A. Bazoka and Mocha's Array

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

Mocha likes arrays, so before her departure, Bazoka gave her an array a consisting of n positive integers as a gift.

Now Mocha wants to know whether array a could become sorted in non-decreasing order after performing the following operation some (possibly, zero) times:

- Split the array into two parts — a prefix and a suffix, then swap these two parts. In other words, let $a = x + y$. Then, we can set $a := y + x$. Here $+$ denotes the array concatenation operation.

For example, if $a = [3, 1, 4, 1, 5]$, we can choose $x = [3, 1]$ and $y = [4, 1, 5]$, satisfying $a = x + y$. Then, we can set $a := y + x = [4, 1, 5, 3, 1]$. We can also choose $x = [3, 1, 4, 1, 5]$ and $y = [\,]$, satisfying $a = x + y$. Then, we can set $a := y + x = [3, 1, 4, 1, 5]$. Note that we are not allowed to choose $x = [3, 1, 1]$ and $y = [4, 5]$, neither are we allowed to choose $x = [1, 3]$ and $y = [5, 1, 4]$, as both these choices do not satisfy $a = x + y$.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 1000$). The description of the test cases follows.

The first line of each test case contains a single integer n ($2 \le n \le 50$) — the length of the array a.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^6$) — the elements of array a.

Output

For each test case, output "Yes" if a could become non-decreasing after performing the operation any number of times, and output "No" if not.

You can output "Yes" and "No" in any case (for example, strings "yEs", "yes", "Yes" and "YES" will be recognized as a positive response).

Standard Input	Standard Output
3 6 1 1 4 5 1 4 5 7 9 2 2 3 3 1 2 3	No Yes Yes

Note

In the first test case, it can be proven that a cannot become non-decreasing after performing the operation any number of times.

In the second test case, we can perform the following operations to make a sorted in non-decreasing order:

- Split the array into two parts: $x = [7]$ and $y = [9, 2, 2, 3]$, then swap these two parts. The array will become $y + x = [9, 2, 2, 3, 7]$.
- Split the array into two parts: $x = [9]$ and $y = [2, 2, 3, 7]$, then swap these two parts. The array will become $y + x = [2, 2, 3, 7, 9]$, which is non-decreasing.

,

B. 378QAQ and Mocha's Array

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

Mocha likes arrays, so before her departure, 378QAQ gave her an array a consisting of n positive integers as a gift.

Mocha thinks that a is *beautiful* if there exist two numbers i and j ($1 \leq i, j \leq n, i \neq j$) such that for all k ($1 \leq k \leq n$), a_k is divisible[†] by either a_i or a_j.

Determine whether a is beautiful.

[†] x is divisible by y if there exists an integer z such that $x = y \cdot z$.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 500$). The description of the test cases follows.

The first line of each test case contains a single integer n ($3 \leq n \leq 10^5$) — the length of the array a.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 10^9$) — the elements of the array a.

It is guaranteed that the sum of n over all test cases does not exceed 10^5.

Output

For each test case, output "Yes" if array a is beautiful, and output "No" otherwise.

You can output "Yes" and "No" in any case (for example, strings "yEs", "yes", "Yes" and "YES" will be recognized as a positive response).

Standard Input	Standard Output
4 3 7 3 8 5 7 1 9 3 5 5 4 12 2 6 3 5 7 49 9 3 1000000000	No Yes Yes No

Note

In the first test case, any two numbers in the array are coprime, so the answer is "No".

In the second test case, we can pick $i = 2$ and $j = 1$. Since every number in the array is divisible by $a_i = 1$, the answer is "Yes".

In the third test case, we can pick $i = 3$ and $j = 5$. 2 and 4 is divisible by $a_i = 2$ while 3, 6 and 12 is divisible by $a_j = 3$, so the answer is "Yes".

,

C. Chamo and Mocha's Array

```
Input file:     standard input
Output file:    standard output
Time limit:     2 seconds
Memory limit:   256 megabytes
```

Mocha likes arrays, so before her departure, Chamo gave her an array a consisting of n positive integers as a gift.

