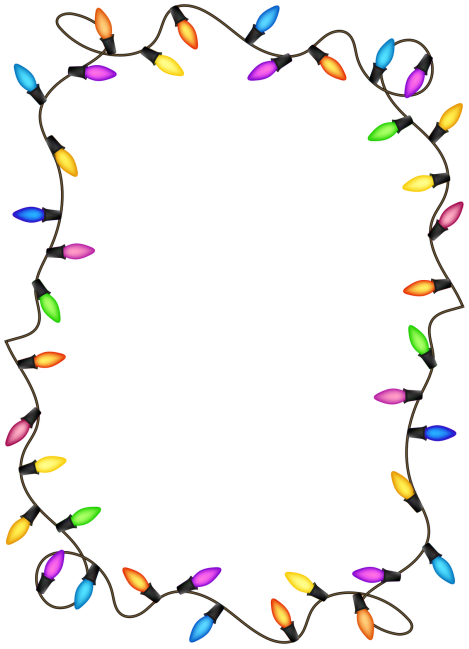
Welcome to Santa's Algorithmic Challenge!

Good luck and have a great time!

Santa and the organizers

Problem A. Christmas Lights

Input file:	`stdin or input.txt`
Output file:	`stdout or output.txt`
Time limit:	2 second(s)
Memory limit:	256 MiB



You recently bought a set of Christmas lights. It consists of n lights arranged in a circle and numbered $1, 2, ..., n$. The expected behavior of the lights is that, when they are turned on, first the light number 1 will be on for one second, then the light number 2 will be on for one second, and so on, until finally the light number n is on for one second, after which the cycle will repeat.

However, you think that the lights might be defective and light up in some other order. You have turned them on some time ago, and now you spent n seconds looking at them. It is indeed true that exactly one light was on at any given time; you have written down the sequence of n numbers of lights that you saw in each one-second interval. Based on this sequence, can you say with confidence that the lights are broken? If yes, after how many seconds could you have concluded that?

Input

The first line of the input contains a single integer n ($3 \le n \le 10^5$) – the number of lights.

The second line of the input contains n space-separated integers $a_1, ..., a_n$ ($1 \le a_i \le n$) – the sequence of numbers that you wrote down.

Output

If you cannot conclude that the lights are broken, print `NO`. Otherwise print `YES` on one line, followed by, on the second line, the minimum number of seconds (that is, the minimum number of items from the beginning of the sequence) after which you can conclude that the lights are broken.

Examples

stdin or input.txt	stdout or output.txt
3 2 3 1	NO
3 1 3 2	YES 2

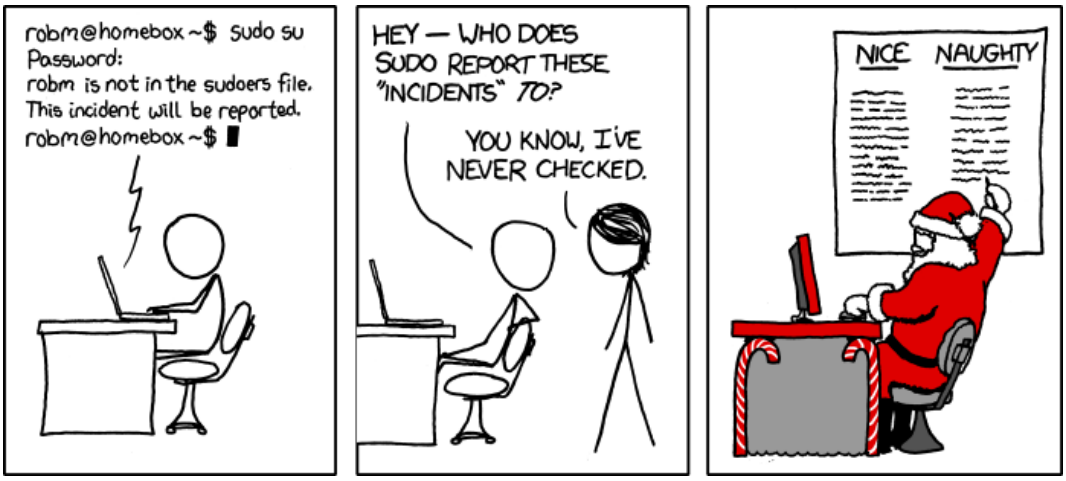
Note

In the second testcase, after seeing `1 3` you already know that the lights are defective, since `1` should have been followed by `2`.

