Problem A. Hard to Compare

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	**8.5** mebibytes

Please pay attention to the unusual memory limit.

Let $f(n, k, x)$, where $n > k > x \geq 1$, denote the number of integer arrays of length n that contain integers from 1 to x exactly once, contain integers from $x + 1$ to k at least twice, and do not contain any other integers. For example, $f(7, 4, 2) = 840$, as there are 7 ways to place 1, then there are 6 ways to place 2, and there are 20 ways to place 3 and 4 in the five remaining spots such that both 3 and 4 appear at least twice.

You are given integers n and k. Find the 9 largest values among $f(n, k, 1), f(n, k, 2), \ldots, f(n, k, k-1)$, and print their sum modulo $10^9 + 7$.

Input

The input contains one or more test cases. The first line contains the number of test cases t ($1 \leq t \leq 10^6$).

The only line of each test case contains two integers n and k ($10^4 \geq n > k \geq 10$).

Output

For each test case, output one integer: the sum of 9 largest values modulo $10^9 + 7$.

Example

standard input	standard output
3 17 12 88 24 6949 4513	567627977 225618886 360966919

Note

In the first test case, $f(17, 12, 1) = f(17, 12, 2) = \ldots = f(17, 12, 6) = 0$, so the answer is just the sum of the remaining nonzero values.

Problem B. Conditions

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

You are given a list a that contains n different positive integers, a positive integer k, and a positive integer ℓ. Find the smallest integer x that is greater than or equal to ℓ and satisfies **exactly** one condition among the following ones:

- x is in the list a;
- x is **not** divisible by k;
- x contains 7 in its decimal representation.

Input

The first line contains two integers n and k ($1 \le n \le 10^5$, $2 \le k \le 10^5$).

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^{18}$).

The third line contains a single integer ℓ ($1 \le \ell \le 10^{18}$).

Output

Output a single integer, the answer to the problem.

Examples

standard input	*standard output*
5 9 181 182 184 178 81 173	183
1 888 888 888	888

Note

In the sample, the integers $173, 174, 175, 176, 177, 179, 181, 182$ satisfy two conditions. The integer 178 satisfies three conditions. The integer 180 does not fulfill any conditions. Finally, the integer 183 satisfies exactly one condition: it is **not** divisible by 9.

Problem C. Modulo 4

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

Let A be the set of all arithmetic expressions consisting of the digits `0`, `1`, and the bitwise OR operator `|`, starting with `1`, such that there is a `1` immediately after each `|`.

Let B_n be the subset of all expressions from A such that their value is equal to $2^n - 1$ when considering the numbers in the expression in binary.

Let C_n be the subset of all expressions from A containing at least one expression from the set B_n as a suffix. For example, the following expressions are in C_3: `10011111`, `111`, `110|1|11`, `11|11001|1010|101`, and these expressions are not in C_3: `111|1011`, `1`, `10|11|11`, `1100|10|100`.

For given positive integers n and k, find the number of expressions from the set C_n that contain exactly k digits (and an arbitrary number of `|`). As the answer may be very large, output it modulo 4.

Input

The input contains $t \leq 10$ test cases. The value t is given on the first line of input.

The only line in each test case contains two integers n and k ($2 \leq n \leq 10^{12}$, $n + 2 \leq k \leq 2 \cdot 10^{12}$).

Output

For each test case, output the number of expressions from the set C_n containing exactly k digits, modulo 4.

Example

standard input	standard output
4 2 4 5 15 147 10000 60 150	2 0 1 3

Note

Here are 14 expressions from the first example:

`1111`, `1011`, `1|111`, `111|1`, `110|1`, `11|11`, `10|11`, `11|10`, `10|1|1`, `11|1|1`, `1|10|1`, `1|11|1`, `1|1|10`, `1|1|11`

Problem D. Blind Gauss

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

Construct a square matrix with n rows and n columns consisting of nonnegative integers from 0 to 10^{18} such that its determinant is equal to 1 and there are exactly a_i odd numbers in the i-th row for each i from 1 to n, or report that there is no such matrix.

