Problem A. Bookshelf Filling

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

Recently, Joseph bought lots of books and was going to put them all on the shelf. He found that there are only two types of books – the type A with height a and thickness 1, and the type B with height b ($a < b$) and thickness 1. The number of each type of book are n and m respectively. And he already knew the height of the bookshelf which is h ($h \geq b$).

In the beginning, he arranged all the books on the shelf in height-ascending order, which means the books of height a are all on the left, and the books of height b are all on the right. The total length these books occupied on the bookshelf will be $n + m$.

However, Joseph found there was still a vacancy on top of these books, so he prepared to select k ($0 \leq k \leq m - 1$) books with height b from the **rightmost** side and put them all horizontally on top of other books to reduce the length they occupied. He wanted to know the minimum width if he arranges these books optimally.

Input

The first line of the input gives the number of test cases T ($1 \leq T \leq 10^3$). T test cases follow.

For each test case, only one line contains five integers a, b, n, m, h ($1 \leq a < b \leq h \leq 10^6, 1 \leq n, m \leq 10^9$)

Output

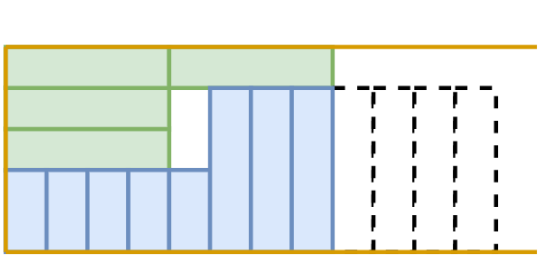
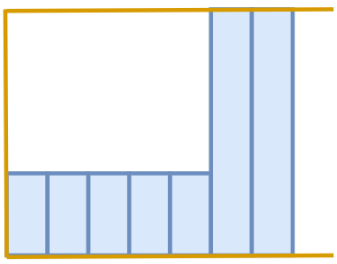
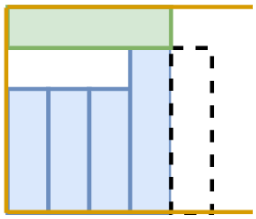
For each test case, print a line contains one integer w, indicating the minimum width.

Example

standard input	standard output
4 2 5 5 8 5 2 4 5 7 5 2 6 5 2 6 3 4 3 2 5	10 8 7 4

Note

The following pictures from left to right show the possible arrangement of books for the testcases 2, 3 and 4.

Problem B. Lovely Fish

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

$Fish$ is a working-class citizen, he works for company H(Heinz). One day, $Fish$ wants to know how much his coworkers like him.

After some research, $Fish$ organized the results into a string S

1. when $S_i = 1$, this means that the ith worker likes Fish.
2. when $S_i = 0$, this means that the ith worker dislikes Fish.

After researching, $Fish$ copied down $S_{l...r}$ as a new string T, and saw that there were many 0s among T, he became very sad. So he wished to insert some 1 to any position of T in order to make himself happier.

If the length of T is m after inserts, $Fish$ would become sad under either of the following two conditions:

1. there exists a p $(1 \leq p \leq m)$ which the number of 0 are strictly larger than the number of 1 within T_1 to T_p.
2. there exists a p $(1 \leq p \leq m)$ which the number of 0 are strictly larger than the number of 1 within T_p to T_m.

$Fish$ also doesn't know for certain the value of l and r, so he wants to know with q possible pairs of l and r, the least number of inserts that Fish needs to make in order to make himself happy.

Input

The first line contains two integers n, q $(1 \leq n, q \leq 1,000,000)$ denoting the numbers of colleagues and questions.

The second line contains a string S.

for the next q lines, each line contains two integers l_i, r_i, requesting the least number of changes if T=$S_{l...r}$

Output

To reduce the impact of output time, you only need to output the `XOR` of each query.

Example

standard input	standard output
10 5 0001001111 1 6 1 10 5 7 4 7 4 6	0

Note

For the example, the answers are $5, 4, 2, 1, 2$.

the possible solutions are:

000100->**10101010101**

0001001111->11100010101111

001->10101

1001->10101

100->10101

Problem C. Tree Division

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

You are given a tree of n nodes, the i-th node has value a_i.

