Problem A. A+Branding

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

The Inter-Galaxy Programming Championship takes place in three divisions: A, B, and C. The number of problems in each division n is the same (from 11 to 14). Some problems may be present in all divisions, some in two adjacent ones, or only in one. Thus, the total number of problems may be less than $3n$.

The organizers of the final decided to create the statements for some problems in collaboration with the general partner of the championship, the company BrainJets, and in each division, the number of joint problems turned out to be exactly k.

Given n and k, as well as the distribution of problems by rounds, determine the minimum number of problem statements that could have been created in collaboration with the BrainJets.

Input

The first line of input contains one integer n ($11 \le n \le 14$) — the number of problems in each division. The second line contains one integer k ($1 \le k \le n$) — the number of problems in each division whose conditions were created in collaboration with the company MTS.

The third line contains n pairwise distinct numbers in the range from 1 to $5 \cdot 10^5$ — unique problem numbers for division A in the Polyhedra problem preparation system.

The fourth line contains n pairwise distinct numbers in the range from 1 to $5 \cdot 10^5$ — unique problem numbers for division B in the Polyhedra problem preparation system.

The fifth line contains a string of n pairwise distinct numbers in the range from 1 to $5 \cdot 10^5$ — unique problem numbers for division C in the Polyhedra problem preparation system.

The same problem number can:

- either appear only in one of the lines from the third to the fifth;
- or appear in all three lines from the third to the fifth;
- or appear in any pair of lines that includes the fourth line.

This means that the corresponding problem is present only in one division, is present in all three divisions, or is in one of the two pairs of adjacent divisions (A and B or B and C) respectively.

Output

Output one integer — the minimum number of problems whose statements were created in collaboration with BrainJets.

Example

standard input
13 522 575 426 445 772 81 447 629 497 202 775 325 982 784 575 426 445 417 81 447 629 156 932 902 728 537 784 857 426 739 417 81 447 918 156 932 902 728 537
standard output
3

Problem B. Big Deadline

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	1024 mebibytes

Two programmers, Nikoloz and Bakhodir, are working on a very important project. They still have some tasks to complete, and the deadline is approaching! Fortunately, everything is already discussed, the tasks are distributed between Nikoloz and Bakhodir, and the time needed to complete each task is calculated. The only thing remaining is to decide in which order they will complete these tasks. The guys don't like doing many tasks simultaneously, so they decided to order the tasks (each person will order his problems) and then complete them one by one in that order.

Unfortunately, there is a small issue. Some tasks are "developer tasks" and some are "tester tasks". This means that in some tasks the guys are required to add some features to the project and in other tasks they need to test the application they develop. Of course, every developer task (even assigned to another person) needs to be completed strictly before any tester task is completed. Note that a person may start doing a tester task before the completion of the developer tasks: the tester may start the testing process with the existing features (but cannot completely test the application without all planned features).

You are the project manager of this project, and your goal is to watch after the guys so they do not waste any time. This means that immediately after one of the guys completes some task, he needs to start doing some other task assigned to him (unless he has already completed all his tasks, of course). More than that, the guys have to start working immediately and simultaneously. Each programmer is free to choose any order of tasks assigned to him, but the condition about developer and tester tasks should be satisfied.

You wonder how many correct ways for the programmers to order their own tasks are there in total. As this number might be very large, calculate it modulo 998 244 353. Two ways are considered different if and only if there are two tasks assigned to one person such that in the first order the first task is completed before the second task and in the second order the second task is completed before the first one.

Input

The first line contains two integers n and m ($1 \le n, m \le 10^5$) — the number of tasks for Nikoloz and Bakhodir to solve, respectively.

The next n lines contain the description of Nikoloz's tasks. Each line contains an integer t_i ($1 \le t_i \le 10^9$), and a letter c_i ($c_i \in \{\texttt{T}, \texttt{D}\}$). The integer is the time to complete this task, and the letter is "`D`" if this task is a developer task and "`T`" if this task is a tester task.

The final m lines contain the description of Bakhodir's tasks in the same format.

Output

Print one integer — the number of correct ways taken modulo 998 244 353.

Examples

standard input	*standard output*
2 2 8 T 100 T 3 D 5 D	2
2 1 10 D 3 T 1 D	1

Problem C. Certainly Best

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	1024 mebibytes

There are n people. Each two people among them are either friends or not. Friendship is bidirectional.