Input

The first line contains a single integer n ($2 \le n \le 50$).

Each of the next n lines contains a single integer a_i ($1 \le a_i \le n$).

Output

If there is no matrix that meets the requirements, output `-1`.

Otherwise, output n lines with n numbers $m_{i,j}$ in each ($0 \le m_{i,j} \le 10^{18}$): the elements of the constructed matrix. If there are multiple solutions, print any one of them.

Examples

standard input	*standard output*
2 1 1	1 0 0 1
2 2 1	1 1 1 2
4 3 3 3 3	1 0 1 1 1 1 1 2 1 1 2 3 0 1 1 3
3 2 2 2	-1
3 3 1 3	-1

Problem E. Eternal Masters

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

This is an interactive problem.

Two players, Red and White, are playing a game according to the following rules (inspired by the rules of "Magic: the Gathering"):

- White has L life total; L is a positive integer.
- Red's goal is to decrease L to zero or below. White's goal is to prevent that.
- Red has n cards in her hand. The i-th card can decrease L by a positive integer r_i.
- White has m cards in her hand. The i-th card can increase L by a positive integer w_i.
- Each card can only be played from hand at most once.
- Players know each other's cards.
- Red's and White's turns alternate; the first turn is Red's turn.
- On each turn, the player whose turn it is can either play a card from her hand (if there is any) or pass.
- There is a zone named "stack", much like a programmer's stack. Initially the stack is empty. Playing a card from the hand **does not** immediately trigger its effect. Instead, it causes the card to be placed on top of the stack. The stack is shared for both players' cards.
- A pass causes the top card of the stack (which, as we can show, is always an opponent's card) to deal its effect. Then the card is removed from the stack and discarded.
- If White's pass causes L to become zero or less, White loses immediately.
- If Red passes while the stack is empty, Red loses immediately.
- It can be shown that every game will eventually end in one of the two ways described above.

Given L, n, r, m, and w, select a player and play for that player against the interactor to win.

Interaction Protocol

The interaction starts with reading the given game state.

The first line contains an integer n ($1 \le n \le 1000$): the number of cards in Red's hand. The next n lines contain n integers $r_1, r_2 \ldots, r_n$ ($1 \le r_i \le 10^6$): the values of Red's cards, one value per line.

The next line contains an integer m ($1 \le m \le 1000$): the number of cards in White's hand. The next m lines contain m integers $w_1, w_2 \ldots, w_m$ ($1 \le w_i \le 10^6$): the values of White's cards, one value per line.

Different cards may have the same value.

The next line contains an integer L ($1 \le L \le 10^6$): the initial life total.

After reading all these values, print a single line containing either a word "`Red`" or "`White`". This denotes the player you will play for.

After that, the game starts with Red's turn.

At your turn, if you want to pass, print "`pass`". If you want to play a card from your hand, print "`play` x", where x is the value of the card you want to play; that card must be available to you.

At the interactor's turn, read a line. This line says either "`pass`" or "`play` x", where x is the value of the card that the interactor has played from the hand.

If the interactor's pass causes you to win, your program should print "`win`" and terminate gracefully.

If your pass causes you to lose, the interactor will print "`win`". After reading that line, terminate your program gracefully to receive the "`Wrong Answer`" verdict.

After printing something, do not forget to output end of line and flush the output. Otherwise, you will get the "`Idleness limit exceeded`" verdict. To flush the output this, use:

- `fflush(stdout)` or `cout.flush()` in C++;
- `System.out.flush()` in Java;
- `flush(output)` in Pascal;
- `stdout.flush()` in Python;
- see documentation for other languages.