Mocha doesn't like arrays containing different numbers, so Mocha decides to use magic to change the array. Mocha can perform the following three-step operation some (possibly, zero) times:

1. Choose indices l and r ($1 \leq l < r \leq n$)
2. Let x be the median[†] of the subarray $[a_l, a_{l+1}, \ldots, a_r]$
3. Set all values $a_l, a_{l+1}, \ldots, a_r$ to x

Suppose $a = [1, 2, 3, 4, 5]$ initially:

- If Mocha chooses $(l, r) = (3, 4)$ in the first operation, then $x = 3$, the array will be changed into $a = [1, 2, 3, 3, 5]$.
- If Mocha chooses $(l, r) = (1, 3)$ in the first operation, then $x = 2$, the array will be changed into $a = [2, 2, 2, 4, 5]$.

Mocha will perform the operation until the array contains only the same number. Mocha wants to know what is the maximum possible value of this number.

[†] The median in an array b of length m is an element that occupies position number $\lfloor \frac{m+1}{2} \rfloor$ after we sort the elements in non-decreasing order. For example, the median of $[3, 1, 4, 1, 5]$ is 3 and the median of $[5, 25, 20, 24]$ is 20.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 500$). The description of the test cases follows.

The first line of each test case contains a single integer n ($2 \leq n \leq 10^5$) — the length of the array a.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 10^9$) — the elements of the array a.

It is guaranteed that the sum of n over all test cases does not exceed 10^5.

Output

For each test case, output the maximum value of the number.

Standard Input	Standard Output
2 2 1 2 5 1 2 3 4 5	1 4

Note

In the first test case, $a = [1, 2]$. Mocha can only choose the interval $(l, r) = (1, 2)$. The array will be changed to $a = [1, 1]$. Therefore, the answer is 1.

In the second test case, Mocha can perform the following operations:

- Choose the interval $(l, r) = (4, 5)$, then $a = [1, 2, 3, 4, 4]$.
- Choose the interval $(l, r) = (3, 5)$, then $a = [1, 2, 4, 4, 4]$.
- Choose the interval $(l, r) = (1, 5)$, then $a = [4, 4, 4, 4, 4]$.

The array contains only the same number, which is 4. It can be proven that the maximum value of the final number cannot be greater than 4.

,

D. Paint the Tree

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 256 megabytes

378QAQ has a tree with n vertices. Initially, all vertices are white.

There are two chess pieces called P_A and P_B on the tree. P_A and P_B are initially located on vertices a and b respectively. In one step, 378QAQ will do the following in order:

1. Move P_A to a neighboring vertex. If the target vertex is white, this vertex will be painted red.
2. Move P_B to a neighboring vertex. If the target vertex is colored in red, this vertex will be painted blue.

Initially, the vertex a is painted red. If $a = b$, the vertex a is painted blue instead. Note that both the chess pieces **must** be moved in each step. Two pieces can be on the same vertex at any given time.

378QAQ wants to know the minimum number of steps to paint all vertices blue.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^4$). The description of the test cases follows.

The first line of each test case contains one integer n ($1 \le n \le 2 \cdot 10^5$).

The second line of each test case contains two integers a and b ($1 \le a, b \le n$).

Then $n - 1$ lines follow, each line contains two integers x_i and y_i ($1 \le x_i, y_i \le n$), indicating an edge between vertices x_i and y_i. It is guaranteed that these edges form a tree.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output the minimum number of steps to paint all vertices blue.

Standard Input	Standard Output
3 2 1 2 1 2 5 1 2 1 2 1 3 1 4 1 5 8 5 4 7 1 1 5 1 8	2 8 13

8 3 7 2 8 6 3 4	

Note

In the first test case, 378QAQ can paint all vertices blue in the following order:

- Initially, P_A is located on the vertex 1, and P_B is located on the vertex 2. The vertex 1 is painted red and the vertex 2 is white.
- 378QAQ moves P_A to the vertex 2 and paints it red. Then 378QAQ moves P_B to the vertex 1 and paints it blue.
- 378QAQ moves P_A to the vertex 1. Then 378QAQ moves P_B to the vertex 2 and paints it blue.

,

E. Chain Queries

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 256 megabytes

You are given a tree of n vertices numbered from 1 to n. Initially, all vertices are colored white or black.

You are asked to perform q queries:

- "u" — toggle the color of vertex u (if it was white, change it to black and vice versa).

After each query, you should answer whether all the black vertices form a chain. That is, there exist two black vertices such that the simple path between them passes through all the black vertices and only the black vertices. Specifically, if there is only one black vertex, they form a chain. If there are no black vertices, they do **not** form a chain.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^4$). The description of the test cases follows.

