TWGSS
OI Team

Cooperation Theory II

SCHEDULE

Date	26 February 2024	
Time	08:15 - 12:15	4 hours

RULES & REMINDERS

- Teams are ordered by the number of **Solved problems**, followed by the **total penalty time**.
- All test cases in a problem must be passed to be considered **Solved (Accepted)**.
- A submission with the verdict **Wrong Answer**, **Time Limit Exceeded** or **Runtime Error** is considered **Rejected**.
- The **penalty time** for a solved problem is the number of minutes past the beginning of the contest that it is solved, plus the number of **Rejected** submissions multiplied by 20 minutes.
- The **total penalty time** is the sum of all solved problems' **penalty time**.
- You are recommended to attempt the problems using C++20. It is not guaranteed that the problems are solvable using other languages.
- Unless otherwise specified, your output will be automatically fixed as follows:
  - Trailing spaces in each line will be removed.
  - An end-of-line character will be added to the end of the output if not present.
- 64-bit integers may be required in some problems, use `long long` in C++ if needed.
- The problems are NOT necessarily arranged in ascending order of difficulty.
- All stories are hypothetically written. In case of coincidences, they are coincidences.

AUTHORS / PREPARED BY

Hsieh Chong Ho

Wong Sheung Chit

PROBLEMS

	Problem			Limit	
				Time	Memory
TP241	*A*		Author's Problem	1s	256MB
	B		Balls of Plastic		
	C		Colourful Wristband		
	D		Decrypting Challenge		
	E		Episodic Intermix		
	F		Failing Rate		
	G		Great Milk Tea		
	H		Heung Shing Index		
	I		Iron Armour		
	J		Jyutping Street		
	K		King of Guessing		
	L		Logical Puzzle		

TP241 Author's Problem

Limit
1s | 256MB

"It is bizarre that a candidate copied the author's name from left to right while the text is printed in the reverse direction."

"This is hilarious. Is that person an official public exam candidate?" You are laughing at the news while you are coding. *"Why is this not working? Oh no! I get the operator precedence in the reverse direction!* `&&` *should be considered before* `||`*!"* Oops! It turns out that you are laughing at yourself.

To remind yourself not making this kind of mistakes again, you decide to write a program to extract the author's name from a reversed sentence that originally begins with `Author is `, i.e. ends with `si rothuA` after reversed.

INPUT

The first and only line consists of a string S, the sentence stating the author's name.

CONSTRAINTS

$11 \leq \text{Length of } S \leq 50$

S ends with `si rothuA`

S only contains English characters and spaces

The name does not begin or end with spaces

OUTPUT

Output a string, the author's name.

SAMPLE TESTS

	Input	Output
1	`uixuD nehC si rothuA`	`Chen Duxiu`
2	`citamelborp si rothuA`	`problematic`

TP241 Balls of Plastic

Limit
1s | 256MB

"In a box, there are 5 red balls and 4 black balls. From the box, 2 balls are randomly chosen at the same time. Find the probability that that the two balls chosen are red."

"It's trivial!" You *don't even bother to give plastic* to these kind of problems in the public examination as they are too easy! Suddenly, you invented a plastic machine that will ~~generate plastic story~~ deal with red and black balls!

There are N plastic balls, all of them are either red or black. In the plastic machine is a plastic tube that can store plastic balls. The order of balls in the tube is called the **ball pattern**. Whenever a ball is inserted to the machine, one of the following will be performed by the machine:

- If a **red** ball (`.`) is inserted, it is added to the end of the tube, i.e. is appended at the end of the ball pattern.
- If a **black** ball (`#`) is inserted, the whole tube flips over, i.e. the ball pattern is reversed first. Then, the black ball is added to the end of the tube, i.e. is appended to the end of the ball pattern, **after its reversal**.

Being so smart, you decide to challenge yourself by writing a program to construct a sequence of inserting balls that would result in a specific final ball pattern!