Each person wants to select best friends — some non-empty subset of their friends. The only condition they want to satisfy is that all sets of best friends among n people must be unique. Note that the "being the best friends" property may not be bidirectional (i.e., it is possible that X is best friend for Y, but Y is not the best friend for X).

You are given all friendships. Find any possible selection of sets of best friends such that the total size of sets of best friends is minimum possible. Or you should state that it is impossible to find such a selection.

Input

The first line contains a single integer t ($1 \le t \le 10^4$) — the number of testcases. Next lines contain descriptions of testcases.

The first line of each testcase contains a single integer n ($2 \le n \le 500$) — the number of people.

Then $n-1$ lines follow, the i-th line contains a string of length $n-i$ consisting of characters 0 and 1. For each j ($i < j \le n$), the $(j-i)$-th character of this string is 1 if i and j are friends, and 0 otherwise.

It is guaranteed that the sum of n^2 does not exceed $2.5 \cdot 10^5$.

Output

Print answers to testcases in the given order.

If it is impossible to select sets of best friends to satisfy the conditions, print a single integer -1.

Otherwise, print n lines. The i-th line should start with s_i ($s_i \ge 1$) — the number of selected best friends for the i-th person. Then s_i different integers $a_{i,1}, \ldots, a_{i,s_i}$ ($1 \le a_{i,j} \le n$, $a_{i,j} \ne i$) in the line should follow — best friends of the i-th person. For each j ($1 \le j \le s_i$) people i and $a_{i,j}$ should be friends.

All sets $\{a_{i,1}, \ldots, a_{i,s_i}\}$ should be different. The sum $\sum_{i=1}^{n} s_i$ should be minimum possible.

If there are multiple possible answers, you should print any.

Example

standard input	*standard output*
2 5 1000 011 10 1 3 11 0	1 2 1 1 1 4 1 3 2 2 4 -1

Problem D. Deep Numbers

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 mebibytes

Consider the positive integer *deep number* if it is not prime and it is impossible to delete some digits from its decimal representation to obtain a prime number. For example, 104 is deep number because neither 0, 1, 4, 10, 14, nor 104 are primes, but 2024 is not because we can delete the first, second, and fourth digits and obtain a prime 2.

Note that the prime numbers are the integers that have exactly two distinct divisors: themselves and one.

Given an integer n, count the number of deep numbers with exactly n decimal digits (without leading zeroes) modulo 998 244 353.

Input

Input contains one integer n ($1 \le n \le 10^6$).

Output

Print one integer — the number of n-digit deep numbers modulo 998 244 353.

Example

standard input	*standard output*
1	5

Note

The one-digit deep numbers, mentioned in the sample, are 1,4,6,8, and 9.

Problem E. Encode The Triangle

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

This is a double-run problem

You need to encode a triangle with a non-zero area and integer sides with a perimeter of p, such that no two sides are equal, into a triangle with a non-zero area and a perimeter of $p - 6$, and then restore the original triangle from the encoded one.

Input

The first line of the input file contains the mode of operation: `encode` if you need to encode triangles, or `decode` if you need to restore the original ones. The second line contains a single integer t ($1 \le t \le 25\,000$) — the number of test cases.

Following are the test cases, one per line.

In encoding mode, each test case contains three integers a, b, and c — the sides of the original triangle. It is guaranteed that $3 \le a + b + c \le 1000$, that a, b, and c are pairwise distinct, and that a triangle with sides a, b, and c exists and has a non-zero area.

In decoding mode, each test case contains three integers a', b', and c' — the sides rearranged in some order, output by your program in encoding mode.

Output

In encoding mode, for each test case, output three integers a', b', c' in arbitrary order, such that a triangle with sides a', b', c' exists and has a non-zero area, and $a' + b' + c' = a + b + c - 6$.

In decoding mode, for each test case, output the three sides of the original triangle. The order of the sides within a single test case is arbitrary.

Examples

standard input	standard output
encode 2 20 24 11 25 36 49	18 18 13 25 30 49
decode 2 18 13 18 25 49 30	11 20 24 25 36 49

Problem F. Funds and Exchange

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	1024 mebibytes

This is an interactive problem.