The first line of each test case contains two integers n and q ($1 \le n, q \le 2 \cdot 10^5$).

The second line of each test case contains n integers $c_1, c_2, \ldots, c_n$ ($c_i \in \{0, 1\}$) — the initial color of the vertices. c_i denotes the color of vertex i where 0 denotes the color white, and 1 denotes the color black.

Then $n - 1$ lines follow, each line contains two integers x_i and y_i ($1 \le x_i, y_i \le n$), indicating an edge between vertices x_i and y_i. It is guaranteed that these edges form a tree.

The following q lines each contain an integer u_i ($1 \le u_i \le n$), indicating the color of vertex u_i needs to be toggled.

It is guaranteed that the sum of n and q over all test cases respectively does not exceed $2 \cdot 10^5$.

Output

For each query, output "Yes" if the black vertices form a chain, and output "No" otherwise.

You can output "Yes" and "No" in any case (for example, strings "yEs", "yes", "Yes" and "YES" will be recognized as a positive response).

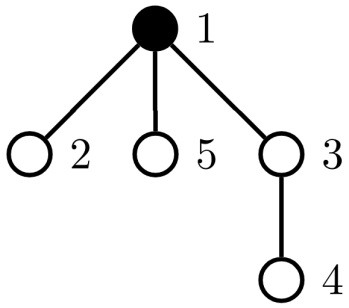
Standard Input	Standard Output
2 2 1 1 0 1 2 1 5 4 1 0 0 0 0 1 2 1 3 1 5	No No Yes Yes No

3 4 4 3 2 5	
4 5 3 1 1 1 1 1 3 5 2 5 3 4 1 5 1 1 1 4 4 0 0 0 0 1 2 2 3 1 4 1 2 3 2 1 1 1 1 1 1 0 1	Yes No Yes Yes Yes Yes No No Yes

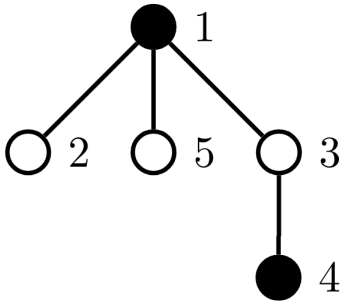
Note

In the second test case, the color of the vertices are as follows:

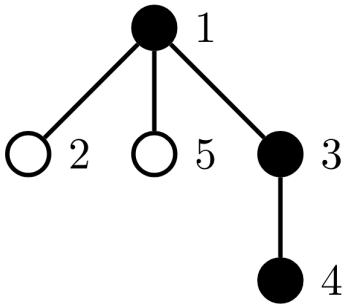
The initial tree:



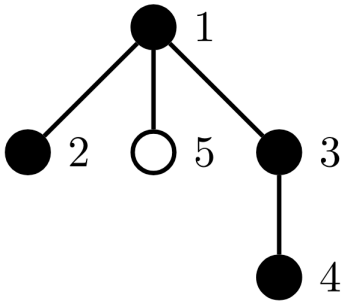
The first query toggles the color of vertex 4:



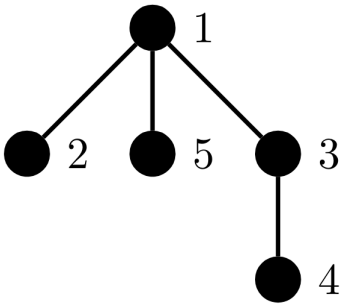
The second query toggles the color of vertex 3:



The third query toggles the color of vertex 2:



The fourth query toggles the color of vertex 5:



,

F. Set

```
Input file:     standard input
Output file:    standard output
Time limit:     2 seconds
Memory limit:   512 megabytes
```

Define the *binary encoding* of a finite set of natural numbers $T \subseteq \{0, 1, 2, \ldots\}$ as $f(T) = \sum_{i \in T} 2^i$. For example, $f(\{0, 2\}) = 2^0 + 2^2 = 5$ and $f(\{\}) = 0$. Notice that f is a bijection from all such sets to all non-negative integers. As such, f^{-1} is also defined.

You are given an integer n along with $2^n - 1$ sets $V_1, V_2, \ldots, V_{2^n-1}$.