INPUT

The first line consists of an integer N, the total number of balls.

The second line consists of a string T of length N, representing the required final ball pattern, which `.` represents a red ball and `#` represents a black ball.

CONSTRAINTS

$1 \leq N \leq 2 \times 10^5$

S consists of `.` and `#` only

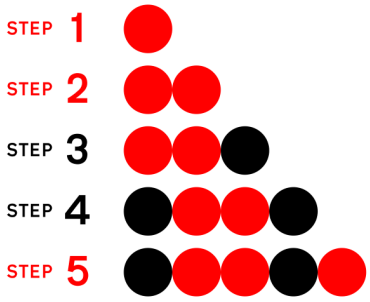
OUTPUT

Output a string of length N consisting of `.` and `#` only, representing the sequence of inserting balls. Which if the balls are inserted into the machine in this order, the final ball pattern would be T.

It is guaranteed that an answer always exists. If there exist multiple possible sequence of inserting balls, you may output anyone of them.

SAMPLE TESTS

	Input	Output
1	5 #..#.	..##.



2	10 #.#.#.#.#.	.#.#.#.##.
3	11 ##.#..#...#	.#..##...##

TP241 Colourful Wristband

Limit: 1s | 256MB

We are OI the best of all, our spirit will never fall!

With your fabulous cheerleading performance on the annual E-Sports Day, you are given a colourful wristband as a gift!

Upon closer inspection, you find that the wristband contains N cells, which we can label them as cell $1, 2, \ldots, N$. Cell i is adjacent to cell $(i+1)$ for all $1 \leq i \leq N-1$, while cell N is adjacent to cell 1. Each cell is in a distinct colour.

As a Maths enthusiast, you discover a spectacular property of the wristband: if you consider different colours as different integers from 1 to N according to the darkness of the colours, it is possible to make all integers on adjacent cells of the wristband are co-prime, that is, they have a GCD (greatest common divisor, or HCF) of 1.

However, some critics do not believe in your mathematics ability and challenge you! Can you construct explicitly the integer sequence to show that it is indeed possible (or to admit that you are bluffing)?

INPUT

The first and only line consists of an integer N, the number of cells in the wristband.

CONSTRAINTS

$1 \leq N \leq 2 \times 10^5$

OUTPUT

If there is no solutions, output `NO` on the first and only line.

Otherwise if there exists a solution, output `YES` on the first line. On the second line, output N integers $a_1, a_2, \ldots, a_N$, describing the integer sequence of your choice.

If there are multiple solutions, you may output any of them.

SAMPLE TESTS

	Input	Output
1	3	YES 1 3 2
2	5	YES 1 4 3 5 2

TP241 Decrypting Challenge

Limit
1s | 256MB

"Thank you for joining the challenge! See you next time! :)"

After the last challenge ended in December 2021, you have been looking forward for the next Decrypting Challenge despite its name is actually grammatically incorrect! Luckily, you joined the OI Team and found the original author of the past two challenges!

"Ding!" You get a mysterious message from the original author! You suspect that the message is encrypted with the method of **XOR Cipher**. To decrypt the message, the ASCII value of each character in it is marked down. Then, each ASCII value is XOR with a fixed unknown value X ($0 \leq X \leq 127$), producing the encrypted message.

Dedicated to win this "Decrypting Challenge", you decide to write a program to find the unknown X for decryption and so the secret message can be revealed.

INPUT

The first line consists of an integer N, the number of characters in the message.

The second line consists of N integers $A_1, A_2, \ldots, A_N$, where A_i represents the ASCII code of the i^{th} character in the encrypted message.

CONSTRAINTS

The message is written in English which contains common English words

$500 \leq N \leq 10^5$

$0 \leq A_i \leq 127$

OUTPUT

Output an integer X, the unknown for the encryption.