Problem B. Letters to Santa

Input file:	`stdin or input.txt`
Output file:	`stdout or output.txt`
Time limit:	2 second(s)
Memory limit:	256 MiB

Every year before Christmas, EPFL students write letters to Santa Claus, describing the presents that they would like to receive. Santa, of course, receives these letters and reads them. Each student is then classified as naughty or nice. (The naughty group is surprisingly large this year.)



xkcd.com

Santa has decided that the nice students will be exactly those who have bothered to write him a *proper letter* – that is, one where opening and closing parentheses ('(', ')') are matched properly. (Santa just hates it when they are not (and can you blame him?). More precisely, there should exist a way to pair up every opening parenthesis with a closing parenthesis that comes after it in the letter, so that every closing parenthesis is matched to exactly one opening parenthesis (and vice versa). There is one exception to this – Santa actually likes emoticons `:)` and `;)`, so any closing parenthesis that comes immediately after a colon `:` or semicolon `;` should be ignored.

Help Santa by writing a program that will automatically process the letters from students – that is, read a letter and decide whether its author is naughty or nice.

Input

The first and only line of the input will contain the contents of the student's letter. This will be a non-empty string of length at most 500, consisting of the following characters: lowercase English letters, uppercase English letters, and the following special characters: `();:_,!.`.

(Note that it does not contain any whitespace – this might make it easier for you to read the input, by treating it as a single "word". The only whitespace is a single newline character that terminates the input.)

Output

Print one word: `NICE` or `NAUGHTY`.

Examples

stdin or input.txt
`Santa_please,_get_me_an_Algorithms_textbook_(but_a_nice_one)!`
stdout or output.txt
`NICE`

stdin or input.txt
SANTA_GET_ME_TEXTBOOK_(CORMEN!
stdout or output.txt
NAUGHTY

stdin or input.txt
Santa_please,_get_me_an_Algorithms_textbook_(but_a_nice_one_;)_)!
stdout or output.txt
NICE

stdin or input.txt
Santa_please,_get_me_an_Algorithms_textbook_(but_a_nice_one_:)))))_)!
stdout or output.txt
NAUGHTY

stdin or input.txt
Santa_please_get_me_lots_of_parentheses:_((()(()))...
stdout or output.txt
NICE

stdin or input.txt
Santa_please_get_me_lots_of_parentheses:_)(...
stdout or output.txt
NAUGHTY

Problem C. Sleigh

Input file:	`stdin or input.txt`
Output file:	`stdout or output.txt`
Time limit:	2 second(s)
Memory limit:	256 MiB

Santa wants to construct a sleigh to help him deliver all the presents for this Christmas. To this end, he is going to pair up all his reindeer, of which there are two kinds: red-nosed and blue-nosed.

Some of the red-nosed reindeer prefer to be paired with another red-nosed reindeer. There are a of them (including Rudolph). Others prefer to be paired with a blue-nosed reindeer – there are b of them. Similarly, there are c blue-nosed reindeer who would like to be paired with a blue-nosed reindeer, and d blue-nosed reindeer who would like to be paired with a red-nosed reindeer.

The number of all reindeer ($a + b + c + d$) is even, so Santa can divide all of them into pairs. He will do it so as to minimize the number of unhappy reindeer. How many will that be?

Input

The first and only line of input will contain four space-separated integers a, b, c and d ($1 \leq a, b, c, d \leq 100$).