Find all sets S that satisfy the following constraint:

- $S \subseteq \{0, 1, \ldots, n-1\}$. Note that S can be **empty**.
- For all **non-empty** subsets $T \subseteq \{0, 1, \ldots, n-1\}$, $|S \cap T| \in V_{f(T)}$.

Due to the large input and output, both input and output will be given in terms of binary encodings of the sets.

Input

The first line of input contains a single integer n ($1 \leq n \leq 20$).

The second line of input contains $2^n - 1$ integers $v_1, v_2, \ldots, v_{2^n-1}$ ($0 \leq v_i < 2^{n+1}$) — the sets V_i given in their binary encoding where $V_i = f^{-1}(v_i)$.

Output

The first line of output should contain an integer k indicating the number of possible S.

In the following k lines, you should output $f(S)$ for all possible S in **increasing order**.

Standard Input	Standard Output
3 15 15 15 15 15 15 12	4 3 5 6 7
5 63 63 63 63 6 63 63 63 63 63 63 5 63 63 63 63 63 63 8 63 63 63 63 2 63 63 63 63 63 63 63	1 19

Note

In the first test case, one possible S is $f^{-1}(3) = \{0, 1\}$. All the non-empty subsets $T \subseteq \{0, 1, 2\}$ and the corresponding $|S \cap T|$, $f(T)$ and $V_f(T)$ are as follows:

T	$\lvert S \cap T \rvert$	$f(T)$	$V_{f(T)}$
$\{0\}$	1	1	$\{0, 1, 2, 3\}$
$\{1\}$	1	2	$\{0, 1, 2, 3\}$

$\{2\}$	0	4	$\{0,1,2,3\}$
$\{0,1\}$	2	3	$\{0,1,2,3\}$
$\{0,2\}$	1	5	$\{0,1,2,3\}$
$\{1,2\}$	1	6	$\{0,1,2,3\}$
$\{0,1,2\}$	2	7	$\{2,3\}$

,

G. Zimpha Fan Club

```
Input file:     standard input
Output file:    standard output
Time limit:     12 seconds
Memory limit:   512 megabytes
```

One day, Zimpha casually came up with a problem. As a member of "Zimpha fan club", you decided to solve that problem.

You are given two strings s and t of length n and m, respectively. Both strings only consist of lowercase English letters, `-` and `*`.

You need to replace all occurrences of `*` and `-`, observing the following rules:

- For each `-`, you must replace it with any lowercase English letter.
- For each `*`, you must replace it with a string of any (possibly, zero) length which only consists of lowercase English letters.

Note that you can replace two different instances of `-` with different characters. You can also replace each two different instances of `*` with different strings.

Suppose s and t have been transformed into s' and t'. Now you're wondering if there's a replacement that makes $s' = t'$.

Input

The first line of input contains two integers n and m ($1 \leq n, m \leq 2 \cdot 10^6$) — the length of the strings s and t, respectively.

The second line contains the string s of length n. It is guaranteed that s only consists of lowercase English letters, `-` and `*`.

The third line contains the string t of length m. It is guaranteed that t only consists of lowercase English letters, `-` and `*`.

Output

For each test case, output "`Yes`" if there is a replacement that makes $s' = t'$, and output "`No`" otherwise.

You can output "`Yes`" and "`No`" in any case (for example, strings "`yEs`", "`yes`", "`Yes`" and "`YES`" will be recognized as a positive response).

Standard Input	Standard Output
10 10 justmonika j-stsayori	No
7 8 ttk-wxx *tt-l-xx	Yes
13 11 asoulwangziji	No

-soulg*z-y-	
7 3 abc*cba a*c	No
20 18 bulijiojio-dibuliduo *li*ji-*ox*i*-du*-	Yes

Note

In the second test case, we can transform both strings into `ttklwxx`. In s, `-` will be replaced with `l`. In t, `*` will be replaced by the empty string with the first and second `-` will be replaced with `k` and `w` respectively.

In the fifth test case, we can transform both strings into `bulijiojioxdibuliduo`.

,

H. 378QAQ and Core

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 256 megabytes

378QAQ has a string s of length n. Define the *core* of a string as the substring[†] with maximum lexicographic[‡] order.

For example, the core of "`bazoka`" is "`zoka`", and the core of "`aaa`" is "`aaa`".