SAMPLE TESTS

	Input	Output
1	20 104 94 17 93 94 95 86 17 69 88 92 84 17 95 94 17 66 84 84 16	49

Encrypted message with the unknown $X = 49$: `Yo long time no see!`

Please note that this sample does not actually satisfy the input constraints as its length is too short, and so it serves the purpose of sample only. The actual test cases have messages of longer lengths.

HINT

In C++, A XOR B can be computed as `A ^ B`.

TP241 Episodic Intermix

Limit
1s | 256MB

You have been recently addicted to an anime - "Suzumiya Haruhi no Yūutsu"!

The latest season of the anime consists of N episodes, which **in story order** are called episode $1, 2, \ldots, N$. You want to watch all of them in the story order (of course)!

However, there is an interesting issue: The **actual order** of the episodes are permutated! That is, the actual order might be different from the story order.

You decide the watch the whole season in its **actual order** several number of times so that you can understand the story completely. That is, you can understand episode 1 once you watch it, and you can understand episode K if you have previously understood episode $K-1$.

As you still have school tomorrow, you decide to write a program to find out the minimum number of times that you need to watch the whole season in its actual order so that you can understand the story completely.

INPUT

The first line consists of an integer N, the number of episodes in the season.

The second line consists of N integers $p_1, p_2, \ldots, p_N$, the actual order of the episodes.

CONSTRAINTS

$1 \leq N \leq 2 \times 10^5$

$1 \leq p_i \leq N$

$p_i \neq p_j$ for $i \neq j$

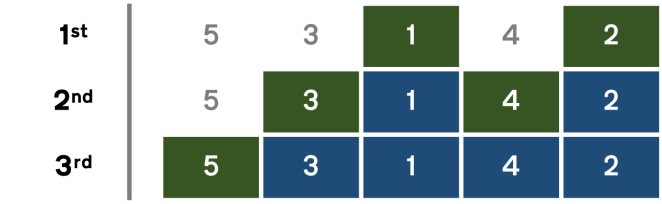
OUTPUT

Output an integer, the minimum number of times that you need to watch the whole season in its actual order so that you can understand the story completely.

SAMPLE TESTS

	Input	Output
1	5 1 3 2 4 5	2

	Input	Output
2	5 5 3 1 4 2	3



TP241 Failing Rate

Limit: 1s | 256MB

Now, they are facing the ultimate challenge — the final exam!

In Byteland, $2N$ students enrolled in a competitive programming course. In the final exam, they are asked to pair up to form N groups of 2, and to collaborate together to improve their score. However, as a super-strong teaching assistant - upon knowing the problemset - can already predict the score of all the $2N$ students accurately! It is decided by professor X that the sum of the scores of the 2 students in a group must be greater or equal to P to pass the course. Otherwise, both students of the group fail the course, and will have to perform the song "I'm a Programming Failure" in front of everyone!

As a kind teaching assistant, you decide to calculate the minimum number of groups that fail the course upon the optimal arrangement.

INPUT

The first line consists of two integers N and P, the number of groups and the passing score respectively.

The second line consists of $2N$ integers $S_1, S_2, \ldots, S_{2N-1}, S_{2N}$, the score of the students in the class respectively.

CONSTRAINTS

$1 \leq N \leq 10^5$

$1 \leq P \leq 200$

$0 \leq S_i \leq 100$

OUTPUT

Output a single integer, the minimum number of groups that fail the course.

SAMPLE TESTS

	Input	Output
1	3 160 64 68 74 83 95 98	1

Here is a possible grouping with one group that fails:

- Student 1 pairs with Student 6. Total Score = $64 + 98 = 162$. Therefore, they passes.
- Student 2 pairs with Student 4. Total Score = $68 + 83 = 151$. Therefore, they fails.
- Student 3 pairs with Student 5. Total Score = $74 + 95 = 169$. Therefore, they passes.

It can be shown that no grouping would lead to the situation that all the groups passes the exam.

2	5 4 1 4 2 3 1 5 4 1 2 1	0

TP241 Great Milk Tea

Limit
1s | 256MB

Typhoon Milktea is approaching Hong Kong.