Output

Print one integer – the minimum number of reindeer unhappy with the assignment.

Examples

stdin or input.txt	stdout or output.txt
1 1 1 1	2
2 1 2 1	0

Note

In the first example, one pairing would be to match the a-reindeer with the b-reindeer and the c-reindeer with the d-reindeer, thus creating a red-nosed pair and a blue-nosed pair. That would make the b- and d-reindeer unhappy.

Problem D. Bad Presents

Input file:	`stdin or input.txt`
Output file:	`stdout or output.txt`
Time limit:	2 second(s)
Memory limit:	256 MiB

Santa Claus is preparing a bag of presents for all the students of EPFL that he has classified as naughty.

"*I see you* when you're sleeping. I *know* when you're awake."
– Liam Neeson auditioning as Santa Claus.

The naughty students are not supposed to receive valuable presents. On the contrary, Santa wants to teach them a lesson and has already made a list of n undesirable items (and checked it twice). His signature collection of punitive presents includes old socks, magazines with only advertisements in them, as well as some very large but empty boxes. The only remaining challenge is to produce the n presents. The i-th present on the list requires a_i units of work.

Santa has a machine at his disposal. It operates as follows. First, a queue containing all n items (in order) is created. Every second, the machine takes the first item from the queue and performs b units of work on it. If the present is finished (i.e., at least a_i units of work have been performed on it in total), it is removed from the queue. Otherwise, it is moved to the end of the queue.

The first present that Santa has prepared is particularly vicious, and he just cannot wait to see it ready. How much time will pass before that happens?

Input

The first line of input holds two space-separated integers n and b ($1 \le n \le 10^5$, $1 \le b \le 10^9$). The second line of input holds n space-separated integers a_i ($1 \le a_i \le 10^9$).

Output

Output a single integer – the number of seconds that will pass before item 1 is finished and removed from the queue.

Examples

stdin or input.txt	stdout or output.txt
5 1 3 4 3 1 2	10
4 2 6 2 5 1	7
5 1 1000000000 1000000000 1000000000 1000000000 1000000000	4999999996

Note

In the first example, the machine will perform work on the following items: 1, 2, 3, 4, 5, 1, 2, 3, 5, 1 (at which point item 1 will be ready).

Make sure to use 64-bit integer variables – otherwise you may have overflow issues (see the last example).

Problem E. Downhill

Input file:	`stdin or input.txt`
Output file:	`stdout or output.txt`
Time limit:	5 second(s)
Memory limit:	256 MiB

Santa will be arriving in Lausanne any moment now. Before this happens, though, he needs your help to plan his route inside the city. The reindeer are going to drop Santa off at crossroads number n. From there, he will make his way through the city on foot, walking on the streets and distributing presents. He wants to finish at crossroads number 1 (in Ouchy).

Lausanne is very hilly, Santa's bag is massive, and he hasn't got any slimmer lately... All these factors combined made his walking experience last year rather miserable (notable quotes include "I'm too old for this" and "*something* Switzerland"). This year, Santa simply refuses to walk uphill, that is, from a crossroads at a lower altitude to a crossroads at a higher altitude. Because in Lausanne every crossroads is at a different altitude, this effectively means that every street (connecting two crossroads) can be traveled by Santa only in one direction. In fact, the crossroads are ordered by altitude – the larger the number, the higher the crossroads is. Therefore the sequence of numbers of crossroads visited by Santa on the way from n to 1 will always be decreasing.

These constraints, however, still leave many possible scenarios for Santa's route. Since the route determines the set of streets where children will receive presents, you don't want to make this decision yourself. Instead, you will write down all possible routes and send them to Santa. But how many routes are there?

Since the answer may be very large, output the remainder of this number modulo $5 \cdot 10^8 + 3$.

Input

The first line of input contains two space-separated integers n and m ($2 \le n \le 10^4$, $1 \le m \le 10^5$) – the number of crossroads and the number of streets, respectively.

