Problem A. Internet Addiction

Input file:	stdin
Output file:	stdout
Time limit:	2 seconds
Memory limit:	256 megabytes

One of the most common forms of Internet addiction is non-stop web surfing without any deliberate reason. Theoretically, there is nothing new about it — this habit is hardly any different than surfing TV channels, for example. However, if you cut the TV cable, no one is likely to sit and count the specks of dust on the screen or listen to the static. Certain Internet users, on the other hand, continue to spend a long amount of time "on the web"after losing their Internet connection, studying a web page that was already open, even if it doesn't contain any new information.

The browser displays a page containing a particular line S, consisting of lower-case Latin letters. To search for text on the page, a special window is used, which highlights all the occurrences of a text on the current page on-the-fly (as you are typing the search text). Your task is to calculate the minimal number of actions necessary to perform in order to have each symbol in line S highlighted at least once.

Actions allowed:

1) Typing a single Latin letter.

2) Deleting the last letter typed.

Input

The first line contains a singular number N ($1 \le N \le 10^5$).

The second line defines a string S of the length N, consisting of lower-case Latin letters.

Output

The unknown number of actions necessary.

Examples

stdin	stdout
6 ababab	2
4 abac	4

Note

In the first test example, after entering the string “a” all the “a” symbols are highlighted. After entering “b” (without deleting “a”) all the “ab” strings are highlighted. In this way, in two actions all the symbols in the string were highlighted at least once.

In the second test example, the following sequence of actions is suggested:

1. Enter symbol “a” — all occurrences of “a” are highlighted.

2. Enter symbol “b” — all occurrences of “ab” are highlighted.

3. Delete symbol “b” — all occurrences of “a” are highlighted.

4. Enter symbol “c” — all occurrences of “ac” are highlighted.

All the symbols in the string have been highlighted at least once, in 4 actions.

Problem B. Lottery

Input file:	stdin
Output file:	stdout
Time limit:	1 second
Memory limit:	256 megabytes

In recent years, televised lotteries have been rapidly losing ratings, ceding a large portion of the market to Internet casinos and bookmaker agencies. In an attempt to correct this trend and increase viewers' loyalty, the organizers of one of the lotteries has introduced a new fair and transparent method for determining the winner.

All lottery tickets are numbered sequentially with numbers from L to R (inclusively). Based on the original ticket numbers and a certain random number X, a sequence of encoded numbers A is generated, of which the i-th element is equal to $(L + i - 1) \oplus X$ $(1 \le i \le R - L + 1)$, where $\oplus$ refers to a bitwise addition of binary numerals using the "exclusive or"operation.

The show invites a special guest star, who does not know the value of numeral X, to choose some number K $(1 \le K \le R - L + 1)$. After this, the winning ticket is announced as the one whose encoded number is the K-th element of the sequence A in ascending order.

Input

The only line contains the numbers L, R, X and K $(0 \le L \le R \le 10^{18}, 0 \le X \le 10^{18}, 1 \le K \le R-L+1)$.

Output

The original number of the winning ticket.

Examples

stdin	stdout
6 15 0 7	12
1 20 10 5	14

Problem C. Street Workout

Input file:	stdin
Output file:	stdout
Time limit:	1 second
Memory limit:	256 megabytes

Street Workout is a fairly new term for athletes who perform strength exercises and tricks on everyday pullup bars in neighborhood courtyards or parks. Of course, many young people have trained on pullup bars and performed difficult elements in the past, but with the arise of social networks and video hosting, Street Workout has reached a global scale.

So for those of us who can only do two or three pullups at a time, how do you get started? The answer is simple and nothing new - start with "Stairs". This is a group game where the participants take turns approaching the bar, doing the same number of pullups. The number of pullups increases with each new approach until it reaches a certain maximum value, and then it begins to decrease. The game is finished when the number of pullups per turn reaches zero.

To make the training truly effective, it is important to choose the correct maximum number of pullups to do per turn in the game. Let's look at one variation of the game that allows two participants to train together, even if their skill levels are unequally matched.