Upon entry to The Coder's University of Hong Kong (CUHK), Daniel gradually becomes addicted to milk tea!

As a milk tea enthusiast, he slowly realised that the proportion of milk and tea affects the taste by a lot! Therefore, he has determined an acceptable range for the proportion of **milk** in a cup of **great milk tea**. Formally, if the lower bound and the upper bound percentage are $L\%$ and $R\%$ respectively, then $L\% \leq \frac{\text{Volume of milk}}{\text{Volume of the milk + tea}} \leq R\%$ must hold.

Now, again, as a milk tea enthusiast, he has bought M cups of milk and T cups of tea to make as many cups of great milk tea as possible! Given the acceptable range of milk, can you find out how many **full** cups of great milk tea can be made?

INPUT

The first line consists of two integers L and R, the lower bound and upper bound of the milk content in a cup of milk tea in percentage, respectively.

The second line consists of two integers M and T, the number of cups of milk and tea owned by Daniel, respectively.

CONSTRAINTS

$1 \leq L \leq R \leq 99$

$0 \leq M, T \leq 1000$

OUTPUT

Output a single integer, the maximum number of acceptable **full** cups of great milk tea Daniel can make.

SAMPLE TESTS

	Input	Output
1	1 25 5 6	8

Optimally, Daniel can make 8 full cups of great milk tea with each consists of 0.25 cups of milk and 0.75 cups of tea. This, in total, will sum up to 2 cups of milk and 6 cups of tea. Daniel cannot make extra full cup of great milk tea with the only 3 cup of milk remaining, as it is not acceptable for Daniel to have more than 25% of milk in his great milk tea.

It can be shown that there is no way of making more than 8 cups of great milk tea by varying the content of milk in each cup.

2	60 70 5 6	8
3	1 99 5 6	11

TP241 Heung Shing Index

Limit: 1s | 256MB

"That's because you are pessimistic, I see the benefits brought by from stability to prosperity."

You are a stock enthusiast! You are tired of seeing the current situation of the Heung Shing Index as it is dropping horribly! Luckily, your friend Chizuru has some *mysterious* ways to know the closing price of the Heung Shing Index for each of the next N days, with the i^{th} day has the closing price P_i. However, to prevent being too suspicious, you decide to only buy and sell one share of the special stock with the same price as the Heung Shing Index **at most once**, which means if you buy the stock on day x and sell it on day y for some $1 \leq x < y \leq N$, you will have the earnings of $P_y - P_x$. You may also choose not to buy the stock if there is no way for you to have positive earnings. In this situation, your earnings are 0.

As you are not good at Maths, you decide to write a program calculating the maximum earnings that you can get.

INPUT

The first line consists of an integer N, the number of days that Chizuru can foresee the price of the Heung Shing Index.

The next line consists of N integers $P_1, P_2, \ldots, P_N$, where P_i refers to the closing price Heung Shing Index on day i.

CONSTRAINTS

$1 \leq N \leq 2 \times 10^5$

$1 \leq P_i \leq 10^9$

OUTPUT

Output an integer, the maximum earnings that you can get by buying and selling the special stock at most once.

SAMPLE TESTS

	Input	Output
1	5 1 5 2 6 3	5

The optimal strategy is to buy on day 1 with price 1, and sell on day 4 with price 6. This gives a net gain of $6 - 1 = 5$.

	Input	Output
2	5 5 4 3 2 1	0

The optimal strategy is to do nothing, with a net gain of 0.

TP241 Iron Armour

Limit
1s | 256MB

"If you cannot solve the problem, solve the problem author."

The problem being too difficult is *Author's Problem*! As the problem author, Daniel does not want to be solved! Therefore, he decides to make an iron armour to protect himself.