Define a node t is valid, if you can divide the n nodes into two sets A and B, such that the following condition would hold:

- $\forall u, v \in A(u \neq v)$, if u is on the path from t to v, then $a_u < a_v$
- $\forall u, v \in B(u \neq v)$, if u is on the path from t to v, then $a_u > a_v$

You need to find out whether the node 1 is valid.

Input

The first line contains one integer $n(1 \leq n \leq 10^5)$.

The second line contains n integers $a_1, a_2, \cdots, a_n(1 \leq a_i \leq n)$.

The next $n-1$ lines each contains two integers x and y $(1 \leq x, y \leq n)$, denoting an edge between node x and y.

Output

Output `YES` if node 1 is valid, otherwise print `NO`.

Examples

standard input	standard output
8 8 1 4 2 5 6 3 7 3 7 5 3 2 4 5 2 6 5 8 6 1 8	YES
6 4 2 1 5 3 1 1 2 2 3 3 4 4 5 1 6	NO

Note

For the first testcase, node 1 is valid.

A possible division is: $A = \{2, 3\}$,$B = \{1, 4, 5, 6, 7, 8\}$.

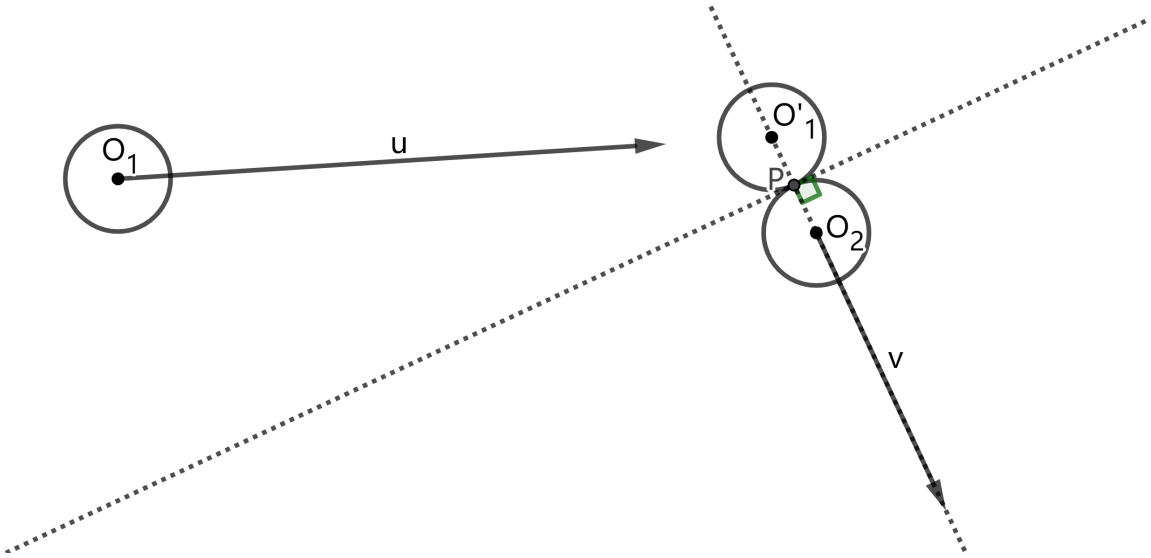
Problem D. Collision Detector

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

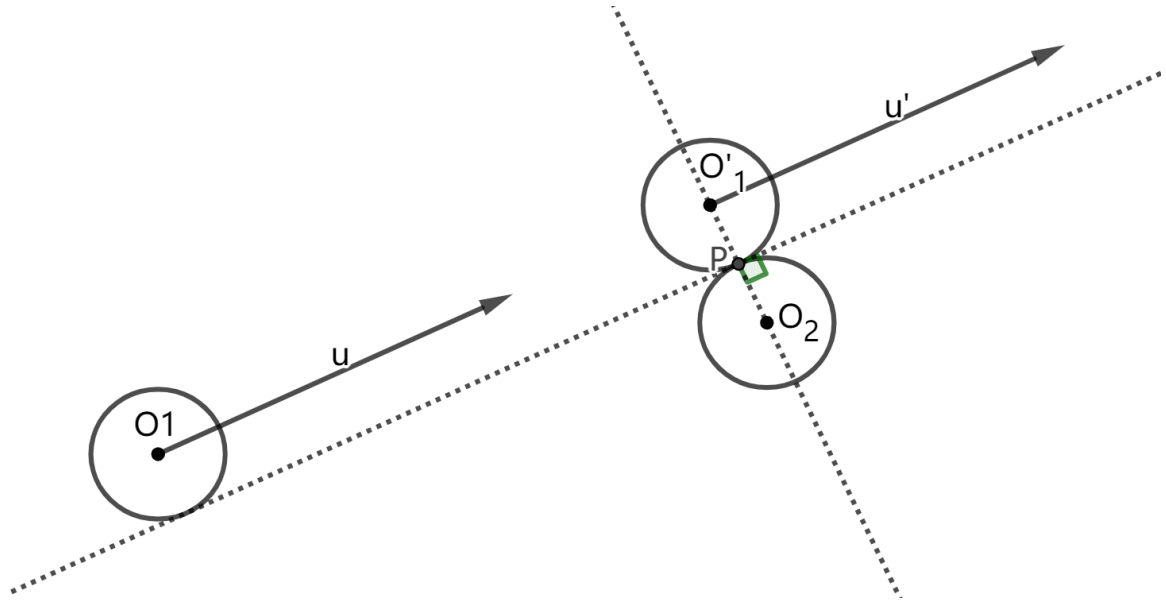
Joseph loves physics. And one day, while Joseph was playing ping-pong, he was obsessed with the study of the collision of two balls.

He found the collision of any two of the balls is completely elastic, which means there is no kinetic energy loss and the sum of the momentum remains the same after the collision.