Example

standard input	*standard output*
3 6 2 2 1 9 6	
	White
play 2	
	pass
play 2	
	pass
play 6	
	play 9
pass	
	pass
pass	
	win

Problem F. Grand Prix of Array Count

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

An array a of length n is called *funny* if, for every pair of indices (i, j) where $1 \le i, j \le n$, the following condition holds: if $i + j$ is an even number, then $a_{(i+j)/2} = \gcd(a_i, a_j)$. For example, an array $[6, 2, 2, 2, 4]$ is funny.

You are given two positive integers n and k. Find the amount of funny arrays of length n consisting only of integers between 1 and k. As this number may be very large, output it modulo $10^9 + 7$.

Input

The only line contains two integers n and k ($5 \le n \le 10^{12}$, $2 \le k \le 10^{12}$).

Output

Print a single number: the answer to the problem modulo $10^9 + 7$.

Examples

standard input	*standard output*
5 2	4
32 5	32

Note

In the first sample, there are 4 funny arrays: $[1, 1, 1, 1, 1]$, $[1, 1, 1, 1, 2]$, $[2, 1, 1, 1, 1]$, $[2, 2, 2, 2, 2]$.

Problem G. 1 :eye: > 100 :ear:

Input file:	*standard input*
Output file:	*standard output*
Time limit:	6 seconds
Memory limit:	1024 mebibytes

Ever wondered how to generate a "random" non-convex polygon? One way to do it is ifsmirnov's algorithm. You can read about it here: `https://codeforces.com/blog/entry/63058#comment-472683`. It is also explained below:

> Let n be the number of vertices in the polygon. We randomly generate n points $p_1, p_2 \ldots, p_n$ inside the square $[0\text{–}10^5] \times [0\text{–}10^5]$ in this way:
>
> - for each point, we choose x and y from the uniform distribution of integers between 0 and 10^5;
> - if the current point lies on some line formed by two other points among the previous ones, its coordinates are generated again until it doesn't lie on any such line.
>
> After that, we build a minimum spanning tree for these points and traverse that tree in a depth-first order; when we visit a vertex for the first time, we write down its number. This sequence of numbers represents some Hamiltonian cycle over the points.
>
> Next, we draw a segment between each consecutive pair of points in that cycle. While there are at least two intersecting segments, we fix this intersection by swapping the segments' ends: if the intersecting segments are formed by points p_i, p_{i+1} and p_j, p_{j+1}, then erase these segments and draw a segment between p_i and p_j and a segment between p_{i+1} and p_{j+1}. It's believed this procedure will eventually stop.

You can download the generator source code to generate some samples locally. To do that, download the archive via the link `https://tinyurl.com/UC4R6`

There, you will find the files `gen.cpp` and `jngen.h`. Run the following commands:

- `g++ gen.cpp -o gen`
- `./gen -n 1000 <seed>`

The parameter `<seed>` may contain digits, letters, spaces, and some punctuation marks.

The problem itself is as follows. You are given two non-convex polygons; both are generated with ifsmirnov's algorithm. Find the area of their Minkowski sum.

The Minkowski sum of two polygons is defined as follows: if a point (x_1, y_1) lies inside the first polygon or on its boundary, and a point (x_2, y_2) lies inside the second polygon or on its boundary, then the point $(x_1 + x_2, y_1 + y_2)$ belongs to their Minkowski sum.

Input

The first line contains a single integer n ($n = 10^3$): the amount of vertices in the first polygon. The next n lines contain the coordinates x_i, y_i of its points ($0 \le x_i, y_i \le 10^5$) in the order of traversal (clockwise or counter-clockwise).

The next line contains a single integer m ($m = 10^3$): the amount of vertices in the second polygon. The next m lines contain the coordinates x_i, y_i of its points ($0 \le x_i, y_i \le 10^5$) in the order of traversal (clockwise or counter-clockwise).

Each of the polygons is non-convex, doesn't have self-intersections, and doesn't contain any three points lying on the same line.

Output

Output one number, the area of the two polygons' Minkowski sum. Your answer will be considered correct if its relative error does not exceed 10^{-4}.