The next m lines describe the streets of Lausanne. Each line contains two space-separated integers a and b ($1 \le a < b \le n$), which specify a street that Santa can use to walk from crossroads number b to crossroads number a. All given streets are different.

Output

Print the number of paths from n to 1 that Santa can use, modulo $5 \cdot 10^8 + 3$.

Example

stdin or input.txt	stdout or output.txt
8 9 1 2 1 3 2 4 3 4 4 5 4 6 5 8 6 8 4 7	4

Note

In the example, there are 4 paths:

- $8 \to 6 \to 4 \to 2 \to 1$,

- $8 \to 5 \to 4 \to 2 \to 1$,
- $8 \to 6 \to 4 \to 3 \to 1$,
- $8 \to 5 \to 4 \to 3 \to 1$.

Again, beware of overflows!

Problem F. Claussian Distribution

Input file:	`stdin or input.txt`
Output file:	`stdout or output.txt`
Time limit:	2 second(s)
Memory limit:	256 MiB

Santa came up with the best collection of presents ever. They are $2n$ brand-name, factory-new-smelling non-empty strings of lowercase English letters. Sadly, there are only n nice students who deserve such a generous gift, so each of them will receive two strings. Now Santa needs to figure out a way to distribute all the presents – that is, to pair them up appropriately.

He calls such a pairing a *Claussian distribution* if, in each pair, the first string is a prefix of the second. That is, the second one can be obtained by appending letters to the end of the first one.

Can you help Santa find a Claussian distribution?

Input

The first line of the input contains a single integer n – the number of students, each of whom should receive two strings ($1 \leq n \leq 10^5$).

The next $2n$ lines contain one non-empty string each. The strings consist of lowercase English letters. Their total length is at most 500 000.

Output

Print n lines containing two space-separated integers each – the Claussian distribution you propose. Each number from 1 to $2n$ should appear exactly once in your answer. In each pair, the string corresponding to the first number should be a prefix of the string corresponding to the second number.

It is guaranteed that at least one Claussian distribution exists. If there are multiple, output any of them. The order of pairs in your output does not matter.

Example

stdin or input.txt	stdout or output.txt
2 abac abacab aba abaa	3 4 1 2

Note

In the example, the only Claussian distribution is to pair up the first string with the second one, and the third one with the fourth one.

We only guarantee that this problem is solvable in C++ and Java (as opposed to e.g. Python).

Problem G. Getting Out

Input file:	`stdin or input.txt`
Output file:	`stdout or output.txt`
Time limit:	2 second(s)
Memory limit:	64 MiB

Santa has got himself locked in his bathroom (again). There is only one way for him to get out, and that is to invoke a manual override of the bathroom lock. This is done using two buttons on the wall – a red one and a blue one.

Santa begins the procedure with 0 points, and the bathroom will only open if he collects exactly x points by pressing the buttons. A press of the red button awards Santa a_i points if it was the i-th consecutive press of the red button, for $i = 1, 2, ..., n$; if $i > n$, then Santa is awarded a_n points. In other words, the i-th consecutive press of the red button awards Santa $a_{\min(n,i)}$ points. A press of the blue button awards 0 points.

How much time will Santa need to spend in the bathroom? Compute the minimum number of button presses needed for him to get out, or conclude that it is impossible.

Input

The first line of the input contains two space-separated integers n and x ($1 \le n \le 100$, $1 \le x \le 10^6$). The second line of the input contains n space-separated integers $a_1, a_2, ..., a_n$ ($1 \le a_i \le 1000$).

Output

Print a single number – the minimum number of button presses needed to earn exactly x points, or -1 if it is impossible.

Examples

stdin or input.txt	stdout or output.txt
4 14 1 2 3 4	5
5 12 1 4 2 6 3	6
3 8 3 4 2	-1

Note

In the first example, five presses of the red button get Santa $1 + 2 + 3 + 4 + 4 = 14$ points.

Note the 64 MB memory limit in this problem.