It is known that in The Coder's University of Hong Kong (CUHK), there are N landmarks and M roads. Each road connects two distinct landmarks while there exists at most one direct road between two distinct landmarks. Daniel is currently at landmark 1 and he cannot move due to tiredness. The **distance** between two landmarks is defined to be the minimum number of roads needed to pass through when travelling from one landmark to another. Interestingly, at each of the N landmarks, there are some iron deposits buried underground, which the i^{th} landmark has A_i units of iron.

In order to make an iron armour costing I units of iron to protect himself, Daniel decides to give himself a powerful magnet: a magnet that can attract all the iron in landmarks that has a distance not exceeding d to landmark 1. As iron is heavy, he wants to minimise the quantity of iron collected while sufficient to make an iron armour.

Now, solving or not solving this problem, is the ultimate problem. Can you find the optimal distance d, or that it is impossible for Daniel to build an iron armour?

INPUT

The first line consists of three integers N, M and I, representing the number of landmarks, the number of roads and the number of units of iron needed to make an iron armour respectively.

The second line consists of N integers $A_1, A_2, \ldots, A_N$, where the i^{th} landmark has A_i units of iron.

The next M lines describe the roads. The i^{th} line consists of two integers u_i and v_i, which the two landmarks are connected by the i^{th} road.

CONSTRAINTS

$2 \leq N \leq 10^5$

$N - 1 \leq M \leq 2 \times 10^5$

$1 \leq I \leq 10^{14}$

$0 \leq A_i \leq 10^9$

$1 \leq u_i < v_i \leq N$

$(u_i, v_i) \neq (u_j, v_j)$ for $i \neq j$

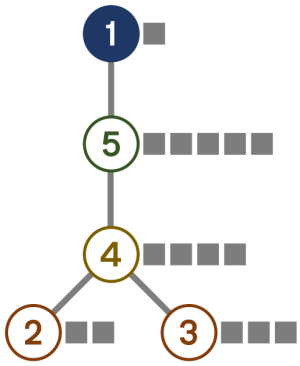
OUTPUT

If it is possible to make an iron armour, output the optimal distance d.

Otherwise, output `-1` if it is impossible to make an iron armour.

SAMPLE TESTS

	Input	Output
1	5 4 13 1 2 3 4 5 3 4 2 4 4 5 1 5	3



2	5 4 16 1 2 3 4 5 3 4 2 4 4 5 1 5	-1
3	5 4 3 1 2 3 4 5 3 4 2 4 4 5 1 5	1
4	5 5 13 3 1 5 4 2 3 4 2 4 4 5 1 5 1 3	2

TP241 Jyutping Street

Limit
1s | 256MB

"A new font type will be used on street name plates to instil a stronger cultural vibe/atmosphere."

Recently, Heung Shing is updating its street name plates in various places. Instead of changing the font that have a "stronger cultural vibe", the city is going for an even more unexpected approach — changing Chinese characters on the plates to their Cantonese romanisation counterpart using Jyutping!

In this advanced era, street name plates are generated using computers by assuming each character or space takes up 1 unit of width. A street name plate consists of two rows. It is divided into three parts, from left to right: **left border**, **names**, **right border**. **Between each part, 1 unit space gap is inserted** to improve the appearance of the plate.

For the **left border** and **right border** part, the two characters in the string dir represents whether the border of the plate is pointing towards left and right respectively. Each of the both parts is 1 unit in width respectively.

dir	*left border*	*right border*
||	| |	| |
<|	/ \	| |
|>	| |	\ /
<>	/ \	\ /

For the **names** part, the top row should show the English name of the street, represented by string E. While the bottom row should show the Jyutping of the street, represented by string C. The width of the names part is the longer of E and C while the shorter one should align to the centre of the longer one. It is guaranteed that **no half unit is needed** to align the strings.

As an honourable Heung Shing citizen, you clearly see the benefits brought by this change. Therefore, you decide to write a program to help the Government to generate new street name plates!

INPUT

The first line consists of a two-character string dir, representing the **left border** and **right border** part.