Example

standard input	*standard output*
`./gen -n 1000 sample`	`38851658799.3`

Note

Just to be sure, the sample starts with:

```
1000
28481 58236
26391 26391
33364 59290
...
```

Problem H. Don't Try This at Home

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

For an integer array b, let's define $f(b)$ as the lexicographically smallest array of the same length that is lexicographically greater than b and its **set** of elements is the same as b's set of elements. If such an array does not exist, $f(b)$ is undefined. For example, $f([2,7,7,5,4]) = [4,2,2,5,7]$. In this problem, "the set of array's elements" means an unordered collection of array's elements, where each element is only considered once regardless of how many times it appears in that array. Arrays $[2,7,7,5,4]$ and $[4,2,2,5,7]$ have the same set of elements $\{2,4,5,7\}$, despite some values appearing a different amount of times in the first and the second array.

Let $f^k(b)$ denote the function f applied k times to b; here, we consider $f(\text{undefined})$ as undefined too. For example, $f^2([2,7,7,5,4]) = f([4,2,2,5,7]) = [4,2,2,7,5]$.

You are given an integer array a of length n. In this problem, the array satisfies an additional constraint: **at least one** integer appears **exactly once** in the array a. Find the smallest positive integer k such that $f^k(a)$ is not undefined and the array $f^k(a)$ satisfies at least one of the following conditions, or report that there is no such k:

- there is an integer that appears only once in a, but at least twice in $f^k(a)$;
- there is an integer that appears at least twice in a, but only once in $f^k(a)$.

As the answer may be very large, find it modulo $10^9 + 7$.

Input

The input contains one or more test cases. The first line contains the number of test cases t ($1 \le t \le 10^5$).

Each test case is given on two lines. The first of these lines contains an integer n ($2 \le n \le 10^5$). The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^6$). There is at least one value that appears exactly once in a.

The sum of n for all test cases does not exceed $6 \cdot 10^5$.

Output

For each test case, output a single line with the answer modulo $10^9 + 7$, or `-1` if there is no answer.

Example

standard input	*standard output*
4 7 4 3 3 2 2 1 1 6 8 8 3 3 5 3 4 10 10 2 9 8 2 4 4 5 5 2 3 7	1 1 -1 3

Problem I. Try This at Home

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

For an integer array b, let's define $f(b)$ as the lexicographically smallest array of the same length that is lexicographically greater than b and its **set** of elements is the same as b's set of elements. If such an array does not exist, $f(b)$ is undefined. For example, $f([3,2,3,2,6,6]) = [3,2,3,3,2,6]$. In this problem, "the set of array's elements" means an unordered collection of array's elements, where each element is only considered once regardless of how many times it appears in that array. Arrays $[3,2,3,2,6,6]$ and $[3,2,3,3,2,6]$ have the same set of elements $\{2,3,6\}$, despite some values appearing a different amount of times in the first and the second array.

Let $f^k(b)$ denote the function f applied k times to b; here, we consider $f(\text{undefined})$ as undefined too. For example, $f^2([3,2,3,2,6,6]) = f([3,2,3,3,2,6]) = [3,2,3,3,3,6]$.

You are given an integer array a of length n. In this problem, the array satisfies an additional constraint: **every** value in the array a appears there **at least twice**. Find the smallest positive integer k such that $f^k(a)$ is not undefined and the array $f^k(a)$ contains some value that appears exactly once in it, or report that there is no such k. As the answer may be very large, find it modulo $10^9 + 7$.

Input

The input contains one or more test cases. The first line contains the number of test cases t ($1 \le t \le 10^5$).

Each test case is given on two lines. The first of these lines contains an integer n ($2 \le n \le 10^5$). The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^6$). Every value in the array a appears there at least twice.

The sum of n for all test cases does not exceed $6 \cdot 10^5$.

Output

For each test case, output a single line with the answer modulo $10^9 + 7$, or `-1` if there is no answer.