TradeOn is a proprietary trading firm that provides liquidity for many exchanges with a particular focus on emerging markets. They also provide professional services as a derivatives market maker on different exchanges.

As a developer at TradeOn, you have been tasked with optimizing the connections between exchange servers and fund servers. There are a total of 10^9 exchange servers and 10^9 fund servers, and each exchange server is directly connected to every fund server, resulting in 10^{18} connections. Testers have noted that two connections are performing slower than expected. Typically, the latency between servers is 1ms; however, in two cases, the latency is 2ms. To resolve the issue, testers need your help to identify which connections are slow.

To find the slow connections, you can measure the total latency of certain connections. Specifically, you can select e_ℓ, e_r, f_ℓ, and f_r, run tests with these parameters, and determine the total latency of all connections between exchange servers numbered $e_\ell \le i \le e_r$ and fund servers numbered $f_\ell \le j \le f_r$.

The testers urgently request information about the slow connections, so you may measure the total latency no more than 125 times.

Interaction Protocol

The interaction begins with your program making queries. Each query is the line in the format "? e_ℓ f_ℓ e_r f_r" (without quotes), where $1 \le e_\ell \le e_r \le 10^9$, $1 \le f_\ell \le f_r \le 10^9$, to represent the parameters of the test. In response, the jury program outputs a single integer — the total latency of selected connections. You may make no more than 125 queries in total.

To output your answer, print the line "! e_1 f_1 e_2 f_2" (without quotes), where e_1 and f_1 are the server numbers that make up the first slow connection, and e_2, f_2 are the server numbers of the second slow connection. Connections can be output in any order. Note that outputting the answer does not count as a query.

It is guaranteed that slow connections do not change during the interaction (i.e., the interactor is not adaptive). Remember to print the newline and flush the output buffer after each query and after the answer, or your solution will get `IL` (`Idleness Limit Exceeded`).

Example

standard input	*standard output*
1 2 9 10 4 2	? 1 1 1 1 ? 2 2 2 2 ? 3 4 5 6 ? 4 4 6 6 ? 6 4 6 6 ? 6 5 6 6 ! 2 2 6 4

Problem G. Great Sign

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	1024 mebibytes

On one of the planets in a far, far away galaxy, a numeral system with base b is used. In one of the major cities on this planet, there is a developed network of buses. The routes of the buses are numbered with positive integers.

One day, the Intergalaxy Collegiate Programming Contest (ICPC) Finals was held in this city. The coach of the team from Earth noticed an interesting fact: at the stop near the contest venue, there are x bus route numbers written on the banner, and each of the b digits appears on this banner exactly once.

The coach found that this fact is the great sign of future victory and asked his team to check for given x and b whether such a situation is possible, and if it is, output the b-ary representation of the minimum possible value of the largest number to be written on the banner in this case.

Input

The first line contains one integer x — the number of bus routes whose numbers are displayed on the banner ($1 \le x \le 100$). The second line contains one integer b ($2 \le b \le 100$) — the base of the numeral system.

Output

If the situation is impossible, output -1. Otherwise, output the b-ary representation of the minimum possible value of the largest bus route number displayed on the banner, as a sequence of integers representing the corresponding digits in decimal notation, ordered from the most significant b-ary digit to the least significant. For example, the number $\texttt{CD3}_{16}$ should be output as "12 13 3".

Examples

standard input	*standard output*
10 10	-1
9 10	1 0

Note

In the first example, it is impossible to form 10 positive numbers from the 10 digits of the decimal system in the required way, as 0 is not a positive number. In the second example, the list of routes will be 10, 2, 3, 4, 5, 6, 7, 8, 9.

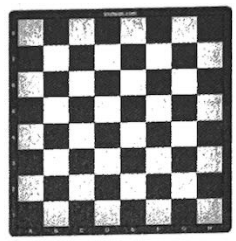
Problem H. Hidden Pieces

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

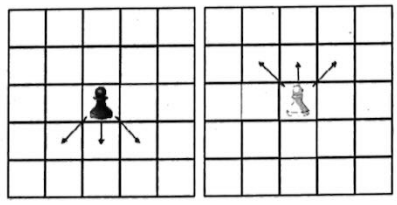
Chess with fog of war («dark chess») differs from regular chess in that players do not see the entire board, but only those squares that their pieces can reach. You are given a setup of pieces, and you need to determine what each player can see.