Let's assume the first player's skill level is equal to A, while the second player's skill level is equal to B. We will designate the number of pullups per turn for the first player with the number N, and for the second player with the number M. At the first approach, as a mandatory warm-up, both players do one pullup each (i.e. initially $N = M = 1$). After both players have completed the first approach, the following rule will apply: if $N \times A < M \times B$, then the number N increases by one and the first player takes a turn, but if $N \times A > B \times M$, then the number M increases by one and the second player takes a turn. The exception is when one of the players is supposed to take two turns in a row (not counting the very first approach), or when a turn has been completed and the condition $N \times A = B \times M$ is met; in these situations, the rule described above does not apply, and each player repeats each of his turns, performing them in reverse order, until the game ends.

Your job is, knowing the skill level of the players, to find out the total number of pullups the two players will complete during the game.

Input

The first line contains two integers A and B ($1 \le A, B \le 10^9$) - the skill levels of the first and second players, respectively.

Output

The total number of pullups performed by the two players during the game.

Examples

stdin	stdout
3 5	18
1 2	8

Problem D. Routes in Disorder

Input file:	stdin
Output file:	stdout
Time limit:	1 second
Memory limit:	256 megabytes

In the 90's, a huge number of shuttles suddenly appeared on the streets of Russian cities (these are small buses or vans that run frequently along a route but stop wherever passengers ask). These were not only a faster and more convenient type of transportation, but also introduced a sense of chaos to the public transportation system. Many drivers regularly strayed from their designated routes, trying to detour around traffic jams and pick up more passengers, while some of these shuttles traveled around the city without any markings at all. The city administration was not happy with the situation, so it was decided to tighten control on following designated routes and set up a new numbering system for the shuttle.

There are N intersections in the city, numbered from 1 to N, as well as M two-way roads that connect certain pairs of intersections. There is at least one path between each pair of intersections.

It was decided to set up R routes. Assume that a particular route has a sequence of reference intersections $A_1, A_2, \ldots, A_k$. A shuttle should start its route from A_1,then follow the shortest path (by number of intersections crossed) to A_2, from A_2 the same way to A_3, from A_i to A_{i+1} (for $1 \le i \le k-1$), and finish its route at the intersection A_k. In addition, if a pair of intersections A_i and A_{i+1} has more than one shortest route available, the lexicographically smaller one is chosen.

Let's look at two different paths X and Y with an identical number of points k. Path X is lexicographically smaller than path Y, and if i $(1 \le i \le k)$ exists, then $X_i < Y_i$, and for all of j $(1 \le j < i)$ it is true that $X_j = Y_j$.

If we take all the intersections that shuttles cross on the path from A_1 to A_k and write them down in order B_1, B_2, ..., B_s, the route number will be the sum $\sum_{i=1}^{s} iB_i$. If some of the routes end up with the same number, we add a special code to all the numbers except the first one. For example, if we have 4 routes with the number 8172, the first one will keep the number 8172, the second will be 8172#2, the third will be 8172#3, and the fourth will be 8172#4.

Your task is to compute the numbers of all the routes using the plan described above.

Input

The first line contains two numbers N $(2 \le N \le 10^3)$ and M $(1 \le M \le 10^4)$.

The next M lines contain information about roads. Each road is described in a separate line using two numbers u and v $(1 \le u, v \le N, u \ne v)$. There is no more than one road between each pair of intersections. All the roads are two-way streets.

The next line contains the number R $(1 \le R \le 10^5)$ — the number of routes.

Next, the routes are defined in R lines, one per line. The description of the i-th route contains the number k_i $(2 \le k_i \le N)$ - the number of reference intersections on the i-th route and k_i numbers $A_{i,j}$ $(1 \le A_{i,j} \le N)$ — the numbers of the reference intersections on the i-th route in the order they are crossed. The reference intersections of a single route are all different.

The sum of k_i does not exceed $2 \cdot 10^5$.

Output

Output R lines, one per route. In the i-th line, print the number of the i-th route.