However, you only need to consider a simpler case in this problem. If we regard all ping-pong balls as the circles of radius 1 on the two-dimensional plane, the collision of a moving ball and a static ball can be described by the figure below. Ball O_1 has the speed $\boldsymbol{u}$ initially, and then it hits a static ball O_2 at the point P. Let the tangent(切线) of circle O_2 at point P be l. The speed of ball O_2 after collision will be a speed of $\boldsymbol{v}$ which is perpendicular(垂直) to l.



Particularly, if the trail of O_1 is just tangent to O_2 (see the figure below), there will be no collision.



Now Joseph put three static balls O_1, O_2, O_3 on the table. He wants to give the ball O_1 an initial speed u and see if it can hit O_2 and let O_2 finally hit O_3. Please write a program to tell him whether the initial speed exists.

Input

The first line of the input gives the number of test cases, T ($1 \le T \le 10^3$). T test cases follow.

For each test case, only one line contains six integers $x_1, y_1, x_2, y_2, x_3, y_3$, which indicates the coordinate of O_1, O_2, O_3 are $(x_1, y_1), (x_2, y_2), (x_3, y_3)$ respectively. It's guaranteed that the input circles do not intersect.

All of the x_i and y_i are in the range $[0, 10^4]$.

Output

Print `yes` if there exists an initial speed of O_1 such that O_1 can hit O_2, and then O_2 moves to hit O_3 finally, otherwise print `no`.

Example

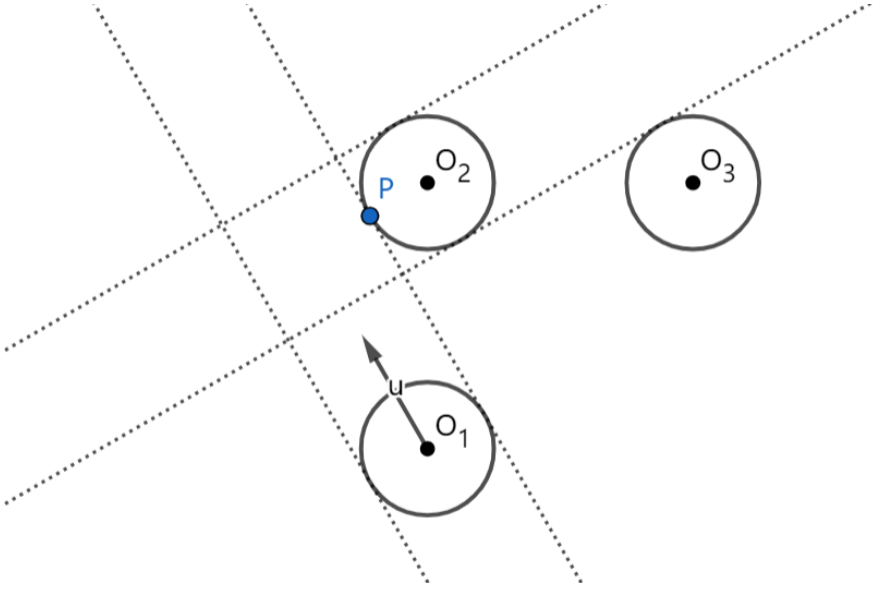
standard input	standard output
4 7 2 4 2 1 2 2 8 3 5 3 0 1 1 1 5 5 5 0 0 2 2 6 0	yes yes no no

Note

For the first testcase, one possible initial speed of O_1 can be $(-1, 0)$.