Example

standard input	*standard output*
3 6 3 2 3 2 6 6 3 6 6 6 8 1 1 4 3 3 4 1 1	1 -1 9

Problem J. Pizza Restaurant

Input file: `standard input`
Output file: `standard output`
Time limit: 2 seconds
Memory limit: 1024 mebibytes

You are given n strings $s_1, s_2, \ldots, s_n$. Find any two **different** indices x and y and a **positive** integer k such that the string $s_x \underbrace{s_y s_y \ldots s_y}_{k \text{ times}}$ is a palindrome with length at most $6 \cdot 10^6$, or report that it is impossible.

Input

The first line contains a single integer n ($2 \le n \le 10^5$).

The next n lines contain n strings $s_1, s_2, \ldots, s_n$, one per line ($1 \le |s_i| < 10^6$). The strings consist of lowercase English letters. The total length of all strings does not exceed 10^6.

Output

If there is no answer, output "`No`" (without quotes).

Otherwise, on the first line, print "`Yes`" (without quotes). On the second line, print three integers x, y, and k ($1 \le x, y \le n$, $x \ne y$, $k \ge 1$, $|s_x| + k \cdot |s_y| \le 6 \cdot 10^6$). If there are multiple solutions, print any one of them.

Examples

standard input	standard output
2 aa aa	Yes 1 2 1
4 a bb bcb cdc	No
2 ap papajoj	Yes 2 1 2

Problem K. Spoiler

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

For a positive integer k and a positive integer f_1, a sequence f is recursively generated according to the following formula for $n \geq 2$:

$$f_n = kn + \left\lceil \frac{f_{n-1}}{n} \right\rceil \cdot n$$

For example, if $k = 4$ and $f_1 = 23$:

$$\begin{aligned} f_1 & & &= 23 \\ f_2 &= 4 \cdot 2 + & 24 &= 32 \\ f_3 &= 4 \cdot 3 + & 33 &= 45 \\ f_4 &= 4 \cdot 4 + & 48 &= 64 \\ f_5 &= 4 \cdot 5 + & 65 &= 85 \\ f_6 &= 4 \cdot 6 + & 90 &= 114 \\ f_7 &= 4 \cdot 7 + & 119 &= 147 \end{aligned}$$

For such a sequence, an index $m \geq 2$ is called the *starting index of interpolation* if there exists a polynomial $P(n)$ such that $P(n) = f_n$ for every $n \geq m$, but $P(m-1) \neq f_{m-1}$. In the example above, 5 is the starting index of interpolation: for every index greater than 4, $f_n = P(n) = 2n^2 + 7n$, but $f_4 = 64 \neq P(4) = 60$.

You are given an integer x. Find any pair of parameters f_1 and k and an index m satisfying the following conditions, or report there are none:

- $1 \leq f_1 \leq 10^{10}$, $1 \leq k \leq 10^5$, $2 \leq m \leq 10^{18}$;
- m is the starting index of interpolation for the sequence generated with these parameters;
- $f_m = x$.

The condition with inequalities is important. In particular, if, for some input, the only triplets (f_1, k, m) which satisfy the last two conditions don't meet the first one, then you should report there is no solution.

Input

Each test contains one or more test cases. The first line contains the number of test cases t ($1 \leq t \leq 100$).

The only line of each test case contains an integer x ($2 \leq x \leq 10^{18}$). There is at most one test case with $x > 10^9$.

Output

For each test case, if there is no answer, print `-1`.

Otherwise, output a line with three integers: f_1, k, m ($1 \leq f_1 \leq 10^{10}$, $1 \leq k \leq 10^5$, $2 \leq m \leq 10^{18}$). If there are multiple solutions, print any one of them.

Example

standard input	*standard output*	*notes*
4 85 7 6 637275712755506	23 4 5 -1 1 2 2 -1	Warning! For the third test case, "2 2 2" is not a right answer because in that case $P(1) = f_1$, thus 2 is not the starting index of interpolation.