Example

stdin	stdout
13 17 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8 5 5 7 1 11 11 12 12 2 11 13 13 10 10 9 9 4 11 10 4 5 1 3 13 12 7 8 1 2 3 11 13 12 4 7 3 1 7 2 2 6 8	644 644#2 211 40

Problem E. Blindfold chess

Input file:	stdin
Output file:	stdout
Time limit:	1 second
Memory limit:	256 mebibytes

To learn to play blindfold chess, besides practicing continuously, you must also do special exercises.

Assume you have a chessboard sized $N \times M$ squares. You must calculate how many squares on the board can be reached from a square $(X; Y)$ by at least one of the chess pieces, provided that the number of moves for this piece is the minimal possible, and is also an odd number.

Input

The first line contains 4 integers N, M, X, Y — the size of the chessboard, and the coordinates of the first square $(1 \le N, M \le 10^9, 1 \le X \le N, 1 \le Y \le M)$.

Output

The unknown quantity of squares.

Examples

stdin	stdout
5 5 3 3	24
8 8 2 3	57

Problem F. The General's Watch

Input file:	stdin
Output file:	stdout
Time limit:	4 seconds
Memory limit:	256 megabytes

A general who had arrived for inspection of an army regiment took a leisurely stroll along the airstrip with the commanding officer, discussing the army's combativity and the global political environment. After returning from their walk, the general discovered that he was missing his prized pocket watch that he was awarded for special service in his youth. And so the commanding officer was threatened with early retirement if his soldiers did not find the watch immediately.

The airstrip is K meters long. There are a total of N soldiers in the regiment, and each of them participates in regular weeding of the grassy airstrip, always assigned to a particular section beginning at A_i and ending at B_i meters from the start of the airstrip.

The commander doesn't have time to select the optimal set of soldiers for a search group, and sending the entire regiment out to search might be too risky of an event. So he called the soldiers to rank single-file, and decided to only send out a group of a few soldiers standing next to each other. Furthermore, each soldier in the search group will only cover the section of the airstrip that he is normally responsible for weeding.

When they have finished searching, the soldiers must set special flags at the beginning and end of each continual section of the airstrip where at least one soldier worked. In this way, if the sections of several soldiers in the search group overlap or border each other, creating a single continuous area, just two flags are put at the edges of this area. If a particular soldier's section does not border or overlap any of the sections of the other soldiers in the search group, two flags are likewise put at the edges of his section.

The flag supply is a bit of a problem, so before sending a group of soldiers out to search, the commander wants to know how many flags the group will need for marking the airstrip.

Input

The first line contains two numbers N ($1 \le N \le 10^4$) and K ($1 \le K \le 10^3$) — the number of soldiers, and the length of the airstrip.

The following N lines contain definitions of the sections of the airstrip that soldiers are responsible for weeding. The $(i+1)$-th line defines the weeding section assigned to the i-th soldier, as two integers A_i and B_i ($0 \le A_i < B_i \le K$).

The next line contains the number M ($1 \le M \le 10^5$) — the number of queries. Each of the following M lines contains a query description formatted as two numbers L_i and R_i ($1 \le L_i \le R_i \le N$) — the numbers of the first and last soldiers in the search group.

Output

Output M lines with query responses: the i-th line should contain a single number — the number of flags that will be needed for marking the airstrip by the search group defined in the i-th query.

Example

stdin	stdout
8 10 2 7 0 3 4 8 3 4 5 6 7 8 4 5 6 7 5 1 2 2 3 1 3 4 6 4 8	2 4 2 6 2

Problem G. Security

Input file:	stdin
Output file:	stdout
Time limit:	1 second
Memory limit:	256 megabytes

Working as front-door security in the student dormitory is nerve-wracking for anyone. All day long, students rush past security, taking offense when they are asked to show ID, and worst of all, constantly letting the door slam!

One of the security officers got tired of losing his temper in vain, and decided to try to contain his frustration. He promised himself that over the course of the day he would not react to the students' behavior until the cumulative volume of the door slamming S either exceeded or equaled the limit to his patience L. As soon as this happened, the security officer would catch up with the student who slammed the door and give him a long lecture. After this, the security officer would calm down again, and the cumulative volume S would be nullified.