For the second testcase, one possible initial speed of O_1 can be $(1, -1)$

The figure below shows the third testcase. In the only solution the trail of O_1 is tangent to O_2, so the collision cannot happen,



Problem E. Exclusive Multiplication

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

Megumi is a girl who loves number theory. Today is her birthday, so Joseph creates a new function $f(n)$ and gives it to her as a gift. The definition of $f(n)$ is shown in the formula as follows.

$$f(n) = \prod_{i=1}^{m} p_i^{a_i \% 2}$$

where $p_1, p_2, \cdots, p_m$ $(p_1 < p_2 < \cdots < p_m)$ are prime factors of n and

$$n = \prod_{i=1}^{m} p_i^{a_i}$$

Now Joseph gives you n integers $b_1, b_2, \cdots, b_n$ and wants you to calculate the following formula.

$$\left(\sum_{1 \le i < j \le n} f\left(b_i \times b_j\right) \right) \mod (10^9 + 7)$$

Input

The first line contains one integer n $(1 \le n \le 2 \times 10^5)$.

The second line contains n integers $b_1, b_2, \cdots, b_n$ $(1 \le b_i \le 2 \times 10^5)$.

Output

Output one integer denoting the value of the formula you calculated.

Examples

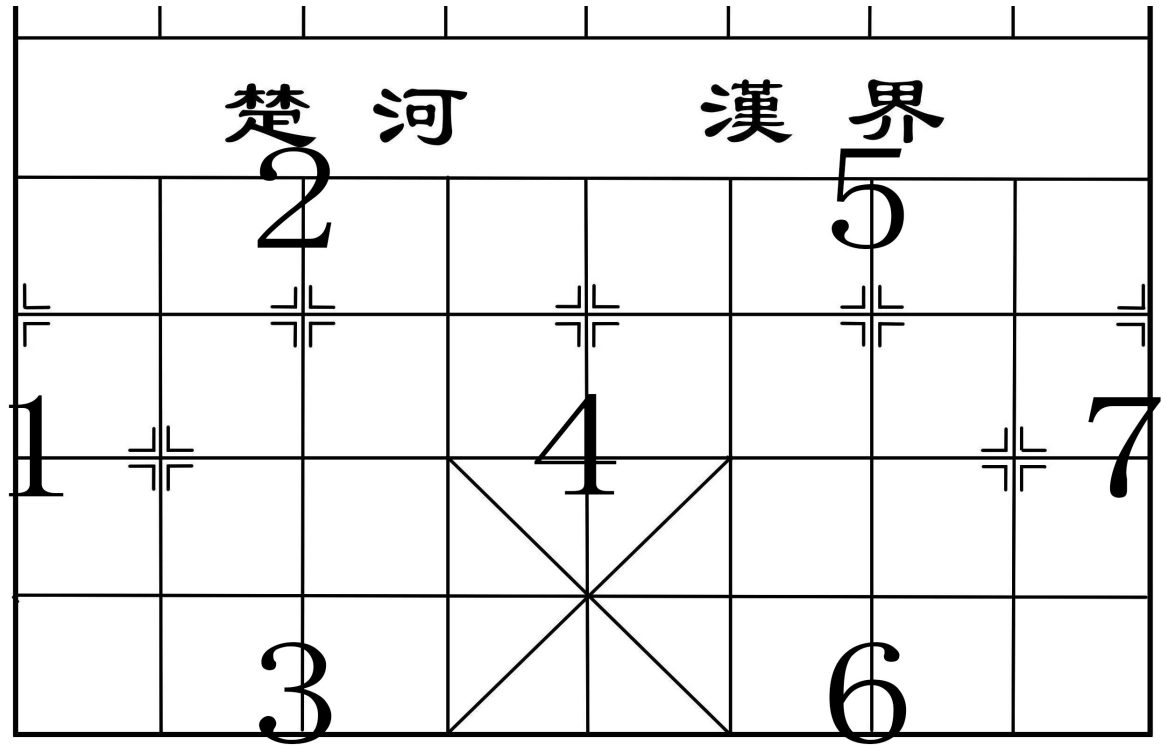
standard input	standard output
4 1 2 3 2	20
5 5 6 2 3 8	86

Problem F. 342 and Xiangqi

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

342 likes playing Xiangqi (as it's called Chinese chess). His favorite piece in Xiangqi is "Xiang" because his opponents usually overlook the existence of Xiang so their major pieces are often taken by 342's Xiang.

