10^6)$ – the binary representation of m without leading zeros.

It is guaranteed that the sum of the length of binary strings in all test cases will not exceed 2×10^6.

Output

For each test case, output three integers: the number of pairs (a, b) such that first player wins, the game ends with a draw or the second player wins, correspondingly.

Example

standard input	standard output

3
111
111
1111
1111
10101010
1001010
	8 24 17
	41 116 68
	2546 6689 3345

Note

For the first sample:

- The pairs when first player wins: $(2, 2)$, $(2, 4)$, $(2, 6)$, $(4, 2)$, $(4, 6)$, $(6, 2)$, $(6, 4)$, $(6, 6)$.
- The pairs when the game ends with draw: $(1, 2)$, $(1, 4)$, $(1, 6)$, $(2, 1)$, $(2, 3)$, $(2, 5)$, $(2, 7)$, $(3, 2)$, $(3, 4)$, $(3, 6)$, $(4, 1)$, $(4, 3)$, $(4, 5)$, $(4, 7)$, $(5, 2)$, $(5, 4)$, $(5, 6)$, $(6, 1)$, $(6, 3)$, $(6, 5)$, $(6, 7)$, $(7, 2)$, $(7, 4)$, $(7, 6)$.
- The pairs when the second player wins: $(1, 1)$, $(1, 3)$, $(1, 5)$, $(1, 7)$, $(3, 1)$, $(3, 3)$, $(3, 5)$, $(3, 7)$, $(4, 4)$, $(5, 1)$, $(5, 3)$, $(5, 5)$, $(5, 7)$, $(7, 1)$, $(7, 3)$, $(7, 5)$, $(7, 7)$.

Problem I. Longest Lyndon Prefix

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

A word w is a lyndon word if and only if it is strictly smaller than all its proper suffixes. For example, `aab` is a lyndon word, while `aa` is not a lyndon word.

Chiaki has a string $s_1 s_2 \dots s_n$ of length n. She would like to know l_i, that is the length of the longest prefix of $s_i s_{i+1} \dots s_n$ which is a lyndon word.

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \le T \le 10^5$), indicating the number of test cases. For each test case:

The first line contains an integer n ($1 \le n \le 10^5$). The second line contains a string $s_1 s_2 \dots s_n$ consists of lowercase characters.

The sum of all n does not exceed 10^5.

Output

For each test case, output n integers denoting $l_1, l_2, \dots, l_n$.

Example

standard input	standard output

3
3
aaa
3
aab
3
cba
	1 1 1
	3 2 1
	1 1 1

Problem J. Program Optimization

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Consider the following C++ code:

```
#include <algorithm>
#include <random>

int query_mex(const int *a, int l, int r);

int simulate(int n, int *a, int q, int k, int s) {
  std::mt19937 gen;
  gen.seed(s);
  int last = 0;
  while (q--) {
    int op = gen() % k;
    int i = (gen() + last) % n;
    if (!op && i) {
      std::swap(a[i - 1], a[i]);
    } else {
      int j = gen() % n;
      last ^= query_mex(a, std::min(i, j), std::max(i, j));
    }
  }
  return last;
}
```

In the program, `query_mex(a, l, r)` returns the minimum non-negative integers which does not occur in $a_l, a_{l+1}, \ldots, a_r$.

Given the value of n, a, q, k and s, find the returned value of the function.

Input

The first line contains four integers n, q, k and s ($1 \le n \le 2 \times 10^5$, $1 \le q \le 10^7$, $1 \le k \le 10^9$, $0 \le s \le 10^9$). The second line contains n distinct integers $a_0, a_1, \ldots, a_{n-1}$ ($0 \le a_i < n$).

Output

Output an integer denoting the returned value.

Example

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

Problem K. Determination

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Chiaki has an $n \times n$ matrix M defined as follows.

1. $M_{i,i} = d_i$ for each $i \in \{1, 2, \ldots, n\}$,
2. $M_{p_i,i} = a_i$, $M_{i,p_i} = b_i$ for each $i \in \{2, 3, \ldots, n\}$,
3. $M_{i,j} = x$, otherwise.

Given the value of d_i, p_i, a_i, b_i and x, find $\det(M)$ modulo $(10^9 + 7)$.

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \le T \le 10^6$), indicating the number of test cases. For each test case:

The first line contains two integers n and x ($1 \le n \le 10^6$, $0 \le x \le 10^9$). The second line contains n integers $d_1, d_2, \ldots, d_n$ ($0 \le d_i \le 10^9$). The i-th of the following $(n-1)$ lines contains three integers $p_{i+1}, a_{i+1}, b_{i+1}$ ($1 \le p_{i+1} \le i$, $0 \le a_{i+1}, b_{i+1} \le 10^9$).

The sum of all n does not exceed 10^6.

Output

For each test case, output an integer denoting the answer.

Example

standard input	standard output

3
1 23333
233
3 1
1 1 1
1 2 3
1 4 5
3 1
2 3 4
1 4 5
2 6 7
	233
	1000000003
	999999923

Problem L. Fraction Reduction

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Chiaki has a fraction $\frac{a}{b}$ (not necessary an irreducible fraction) and can perform the following 2 operations:

- If the current fraction is x, Chiaki can change it to $-\frac{1}{x}$.
- If the current fraction is x, Chiaki can change it to $x + 1$.

Now, Chiaki would like to know the number of minimum operations needed to make $\frac{a}{b}$ become 0. Since this number may be very large, you are only asked to calculate it modulo $10^9 + 7$.

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \le T \le 10^5$), indicating the number of test cases. For each test case:

The first line contains two integers a and b ($-10^{18} \le a \le 10^{18}, 1 \le b \le 10^{18}$), denoting the fraction.

Output

For each test case, output an integer denoting the the number of minimum operations modulo $10^9 + 7$, or -1 if there's no such operations to make $\frac{a}{b}$ become 0.

Example

standard input	standard output

5
0 1
1 1
-1 2
-2 4
8 5
	0
	2
	4
	4
	10

Note

For the 1-st sample, you don't need any operations.

For the 2-nd sample, one possible sequence is: $\frac{1}{1} \to -\frac{1}{1} \to 0$.

For the 3-rd sample, one possible sequence is: $-\frac{1}{2} \to \frac{1}{2} \to -\frac{2}{1} \to -\frac{1}{1} \to 0$.

For the 4-th sample, one possible sequence is: $-\frac{2}{4} \to \frac{2}{4} \to -\frac{4}{2} \to -\frac{2}{2} \to 0$.

For the 5-th sample, one possible sequence is: $\frac{8}{5} \to -\frac{5}{8} \to \frac{3}{8} \to -\frac{8}{3} \to -\frac{5}{3} \to -\frac{2}{3} \to \frac{1}{3} \to -\frac{3}{1} \to -\frac{2}{1} \to -\frac{1}{1} \to 0$.