With a known volume of A_i for each time the door slams per day, you must calculate the number of lectures given by the security officer.

Input

The first line contains two numbers N ($1 \le N \le 1000$) and L ($1 \le L \le 1000$) — the number of times the door slammed and the limit to the security officer's patience. The second line contains N of the numbers A_i ($1 \le A_i \le 1000$) — the volume of the door slams.

Output

Number of lectures given per day.

Examples

stdin	stdout
6 3 1 3 2 2 3 2	3

Problem H. Numismatists

Input file:	stdin
Output file:	stdout
Time limit:	1 second
Memory limit:	256 megabytes

Numismatists are people who collect coins. Many of them do this as a hobby, trying to catch the spirit of an age gone by, while for others the coins are primarily a way to make a profit.

Most numismatists collect coins belonging to a specific epoch. To make it easier for buyers to find the coins they want, sellers are forced to keep the coins in correct order in the display case, sorting them by ascending order of minting date. The coins often get out of order while they are being examined and purchased, so to solve this problem a special coin-sorting algorithm was developed.

Let N be the number of coins in the display case. In a single operation, the seller can select a group of coins placed in a row in the display case, whose length must be equal to a power of two $(2, 4, 8, \dots)$, and move the oldest coin in the group to the beginning by taking all the coins in the group that were ahead of it and shifting them one position away from the beginning.

For a given number of coins, you must calculate the minimal number of operations necessary to put them in order by ascending minting date according to the algorithm described above, for the worst case scenario.

Input

The first line contains the number of coins in the display case N $(1 \le N \le 10^9)$.

Output

The necessary number of operations.

Example

stdin	stdout
3	3

Problem I. Tram Fans

Input file:	stdin
Output file:	stdout
Time limit:	2 seconds
Memory limit:	256 megabytes

Tram fans is a nickname for people who don't just believe that trams are a green form of transportation that reduce traffic, but who see trams as a true object of adoration, as well as the main attraction of any city. These are the people who organize committees and demonstrations to save the trams in their city, which local officials are trying to shut down, citing their supposed unprofitability. Tram fans spend their vacation traveling around the country or the world in hopes of seeing rare makes of trams and capturing them in photos.

A group of tram fans has arrived in a city that is rumored to have several rare makes of trams. Having procured a map of tram routes, they separate and set out on their search. When one of the tram fans sees a rare tram, he starts to call all his friends, and tells them the final destination of its route.

The city's tramway network is a connected set of N stops and $N - 1$ sections of rails between them. Each tram route is the shortest path between two particular stops. Your task, knowing the origin and destination stops A and B of a rare tram, is to determine the minimal number of stops that a tram fan who is located at stop C must travel in order to get onto the tram's route.

Input

The first line contains the number of stops in the tramway network N ($1 \le N \le 10^5$). The following $N - 1$ lines define number pairs U_i and V_i ($1 \le U_i, V_i \le N$) — the numbers of the stops that connect the i-th section of rails.

The next line defines the number of queries M ($1 \le M \le 10^5$). The following M lines contain query descriptions as three numbers A_i, B_i and C_i ($1 \le A_i, B_i, C_i \le N$), where A_i and B_i are the origin and destination stops of a rare tram, and C_i is the stop where a tram fan is located.

Output

Output M lines containing query responses, one per line. The response to the i-th query is the minimal number of stops on the journey of the tram fan from stop C_i to the route of the tram with origin and destination stops A_i and B_i.

Example

stdin	stdout
6 1 2 6 2 3 1 3 4 3 5 3 1 2 3 6 4 5 4 3 2	1 1 2

Problem J. Entrance Control

Input file:	stdin
Output file:	stdout
Time limit:	1 second
Memory limit:	256 megabytes

At any major event, whether it be a concert or a soccer game, turnstiles are installed at the entrance to the arena for safety reasons. People line up waiting to pass through the turnstiles, and police officers maintain order inside as the stadium fills up.