For those of you who don't know how Xiang moves in Xiangqi, Xiang is a minor piece that each player initially has two, located at the position 3 and 6 respectively. The number for each position is showed in the illustration bellow. In a single move, Xiang should move 2 squares horizontally as well as vertically. But note that it can neither move out of the half board nor it can overlap with other pieces. The illustration below showed all the 7 possible positions that Xiang could reach. For example, in a single move,



Centered scaled image.

- A Xiang at position 2 **can** move to position 4,
- A Xiang at position 7 **can** move to position 6,
- A Xiang at position 3 **cannot** move to position 5.

342 has 2 Xiangs as the only 2 pieces in the half board. Their present positions are given and guaranteed to be 2 different positions out of the 7 possible ones. But 342 disliked the present positions of his 2 Xiangs and would like to adjust their positions by a few moves. In each move, he can choose one of the Xiangs and move it following the rules explained. The final positions 342 want are also given and guaranteed to be 2 different positions out of the 7 possible ones. Note that **the 2 Xiangs are considered the same**, meaning that he doesn't restrict that which of the 2 Xiangs has to move to which of the 2 positions.

Now 342 wonders, **at least how many moves** he need to adjust the 2 Xiangs from the present positions to the final positions that 342 want?

Input

The first line contains a single integer T ($1 \leq T \leq 10^5$) – the number of test cases.

This is followed by T lines, representing each test case, each containing a single line of four integers a_1, a_2, b_1, b_2 such that $1 \leq a_1, a_2, b_1, b_2 \leq 7,\ a_1 \neq a_2,\ b_1 \neq b_2$. a_1, a_2 represent the present positions of the 2 Xiangs while b_1, b_2 represent the final positions of them.

Output

For each test case, print a single integer: the least number of moves that 342 needs to adjust his 2 Xiangs to the final positions, considering the 2 Xiangs to be the same?

Example

standard input	standard output
3 1 5 1 7 2 7 4 6 2 3 3 2	1 2 0

Problem G. Chevonne's Necklace

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

For her excellent performance, pianist Chevonne was awarded a pearl necklace. The necklace consisted of n pearls which form a circle and are numbered $1, 2, \cdots, n$ clockwise.

To prevent the pearls from being covered with dust, she wants to move pearls from the necklace and wipe them. However, the way to remove the pearls is complex because of their uniqueness of them.

Specifically, every pearl on the necklace has a continuity value c_i ($c_i \geq 0$). Each turn, Chevonne can choose a pearl i with $c_i \geq 1$ and move the pearl i and $c_i - 1$ clockwise consecutive pearls (c_i pearls in all). Be careful! If the number of remaining pearls is **less than** c_i, she will be unable to choose pearl i. After that, the remaining pearls will form a new circle.

Chevonne will repeat the process till she can't remove any pearls or all the pearls have been removed. She is curious about the maximum number of pearls she could remove and the number of such solutions.

Here, we use a set to describe a solution. The set contains the number i if Chevonne chose pearl i in some step. Two solutions are different if and only if the sets are different.

Input

The first line of input contains one positive integer n ($1 \leq n \leq 2000$).

The following one line contains n non-negative integers separated by spaces. The i-th number represents c_i ($0 \leq c_i \leq 2000$). Pearls are numbered clockwise from 1 to n.

Output

Please output two integers separated by one space.

The first one is the maximum number of pearls Chevonne could remove and the second one is the number of such solutions. Since the number of solutions may be very large, please output it modular 998244353.

Example

standard input	standard output
6 0 1 1 3 3 1	6 3

Note

In the example, all the pearls could be removed by Chevonne, the 3 possible solutions are shown as follows:

1. remove the $2, 3, 6, 4$-th pearl in order.
2. remove the $2, 3, 6, 5$-th pearl in order.
3. remove the $5, 4$-th pearl in order.