The second line consists of a string E, the English name of the street.

The third line consists of a string C, the Jyutping of the street.

CONSTRAINTS

$dir =$ `||`, `<|`, `|>` or `<>`

$1 \leq \text{length of } E, C \leq 100$

E, C consist of printable ASCII characters

OUTPUT

Output the resultant street name plate, which should consist of two lines.

SAMPLE TESTS

	Input	Output
1	|| Lo Tak Court lou6 dak1 wai4	| Lo Tak Court | | lou6 dak1 wai4 |
2	<| Hoi Pa Street hoi2 baa3 gaai1	/ Hoi Pa Street | \ hoi2 baa3 gaai1 |
3	|> Cooke Street kuk1 gaai1	| Cooke Street \ | kuk1 gaai1 /
4	<> Queen's Road East wong4 hau6 daai6 dou6 dung1	/ Queen's Road East \ \ wong4 hau6 daai6 dou6 dung1 /

TP241 King of Guessing

Limit: 1s | 256MB

"It's MC Daniel time!"

The long-awaited MC Daniel's performance is finally here! *"Welcome to my interactive performance! Who answers the most questions correctly can be the 'King of Guessing'!"* A multiple-choice answer sheet with Q questions suddenly appears in front of you! You have completely no idea of what the questions are actually about as there is no question paper! Facing this devastating situation, you decide to use your ultimate power — luck!

For each of the Q questions, there are X choices. So by simply independently guessing each question, you will get it correct with a fair $\frac{1}{X}$ probability. You know that you will need to answer **at least** P questions correctly to be the "King of Guessing".

Being so bored after guessing the questions, you decide to write a program to calculate the probability that you can be the "King of Guessing".

INPUT

The first and only line consists of three integers Q, P and X, representing the number of questions, the number of correct answers needed to be the "King of Guessing" and the number of choices for each question respectively.

CONSTRAINTS

$1 \leq P \leq Q \leq 100$

$2 \leq X \leq 5$

OUTPUT

Output a floating-point number, the passing probability.

Your answer will be accepted if the absolute or relative error of your answer compared to the author's solution is less than or equal to 10^{-6}.

SAMPLE TESTS

	Input	Output
1	1 1 5	0.2000000000
2	20 10 4	0.0138644169

HINT

The following C++ code snippet can output a floating point number a rounded off to 10 decimal places:

```
cout << fixed << setprecision(10) << a;
```

TP241 Logical Puzzle

Limit
1s | 256MB

It is puzzled to solve the puzzle of a problem on a puzzling puzzle.

Nicolas recently likes solving puzzles ~~(especially the game 2048)~~. Here is a puzzle that his friends proposed to him:

Initially, two 01-strings S and T of length N are given. The goal of the puzzle is to do some (possible zero) operations to make S equal to T.

An operation consists of the followings:

- First, select an index i ($2 \leq i \leq N-2$), which its neighbours, i.e. S_{i-1} and S_{i+1}, are different.
- Then, flip the i^{th} bit of S. That is, if it is originally `0`, you change it to `1`; if it is originally `1`, you change it to `0`.

Nicolas does not know how to solve the puzzle, so he asks you for help! Can you write a program to determine if it is possible to solve the puzzle?

INPUT

The first line consists of an integer N, the number of bits in each of the strings.

The second line consists of a 01-string S of length N, representing to the starting state of the puzzle.

The third line consists of a 01-string T of length N, representing to the target state of the puzzle.

CONSTRAINTS

$3 \leq N \leq 5 \times 10^5$

S and T consist of `0` and `1` only

OUTPUT

Output `YES` if it is possible to convert S to T using finitely many (possibly zero) operations.

Otherwise, output `NO`.

SAMPLE TESTS

	Input	Output
1	5 11010 10100	YES

A possible way to solve the puzzle: `11010` → `10010` → `10110` → `10100`

2	9 110010101 011011100	NO

3	2 11 01	NO