At a stadium entrance, N turnstiles are installed in a row. At certain intervals, several people are allowed to enter the stadium from a group of turnstiles that are next to each other (one person per turnstile). A group of K turnstiles is considered to be optimally controllable, if a number $p \geq 1$ exists for the group, where for each of the first p turnstiles pairs of fans have different grandstand section numbers on their tickets ($A_i \neq A_j$ for all $i \neq j, 1 \leq i, j \leq p$), and for everyone else ($i > p$) is true: $A_i = A_{i-p}$.

For each turnstile, we know the section number on the ticket of the first fan in line for this turnstile. You must learn how to determine whether a certain group of turnstiles in a row is optimally controllable at the given moment or not.

Input

The first line contains the number of turnstiles N ($1 \leq N \leq 10^5$). The second line contains N integers A_i ($1 \leq A_i \leq 10^5$) — the section numbers on the fans' tickets, one for each turnstile.

The third line contains the number of queries M ($1 \leq M \leq 10^5$). The following M lines contain query definitions, one per line. A query is defined by two numbers L_i and R_i ($1 \leq L_i \leq R_i \leq N$) — the code numbers of the turnstiles at the two ends, defining which turnstiles are included in the i-th group being checked.

Output

Print a line S of M symbols. If the group of turnstiles defined in query i is optimally controllable, then the i-th symbol in line S should be equal to 1, else 0.

Examples

stdin	stdout
10 7 1 2 3 1 2 3 3 1 7 7 1 10 1 4 1 5 2 6 2 7 2 8 8 10	0101101
5 1 1 1 2 1 4 1 3 1 4 1 5 3 4	1001

Problem K. Stolen Dissertation

Input file:	`stdin`
Output file:	`stdout`
Time limit:	2 seconds
Memory limit:	256 megabytes

Certain graduate students are not able to decide what their dissertation research will focus on. When their adviser begins to lose paticence, they have no choice but to go to the university library, pick up the first research they find in their area of study, and, after making some minimal changes, attempt to pass off a large portion of the text as their own work. This option never fails — even if their supervisor rejects most of it due to a lack of scientific novelty, at the very least the introduction and the subject area overview won't have to be written from scratch.

One such graduate student stumbled upon somebody's dissertation devoted to analyzing the Weierstrass sigma function.

Coefficients of the Weierstrass sigma function are given as:

$A_{i,j} = 0$ if $i < 0$ or $j < 0$;

$A_{0,0} = 1$;

$A_{i,j} = 3(i+1) * A_{i+1,j-1} + \frac{16(j+1)}{3} A_{i-2,j+1} - \frac{(3j+2i-1)*(6j+4i-1)}{3} A_{i-1,j}$ in all other cases.

The dissertation was written way back in the 80's, when the performance of modern computers was nothing but a dream. Nonetheless, the author was able to get a few tables.

The first table presents the values of $A_{i,j}$ for $0 \le i, j$ and $i + j \le 5$.

$$\begin{pmatrix} 1 & -1 & -9 & 69 & 321 & 160839 \\ -3 & -18 & 513 & 33588 & 2808945 & \\ -54 & 4968 & 257580 & 20019960 & & \\ 14904 & 502200 & 162100440 & & & \\ 1506600 & 796330440 & & & & \\ 2388991320 & & & & & \end{pmatrix}$$

The second table presents the second power values when expanding $A_{i,j}$ to prime factors for $0 \le i, j$ and $i + j \le 20$.