Problem H. Kanbun

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

Paulliant is a linguist. Recently he received an article written in language A and he was told to translate it into language B. The two languages only differ in word order, so, Paulliant decided to translate by redefining the reading order of the articles, and came up with the following method.

Suppose that the article consists of n words, numbered as $1, 2, 3 \cdots n$. Paulliant generated a string *str* consisting of only '(', ')' and '-', whose length is also n. The string must satisfy the rule of bracket matching. Formally speaking, define $s_{(i}$ and $s_{)i}$, representing the number of character '(' or ')' within the first i characters of the string, respectively, there is $s_{(i} \geq s_{)i}$ for every $1 \leq i \leq n$ and $s_{(n} = s_{)n}$.

The process to read the article obeys the following rules.

(1) The process starts from the first word.

(2) For the i-th word, if str_i is '-' or ')', then read the i-th word directly, and jump to the $(i+1)$-th word.

(3) For the i-th word, if str_i is '(', then find the matching right bracket of it. Supposing it to be str_j, read the $(i+1)$-th to j-th word, then read the i-th word, and finally jump to the $(j+1)$-th word.

(4) If we are to read the $(n+1)$-th word (obviously it doesn't exist), the process ends.

Note that **the process is recursive**, meaning that when reading the $(i+1)$-th to j-th word in rule (2), it still follows the four rules.

For example,

- If $n = 4$, *str* is "(-)- the reading order will be 2, 3, 1, 4.
- If $n = 7$, *str* is ((-)-) the reading order will be 1, 4, 5, 3, 6, 7, 2.

Now, give you the number of words n, and the string *str* Paulliant generated. Your task is to print the reading order following the four rules. It can be proved that the reading order should be **a permutation** of numbers from 1 to n.

Input

The first line contains a single integer n ($3 \leq n \leq 10^5$) — the number of words.

The second line contains a string *str* of n characters. It consists of n characters of only '(', ')', '-', and it's guaranteed to satisfy the rules of bracket matching.

Output

Output a line consisting of n integers which represent the reading order, it should be a permutation of numbers from 1 to n. The n integers are separated by spaces.

Examples

standard input	standard output
4 (-)-	2 3 1 4
7 -((-)-)	1 4 5 3 6 7 2
14 ((-)-()--)(-)-	3 4 2 5 7 6 8 9 10 1 12 13 11 14

Problem I. Equal Sum Arrays

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

Define function $f(n)$ to be the number of different arrays a of positive integers, such that the sum of the elements in array a equals to n.

For example, $f(3) = 4$, and the four different arrays are $\{1, 1, 1\}, \{1, 2\}, \{2, 1\}, \{3\}$.

You are given a positive integer k. Please find the answer of $f(k)$.

Input

There is only one line in the input containing a single integer k ($1 \leq k \leq 20$).

Output

Print a single integer: the answer of $f(k)$.

Examples

standard input	standard output
3	4
4	8

Problem J. JOJO's Happy Tree Friends

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

Joseph is playing a game with his friends. The game goes on a tree – a connected graph with n vertices and $n-1$ edges. The vertices are labeled by $1, 2, \cdots, n$. The root of the tree is 1.

There is a token initially on one of the vertices. At each step, Joseph will randomly choose a vertex from the n vertices with equal probability, and then move the token according to the rules as follows:

Assume the token is currently on vertex u. And then vertex v is selected by Joseph at random

- If u is an ancestor of v, move the token to the vertex v.
- If u is not an ancestor of v, move the token to the vertex $\texttt{LCA}(u, v)$. $\texttt{LCA}(u, v)$ denotes the lowest common ancestor of vectex u and v.

There is also a target vertex w in the tree. As soon as the token is moved to the target vertex, the game is over. Otherwise, the game will keep going until it hits the target vertex.

He wants to know the expected number of steps to hit the target when the token starts from each vertex.

Let's denote the expectation of steps starting from vertex u modulo $10^9 + 7$ by $E(u)$. You are only expected to print

$$\sum_{u=1}^{n} E(u) \oplus u$$

Input

The first line contains one integer n ($1 \le n \le 2 \times 10^5$), indicating the number of vertices.

The following line contains $n-1$ integers $p_2, p_3, ..., p_n$, indicating the parent of each vertex.