In this problem, simplified rules of chess apply (including the setup of pieces), which differ from the actual rules of the game:

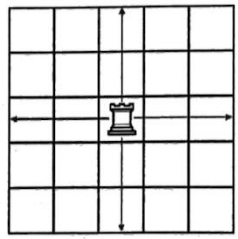
- The game is played on an 8×8 board, with squares colored in black and white such that no two squares sharing a border have the same color. The bottom left square of the board is black.



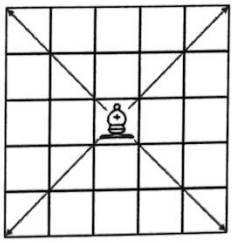
- A square can either be empty or occupied by a single piece. A piece has a color (black or white) and a type (king, queen, knight, bishop, rook, pawn).
- Each side (white and black) must have exactly one king on the board.
- Each side can have from zero to eight pawns, from zero to two rooks, from zero to two knights, from zero to two bishops, and from zero to one queen. If a player has two bishops, one must be on a black square and the other on a white square.
- Pawns cannot be placed on the very bottom and very top ranks.
- Squares occupied by pieces of the same color are always visible to the player playing that color.
- Pieces cannot move off the board.
- A black pawn can move one square down, while a white pawn can move one square up. A pawn can capture a piece only if it is located in a square that shares exactly one point with the current square and is in the direction of the pawn's movement (i.e., "diagonally forward"). A pawn sees both the square it can move to and the squares where it could capture a piece, regardless of whether there is a piece there. No other pawn moves that occur in regular chess are present in this problem.



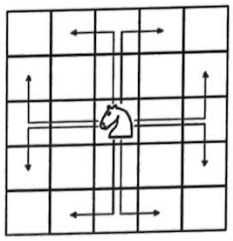
- A rook can move to any free or occupied square of the opposite color in the same row or column, provided there are no other pieces between it and that square.



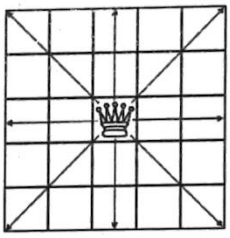
- A bishop can move to any free or occupied square of the opposite color in any of the four diagonal directions, provided there are no other pieces between it and that square.



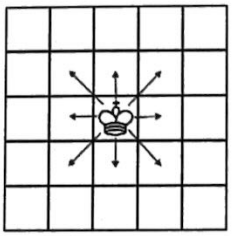
- A knight can move to any free or occupied square of the opposite color that it can reach by moving two squares in one direction and one square in a perpendicular direction.



- A queen can move to any square that a bishop or rook of the same player could move to from the same square where the queen is located.



- A king can move to any free or occupied square of the opposite color that shares at least one point with the current square (i.e., to any of the eight neighboring squares).



- The set of squares visible to a piece is the same as the set of squares it can move to. The set of squares visible to a player is the union of the sets of squares visible to the pieces of the corresponding color.

No other rules (castling, en passant, etc.) are considered in this problem and do not affect the visibility of the board. There are also no restrictions on the mutual placement of pieces (in particular, two kings can indeed stand next to each other in real dark chess).

Input

The input consists of 8 lines, each containing 8 characters. If a character is `.`, then the corresponding square is free. If the character is a lowercase letter, then the piece is white. The letter `k` corresponds to a king, `r` to a rook, `q` to a queen, `n` to a knight, `b` to a bishop, and `p` to a pawn.

If the character is an uppercase letter, then the piece is black. The letter `K` corresponds to a king, `R` to a rook, `Q` to a queen, `N` to a knight, `B` to a bishop, and `P` to a pawn.

Output

Output two blocks of 8 lines, each containing 8 characters: the first block corresponds to the visibility of the board from the perspective of the white pieces (squares that white cannot see should be marked with ?, while the other squares should contain the same pieces as in the input), and the second block corresponds to the visibility from the perspective of the black pieces in the same format.

Examples

standard input	standard output
RNBKQBNR PPPPPPPP .... pppppppp rnbkqbnr	???????? ???????? ???????? ???????? ???????? .... pppppppp rnbkqbnr RNBKQBNR PPPPPPPP .... ???????? ???????? ???????? ???????? ????????
.r...... P...q.p. ....n..P P...b..P .kRp.... ..P..... .Kpp.... ..B.....	.r...... P...q.p? ?.?.n.?? P...b?.? .kRp?.?. ..P..?.? ??pp???. ???????? ??.????? P?.????? ...????P P?.???.P .kRp??.. ..P????? .Kpp???? ..B?????

