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 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 $s_1 s_2 \ldots s_n$ of length n. She would like to know l_i, that is the length of the longest prefix of $s_i s_{i+1} \ldots s_n$ 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 $s_1 s_2 \ldots s_n$ consists of lowercase characters.

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

Output

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

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 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: $\frac{8}{5} \to -\frac{5}{8} \to \frac{3}{8} \to -\frac{8}{3} \to -\frac{5}{3} \to -\frac{2}{3} \to \frac{1}{3} \to -\frac{3}{1} \to -\frac{2}{1} \to -\frac{1}{1} \to 0$.

Problem M. Apples

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

In Appleland, there are N cities ($1 \le N \le 5000$), with each city having various trade routes to other cities. In total, there are T trade routes ($0 \le T \le 2.5 \cdot 10^7$) in Appleland. For each trade route between two cities x and y, there is a transportation cost $C(x, y)$ to ship between the cities, where $C(x, y) \ge 0$, $C(x, y) \le 10^4$ and $C(x, y) = C(y, x)$. Out of the N cities, K ($1 \le K \le N$) of these cities have stores with really nice apples that can be purchased on-line. The price for each apple in city x is P_x ($0 \le P_x \le 10^4$).

Find the minimal price to purchase one apple on-line and have it shipped to a particular city D ($1 \le D \le N$) using the cheapest possible trade-route sequence. Notice that it is possible to purchase the apple in city D and thus require no shipping charges.

Input

The first line of input contains one integer N, the number of cities. You can assume the cities are numbered from 1 to N . The second line of input contains T, the number of trade routes. The next T lines each contain 3 integers, x, y, $C(x, y)$, to denote the cost of using the trade route between cities x and y is $C(x, y)$. The next line contains the integer K, the number of cities with a store that sells really nice apples on-line. The next K lines contains two integers, z and P_z , to denote that the cost of a apple in city z is P_z. The last line contains the integer D, the destination city.

Output

Output the minimal total cost of purchasing a apple on-line and shipping it to city D.

Examples

standard input	standard output
3 3 1 2 4 2 3 2 1 3 3 3 1 14 2 8 3 3 1	6

Problem N. Bar Graphs

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

A bar graph is a graph consisting of vertical stacks of blocks ('#') of different height.

You will have to draw the graph given the values it has to represent. You will be given the desired height of the graph in number of blocks, H, and the values of the N bars of the graph $X_1, \ldots, X_N$.

The bar with the maximum value should have height H and the other bars should be scaled proportionally to that a bar has n blocks if and only if the proportional value is more or equal than n, but less than $n + 1$.

Input

Input file consists of one line, beginning with two integers H and N (graph height in lines and number of values to be represented, respectively). Then follow these values — N integers $X_1, \ldots X_N$, atleast one between them is non-zero ($1 \le H \le 100,\ 1 \le N \le 100,\ 0 \le X_i \le 10^4$).

Output

Print requested bar graph (see example for reference).

Example

standard input	standard output

16 7 30 21 39 1 17 16 62	`      #`
	`      #`
	`      #`
	`      #`
	`      #`
	`      #`
	`  #   #`
	`  #   #`
	`  #   #`
	`# #   #`
	`# #   #`
	`###   #`
	`### ###`
	`### ###`
	`### ###`
	`### ###`

Problem O. Caesar Cipher

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

For security reasons, Pooh and Rabbit have decided to encode their messages using a Caesar Cipher.

Each message will consist of upper and lower case Latin characters, along with some punctuation. For each character in the message, sender increases its value by the shift amount, preserving case and wrapping around in case of overflow. For example, with a shift of 1 'a' becomes 'b', 'b' becomes 'c', 'z' becomes 'a', 'K' becomes 'L' etc. Non-letter characters should be output unchanged.

Your tast is to encrypt given message with given shift amount.

Input

Input will consist of the ont integer n — length of shift ($1 \le n \le 10^9$), followed by the contents of the message. Each message will be no longer than 1024 characters, but may contain multiple lines. Each line will consist of characters with codes between 32 and 126, inclusively.

Output

Print encoded message.

Example

standard input	standard output

27
This is @ Test Message!
No eNEmY can read it!
Are you sure that
  Code works?
	Uijt jt @ Uftu Nfttbhf!
	Op fOFnZ dbo sfbe ju!
	Bsf zpv tvsf uibu
	  Dpef xpslt?

Problem P. Diamond Numbers

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

Diana likes interesting numbers like 64. It turns out that 64 is both a square and a cube, since $64 = 8^2$ and $64 = 4^3$. Diana calls these numbers *d*iamond. More formally, number n is called diamond, if $n = k^p$ and $n = l^q$, where k, p, l, q are positive integers, $gcd(p, q) = 1$ and both p and q are greater, than 1.

Write a program that helps Diana figure out how many integers in a given range are diamond.

Input

On the first line of input, you are given an integer a ($1 \le a \le 10^8$). On the second line of input, you are given an integer b such that $a \le b \le 10^8$ — first and last integer in a given range.

Output

The output should be the number of diamond numbers in the range a to b inclusively.

Example

standard input	standard output
1 82	2
101 1001	1

Problem Q. Easy Puzzle

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

Given a grid of R rows of lights, with each row containing L lights. Lights can be in one of two states: on or off. For this question, the topmost row is row R, and the bottom-most row is row 1. Also, beside all rows except the topmost row (row R), there is a button which can be pushed.

Pushing the button beside row k ($1 \le k < R$) will peform an ¡¡exclusive-or¿¿ operation on each light of row k, which is described below. Consider column i in row k, where $1 \le i \le L$. If the lights in column i of row k and column i of row $k+1$ are both the same (i.e., both on, or both off), then pushing the button beside row k will cause the light in column i of row k to be off. If the lights in column i of row k and column i of row $k+1$ are different (i.e., one is on, and the other is off), then pushing the button beside row k will cause the light in column i of row k to be on. An example is shown below, for $L = 4$:

Column numbers	1	2	3	4
Row k + 1	on	on	off	off
Row k before button pushed	on	off	on	off
Row k after button pushed	off	on	on	off

You are told which lights are initially on and which are initially off. You must calculate how many different light patterns are possible for the bottom row by any sequence of button pushes.

Input

The first line of input will contain the integer R, the number of rows ($1 < R < 30$). The second line of input will contain the integer L ($1 \le L < 8$) , the number of lights per row. The next R lines of input will contain L integers, where the integer will either be 0 (to indicate ¡¡off¿¿) or 1 (to indicate ¡¡on¿¿). Pairs of consecutive integers will be separated by one space character. These R lines will be given in top-down order: that is, the third line of input will be the description of row R, the fourth line will be $R-1$, and so on, until the last line describes the bottom row.

Output

Output the number of possible light patterns of the bottom row.

Example

standard input	standard output
4 3 0 0 1 0 1 1 1 0 1 0 0 1	4