The third contains one integer w ($1 \le w \le n$), indicating the target vertex.

Output

Only one integer – the answer after encrypting. See the description for details.

Examples

standard input	standard output
4 1 2 1 1	333333343
6 1 2 3 4 5 5	23
10 1 2 2 1 5 6 6 8 8 8	3263736426

Note

For the first example, the expected number of steps to hit the target starting from vectices $1, 2, 3, 4$ are $0, 2, 2, \frac{4}{3} \equiv 333333337$ respectively.

For the second example, the expectations are $6, 6, 6, 6, 0, 6$ respectively.

Problem K. Monkey Joe

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	512 megabytes

Monkey Joe likes to eat fruit. As monkey Joe is fond of climbing here and there, rather than eating fruits that are already in the grocery shops, he choose to climb the tree and eat the fruit on the tree.

A tree is a connected graph containing n nodes and $n-1$ edges. On every node of the tree, there exists one single fruit. For a fruit on node u, it has a tiredness value of val_u.

Every time Joe eats the fruits, he only travels the simple path from u to v (a simple path is a path that every node and edge is visited only once). When he travels from u to v, he will eat every fruit on the simple path. However, Joe gets very tired of eating all the fruits.

Formally, his tiredness of eating fruit i that lies on the path from u to v is defined as $rank_i \times val_i$, where $rank_i$ denotes the rank of the fruit on node i if you sort all the fruit on the simple path from u to v based on their tiredness values in increasing order. The whole tiredness of path (u, v) is the sum of the tiredness of eating every fruit on the path. It is guaranteed that the tiredness value of each fruit are distinct.

Joe is uncertain which path he would choose to climb, so he would like you to calculate the tiredness sum of all distinct paths of the tree. Since the answer may be very large, you only have to output the answer module $10^9 + 7$

Input

The first line contains one integer n $(1 \leqslant n \leqslant 5 \times 10^5)$—the number of nodes on the tree.

The second line contains n integers $val_1, val_2, ...val_n (1 \leqslant val_i \leqslant 10^9)$—the tiredness value of every node on the tree.

The i-th of the following $n-1$ lines contains two integers u_i and v_i $(1 \leqslant u_i, v_i \leqslant n)$ denoting that there exists an edge connecting vertices u_i and v_i. It is guaranteed that the given edges form a tree.

Output

Print a single integer — the tiredness sum of all paths on the tree module $10^9 + 7$

Examples

standard input	standard output
2 1 2 1 2	8
5 1 2 9 8 4 1 3 1 2 2 5 5 4	357

Note

The path from u to v is considered the same as the path from v to u.

Path with the form (u, u) is regarded as a valid simple path.

Problem L. Let's Swap

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

Joseph developed an editing software called Pandote recently, and now he is testing it to make sure it works correctly, otherwise, he will probably get fired!

Joseph starts his testing by implementing copying and pasting as well as reversing operations to the string on the Pandote. More specifically, in each step, if the string on the screen is S, he will do the following operations in order.

1. Choose a prefix of length l ($1 \le l \le |S|$), then S can be denoted by AB ($|A| = l$). Note that the string B can be empty.
2. Swap the two parts and get the string BA.
3. Reverse the whole string and get the string $(BA)^r$

However, since the function of Pandote is limited, there are only two different lengths l_1 and l_2 of prefix he can choose in each step. Now Joseph wants to know whether he can convert the string S to T through several (possibly zero) steps.

Input

The first line of the input gives the number of test cases T ($1 \le T \le 5 \times 10^5$). T test cases follow.

For each test case, the first line contains the string S and the second line contains the string T. Both S and T are consisting of lowercase Latin letters and $|S| = |T|$.

the third line contains two integers l_1 and l_2 ($1 \le l_1, l_2 \le |S|, l_1 \ne l_2$), denoting the length of the prefix he can choose in each step.

It is guaranteed that the sum of $|S|$ over all test cases does not exceed 5×10^5.

Output

For each test case, print `yes` if he can convert S to T. Otherwise, print `no`.

Example

standard input	standard output
3 ljhelloh hellohlj 2 4 thisisastr htrtsasisi 3 5 abcde bcdea 1 4	yes no yes

Note

For the first testcase, one possible method is `ljhelloh` $\xrightarrow{4}$ `ehjlholl` $\xrightarrow{2}$ `hellohlj`