Problem I. Interesting Travel

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

The Berland company "InterCity Jets" (ICJ) operates flights between n cities. In total, there are m bidirectional airlines in the ICJ schedule. Each airline connects two cities, and it is guaranteed that it is possible to fly between any two cities using the ICJ airlines with zero or more transfers.

If the arrival flight and the departure flight during a transfer are scheduled from different terminals, then to avoid repeated security checks, passengers are transported on commuter buses. In the case of a transfer at the same terminal, the transfer is made via jet bridges.

A well-known travel blogger, LazyBoy, plans to travel on ICJ planes from city A to city B. LazyBoy dislikes using buses, and since all his flights in the travel are sponsored by ICJ, he first wants to minimize the number of inter-terminal transfers. Only if there are multiple options with the minimum number of inter-terminal transfers, he plans to minimize the number of flights.

Given the ICJ flight schedule, as well as the starting and ending cities of the journey, determine the minimum number of times LazyBoy will have to take the bus and the minimum number of flights he must make to get from the starting city to the destination. In the departure and arrival cities, LazyBoy can choose any terminal as he travels by taxi.

Input

The first line of input contains four integers n, m, s, and f ($2 \le n \le 10^5$, $n-1 \le m \le 2 \cdot 10^5$, $1 \le s, f \le n$, $s \ne f$) — the number of cities and the number of airlines in the ICJ schedule, as well as LazyBoy's starting and ending cities, respectively.

Each of the following m lines contains a description of one bidirectional airline and consists of four integers a_i, ta_i, b_i, tb_i ($1 \le a_i, b_i \le n$, $1 \le ta_i, tb_i \le m$, $a_i \ne b_i$) — the first airport number, terminal number of the first airport, the second airport number and terminal number of the second airport for that airline, respectively.

You may assume that it is possible to fly between any two cities using ICJ and that any two airports are directly connected by no more than one ICJ airline.

Output

Print two integers — the minimum number of bus trips LazyBoy will have to take during his journey and the minimum number of flights that can be made with that number of bus trips.

Example

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

Problem J. J322

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

This is an interactive problem

For a large telecommunications operator, the diversification of the equipment used while maintaining the unification of the software is a vital issue. Therefore, the engineers of the company Cell-U-Lari decided to investigate the portability of system software between fundamentally different types of processors.

A module for auto-adaptation, code-named «J322», is planned to be developed, which will recognize the architecture of the microprocessor and generate efficient code for it. Your task within this development is to write a module that distinguishes between two types of input-output stream organizations: stack and queue.

You are given a data structure p, which represents either a stack or a queue. The structure can be empty or contain a certain number of elements (no more than 297).

Your module has exclusive access to the structure and can make up to 300 requests of one of two types: `put` x — to add the number x (an integer from 1 to 1000) to the data structure, or `get` — to retrieve the value from the head of the queue (top of the stack) and remove the corresponding element. If the structure is empty, you receive a special message. After 300 requests (or earlier), you must determine whether it is a stack or a queue.

Interaction Protocol

The interaction begins with your program outputting one of the commands. If you plan to place the element x in the data structure, output the command `put` x ($1 \le x \le 1000$). If you plan to remove an element and find out its value, output the command `get`. The jury program will output a single integer — the value of the retrieved element or -1 if the structure is empty. It is guaranteed that the structure initially contains no more than 297 elements.

As soon as you are ready to determine which data structure is hidden, output `stack` if it is a stack, and `queue` if it is a queue. This action is not counted towards the number of requests.

Note that the interactor is adaptive, meaning the final answer is constructed during the interaction (but always satisfies all available information and the problem conditions).

Example

standard input	standard output
3	put 3 put 4 get queue

Problem K. Know How To Format

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	1024 mebibytes

You are given an expression of the form $a = b + c$, where a is a keyword (a non-empty string of digits and lowercase or uppercase English letters, starting with a letter), and b and c can be either a keyword or a positive integer numeral in decimal notation (a non-empty string of digits from 0 to 9, starting with a non-zero digit).