378QAQ wants to rearrange the string s so that the core is lexicographically minimum. Find the lexicographically minimum possible core over all rearrangements of s.

[†] A substring of string s is a continuous segment of letters from s. For example, "`defor`", "`code`" and "`o`" are all substrings of "`codeforces`" while "`codes`" and "`aaa`" are not.

[‡] A string p is lexicographically smaller than a string q if and only if one of the following holds:

- p is a prefix of q, but $p \neq q$; or
- in the first position where p and q differ, the string p has a smaller element than the corresponding element in q (when compared by their ASCII code).

For example, "`code`" and "`coda`" are both lexicographically smaller than "`codeforces`" while "`codeforceston`" and "`z`" are not.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 10^5$). The description of the test cases follows.

The first line of each test case contains a single integer n ($1 \leq n \leq 10^6$) — the length of string s.

The next line of each test case contains the string s of length n. The string s consists of lowercase English letters.

It is guaranteed that the sum of n over all test cases does not exceed 10^6.

Output

For each test case, output the lexicographically minimum possible core over all rearrangements of s.

Standard Input	Standard Output
6 3 qaq 4 cccc 6 bazoka 6 zazzzz 7 ababbbb	qaq cccc z zzz bbababb cbcacbc

7 ccbabcc	

Note

In the first test case, all possible rearrangements and their corresponding cores are as follows:

- "**qaq**", its core is "**qaq**".
- "**aqq**", its core is "**qq**".
- "**qqa**", its core is "**qqa**".

So the core with the minimum lexicographic order in all rearrangement plans is "**qaq**".

,

I. Mind Bloom

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

This is the way it always was.
This is the way it always will be.
All will be forgotten again soon...

Jellyfish is playing a one-player card game called "Slay the Spire". There are n cards in total numbered from 1 to n. The i-th card has *power* c_i.

There is a binary string s of length n. If $s_i = \texttt{0}$, the i-th card is initially in the draw pile. If $s_i = \texttt{1}$, the i-th card is initially in Jellyfish's hand.

Jellyfish will repeat the following process until either her hand or the draw pile is empty.

1. Let x be the power of the card with the largest power in her hand.
2. Place a single card with power x back into the draw pile.
3. Randomly draw x cards from the draw pile. All subsets of x cards from the draw pile have an equal chance of being drawn. If there are fewer than x cards in the draw pile, Jellyfish will draw all cards.

At the end of this process, find the probability that Jellyfish can empty the draw pile, modulo $1\,000\,000\,007$.

Formally, let $M = 1\,000\,000\,007$. It can be shown that the answer can be expressed as an irreducible fraction $\frac{p}{q}$, where p and q are integers and $q \not\equiv 0 \pmod{M}$. Output the integer equal to $p \cdot q^{-1} \bmod M$. In other words, output such an integer x that $0 \le x < M$ and $x \cdot q \equiv p \pmod{M}$.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 100$). The description of the test cases follows.

The first line of each test case contains a single integer n ($1 \le n \le 120$) — the number of cards.

The second line of each test case contains n integers $c_1, c_2, \ldots, c_n$ ($0 \le c_i \le n$) — the powers of the cards. It is guaranteed that $c_1 \le c_2 \le \ldots \le c_n$.

The third line of each test case contains a binary string s of length n. If $s_i = \texttt{0}$, the i-th card is initially in the draw pile. If $s_i = \texttt{1}$, the i-th card is initially in Jellyfish's hand.

It is guaranteed that the sum of n^2 over all test cases does not exceed 120^2.

Output

For each test case, output the probability that Jellyfish can empty the draw pile modulo $1\,000\,000\,007$.

Standard Input	Standard Output
4 5 0 1 1 1 2 00100 3	500000004 0 0 675898154

2 3 3 000 10 0 0 0 0 0 0 0 1 1 1 1111011111 20 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 2 3 3 4 00000000001000101010	

Note

In the first test case, Jellyfish will keep playing cards with power 1 until Jellyfish draws a card with power 0 or power 2. If Jellyfish draws a card with power 0, she will eventually empty her hand. If Jellyfish draws a card with power 2, she will eventually empty the draw pile. Since there is an equal chance of drawing 0 or 2, the answer is $\frac{1}{2}$, and $2 \cdot 500\,000\,004 \equiv 1 \pmod{10^9 + 7}$