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 2 & 0 & 1 & 0 & 3 & 0 & 1 & 0 & 2 & 0 & 1 & 0 & 4 & 0 & 1 & 0 & 2 & \\ 1 & 3 & 2 & 3 & 1 & 4 & 3 & 4 & 1 & 3 & 2 & 3 & 1 & 5 & 4 & 5 & 1 & 3 & 2 & & \\ 3 & 3 & 3 & 3 & 4 & 4 & 4 & 4 & 3 & 3 & 3 & 3 & 5 & 5 & 5 & 5 & 3 & 3 & & & \\ 3 & 3 & 5 & 5 & 4 & 4 & 5 & 5 & 3 & 3 & 6 & 6 & 5 & 5 & 6 & 6 & 3 & & & & \\ 3 & 6 & 5 & 6 & 4 & 6 & 5 & 6 & 3 & 7 & 6 & 7 & 5 & 7 & 6 & 7 & & & & & \\ 6 & 7 & 6 & 8 & 6 & 7 & 6 & 10 & 7 & 8 & 7 & 9 & 7 & 8 & 7 & & & & & & \\ 7 & 7 & 8 & 8 & 7 & 7 & 10 & 10 & 8 & 8 & 9 & 9 & 8 & 8 & & & & & & & \\ 7 & 7 & 7 & 7 & 9 & 9 & 9 & 9 & 8 & 8 & 8 & 8 & 9 & & & & & & & & \\ 7 & 8 & 7 & 11 & 9 & 10 & 9 & 11 & 8 & 9 & 8 & 11 & & & & & & & & & \\ 8 & 12 & 11 & 12 & 10 & 12 & 11 & 12 & 9 & 12 & 11 & & & & & & & & & & \\ 12 & 12 & 12 & 12 & 12 & 12 & 12 & 12 & 12 & 12 & & & & & & & & & & & \\ 12 & 12 & 13 & 13 & 12 & 12 & 14 & 14 & 12 & & & & & & & & & & & & \\ 12 & 14 & 13 & 14 & 12 & 15 & 14 & 15 & & & & & & & & & & & & & \\ 14 & 15 & 14 & 17 & 15 & 16 & 15 & & & & & & & & & & & & & & \\ 15 & 15 & 17 & 17 & 16 & 16 & & & & & & & & & & & & & & & \\ 15 & 15 & 15 & 15 & 15 & & & & & & & & & & & & & & & & \\ 15 & 16 & 15 & 17 & & & & & & & & & & & & & & & & & \\ 16 & 18 & 17 & & & & & & & & & & & & & & & & & & \\ 18 & 18 & & & & & & & & & & & & & & & & & & & \\ 18 & & & & & & & & & & & & & & & & & & & & \end{pmatrix}$$

The third table presents the third power values when expanding $A_{i,j}$ to prime factors for $0 \le i, j$ and $i + j \le 20$.

0	0	2	1	1	3	2	4	5	4	4	6	5	5	8	7	8	9	8	8	12
1	2	3	3	3	6	5	5	7	5	6	7	8	8	10	9	9	11	9	12	
3	3	5	4	6	7	7	7	9	7	7	10	9	10	11	11	11	15	13		
4	4	7	6	6	8	7	8	9	9	9	11	10	10	12	11	14	15			
5	7	8	8	8	10	9	9	12	10	11	12	12	12	16	15	15				
8	8	10	9	10	11	12	12	14	12	12	14	13	16	17	17					
9	9	11	10	10	13	12	13	14	13	13	17	16	16	18						
10	11	12	13	13	15	14	14	16	14	17	18	18	18							
12	12	15	14	15	16	16	16	20	18	18	20	19								
13	13	15	14	14	16	15	18	19	18	18	20									
14	15	16	16	16	20	19	19	21	19	20										
16	16	18	17	20	21	21	21	23	21											
17	17	21	20	20	22	21	22	23												
18	21	22	22	22	24	23	23													
22	22	24	23	24	25	26														
23	23	25	24	24	27															
24	25	26	27	27																
26	26	29	28																	
27	27	29																		
28	29																			
30																				

In order to add some «scientific novelty» to his work, the graduate student decided to write a program that will relatively quickly calculate for given n and m the values of power of 2 and power of 3 when expanding $A_{n,m}$ to prime factors.

Input

The input data include several tests.

The first line contains the number of tests $0 < T \le 1000$

Each of the following T lines contains a test description in the form of two integers n and m $(0 \le n, m \le 1000)$.

Output

For each of the tests from the input data, print two numbers on a separate line: the powers of 2-nd and 3. If $A_{n,m}$ is not an integer, you must print the line "`-1 -1`"(without quotation marks).

Example

stdin	stdout
3 2 5 1 1 15 7	5 10 1 2 10 19