You need to place apostrophes in all numerals in the expression, if they are present. The first apostrophe is placed between the third rightmost and fourth rightmost digits (if both digits are present), the second between the sixth rightmost and seventh rightmost (if both digits are present), in general, the k-th apostrophe is placed between the $3k$-th rightmost and $3k + 1$-th rightmost digits (if both digits are present).

Input

The input consists of a single line of the form a=b+c, where a is a non-empty string composed of lowercase and uppercase English letters and digits, starting with a letter, and each of b and c is either a non-empty string composed of lowercase and uppercase Latin letters and digits, starting with a letter, or a non-empty string of digits starting with a non-zero digit. The length of each of the strings a, b, and c does not exceed 1000 characters. There are no spaces between the strings and the characters + and =.

Output

Output the expression obtained from the original by inserting apostrophes into the numerals. No other changes (such as inserting spaces) are allowed.

Examples

standard input	*standard output*
mrc2024=mrcq+20241117	mrc2024=mrcq+20'241'117
success=skill+luck	success=skill+luck

Problem L. Laura and Arrays

Input file:	*standard input*
Output file:	*standard output*
Time limit:	5 seconds
Memory limit:	1024 mebibytes

There is an array a of length n consisting of independent uniformly random integers a_i ($1 \le a_i \le 10^9$). Also, there is an array b of length n consisting of independent uniformly random integers b_i ($0 \le b_i \le 1$). Laura wants to erase some (possibly zero) elements from array a, then take the prefix of array b with the matching length, and maximize the resulting dot product of the arrays (i.e. $\sum_{i=1}^{m} a_i \cdot b_i$). Help her to do that.

Input

In the first line, there is one integer n ($1 \le n \le 4 \cdot 10^5$)— the length of the arrays a and b.

In the second line, there are n integers $a_1, \ldots, a_n$ ($1 \le a_i \le 10^9$) — the elements of the array a.

In the third line, there are n integers $b_1, \ldots, b_n$ ($0 \le b_i \le 1$) — the elements of the array b.

It is guaranteed that in all tests, except for the first one (from the examples), all numbers a_i and b_i are generated independently from a uniform distribution over the corresponding ranges.

It is guaranteed that there are no more than 20 tests in total.

Output

Print one number — the maximum possible dot product after erasing some elements from array a.

Examples

standard input	*standard output*
8 1 4 6 5 1 2 3 6 1 0 1 0 1 0 0 1	15
4 843693973 430360361 788359887 531088030 1 1 1 0	2163141890

Note

In the first example, we can erase the first, fifth and sixth elements from a. The result will be equal to the dot product of the arrays $[4, 6, 5, 3, 6]$ and $[1, 0, 1, 0, 1]$ which equals $4 \cdot 1 + 6 \cdot 0 + 5 \cdot 1 + 3 \cdot 0 + 6 \cdot 1 = 15$.

Problem M. Modern Artist and Ancient Legend

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

Ancient Roman philosopher Seneca was the tutor of the future emperor Nero. Nero once asked him how great a number may be. Seneca answered that there are numbers so large that even the wisest citizens of Rome cannot imagine them; for example, numbers that can be created by multiplying usual Roman numbers greater than 1. To avoid the possible ambiguity, Nero decided that only the integers that have a unique representation in this form are counted.

Now a modern artist decided to create an installation dedicated to Nero and Seneca and wants to use some positive integers not exceeding 10^{18} in it. But he wants to be sure that they can be uniquely represented as a product of one or several numbers greater than 1 in Roman numeral representation (more formally, the representations are counted as one representation if they are identical as the multisets, so the representations $4 \cdot 2$ and $2 \cdot 4$ are the same representation).

Any positive integer strictly less than 4000, and only them, has a unique representation in Roman numerals; other rules for writing Roman numerals are well known to all participants, so they will not be provided here for brevity.

Input

The first line of the input contains one integer t ($1 \le t \le 2 \cdot 10^5$) — the number of the testcases.

Each of the following t lines describes one testcase and contains one integer n ($2 \le n \le 10^{18}$) that needs to be checked.

Output

For each request, print 1 on a separate line if the number can be uniquely represented as a product of Roman numerals greater than 1, and 0 otherwise.

Example

standard input	standard output
4 2024 2025 2026 2027	0 0 0 1