We only guarantee that this problem is solvable in C++ and Java (as opposed to e.g. Python).

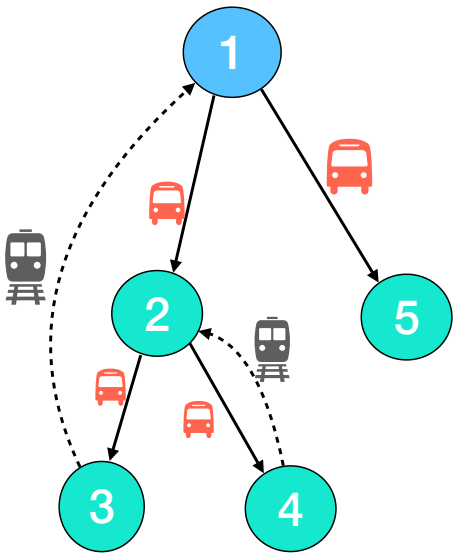
Problem H. Christmas Tree

Input file:	`stdin or input.txt`
Output file:	`stdout or output.txt`
Time limit:	3 second(s)
Memory limit:	256 MiB

Santa's job is not easy – he needs to distribute presents to nice children (and students) all around Lausanne, while at the same time avoiding discovery (especially by those who don't believe he exists – he has learned over time that it is better to avoid such encounters). What's worse, as you cross the Swiss border, reindeer services suddenly become expensive, and so Santa's task today is to come *as close to EPFL as possible* using the public transport system consisting of buses and trains.

The *bus* transport network consists of bidirectional non-stop routes between various pairs of stations in the city. It has an interesting *tree* property – it is possible to travel between any two stations in exactly one way (without using the same route twice). Santa can use the buses for free, but he needs to avoid detection and so he can only travel on the buses in the direction *away from EPFL*.

Fortunately, he can also use trains. Trains also form bidirectional non-stop routes between various pairs of stations in the city. In addition, each train route has the property that it is possible to travel from one of its endpoints to the other using only bus routes in the direction away from EPFL. Santa can use trains in either direction, but it costs him 1 CHF every time.



An illustration of the example testcase.

Santa is going to be dropped off at the station number s and has a budget of k CHF. How close to the EPFL, which is the station number 1, can he get? Answer this question for various settings of s and k.

Input

The first line of the input contains a single integer n – the number of stations ($2 \leq n \leq 200\,000$). The next $n-1$ lines contain two space-separated integers a_i and b_i each ($1 \leq a_i, b_i \leq n$, $a_i \neq b_i$) – the numbers of stations connected by a bus route.

The next line contains a single integer m – the number of train routes ($0 \leq m \leq 200\,000$). The next m lines contain two space-separated integers u_i and v_i each ($1 \leq u_i, v_i \leq n$, $u_i \neq v_i$). It is guaranteed that one can travel from v_i to u_i using only bus routes in the direction away from the EPFL.

The next line contains a single integer q – the number of scenarios to consider ($0 \leq q \leq 200\,000$). The next q lines contain two space-separated integers s_i and k_i each ($1 \leq s_i \leq n$, $0 \leq k_i \leq m$) – the number of the station where Santa begins his journey, and his budget in CHF, respectively.

Output

For each scenario output one number on a single line – the number of the station closest to the EPFL (in terms of distances given by the bus network) that Santa can reach.

Example

stdin or input.txt	stdout or output.txt
5 1 2 2 3 2 4 1 5 2 3 1 4 2 4 2 0 4 1 4 2 5 2	2 2 1 5

Note

In the first example scenario, Santa is broke and cannot do much – it is best for him to stay at station 2.

In the second scenario, he can use the train from 4 to 2.

In the third scenario, he can use the train from 4 to 2, then the bus from 2 to 3, then the train from 3 to 1 and thus end up at EPFL.

In the fourth scenario, Santa is not allowed to leave station 5.

We only guarantee that this problem is solvable in C++ and, probably, Java (as opposed to e.g. Python).