The 2nd Universal Cup

Stage 27: India

April 6-7, 2024

This problem set should contain 12 problems on 25 numbered pages.

Based on

International Collegiate Programming Contest (ICPC)

Problem A. Basic Blooms

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

In a hidden realm where numbers dance and bases intertwine, there exists a garden of exquisite beauty known as the Number Garden. Within its vibrant meadows, a unique species of flowers flourishes—the Basic Bloom.

Each Basic Bloom flower contains some petals. Basic Blooms are unique because their number of petals can be expressed as a sequence of a single digit **one or more times**, in **at least one base** between 2 and 16 (inclusive). Digits beyond 9 are represented in lexicographic order of the English alphabet. Specifically, 10 is represented as `a`, 11 as `b`, 12 as `c` and so on.

For example:

- a flower with 14 petals is a Basic Bloom because 14 can be represented as 22 in base 6 or `e` in base 15.
- a flower with 10 petals is a Basic Bloom because 10 can be represented as 11 in base 9.
- a flower with 931 petals is a Basic Bloom because 931 can be represented as 777 in base 11.
- a flower with 1570 petals is a Basic Bloom because 1570 can be represented as `aaa` in base 12.

Notably, and for each positive integer x that can be expressed as a single digit in at least one base between 2 and 16, there is at least one Basic Bloom with x petals. Also, no two Basic Blooms have the same number of petals.

The Guardians of the Garden, seeking to share the wonders of this numerical symphony, have issued a challenge to code wizards across the realms. They seek those who can harness the power of algorithms and traverse the Garden's mathematical pathways.

The challenge is as follows: Arrange all the Basic Bloom flowers in increasing order of number of petals. Given k_1 and k_2, find the **sum of the number of petals** from the k_1th flower to the k_2th flower modulo 998244353.

In other words, let p_i denote the number of petals in the ith flower. Find the sum of p_i for i from k_1 to k_2 modulo 998244353.

Input

The first line of of the input contains an integer t denoting the number of test cases.

Each test case is a single line containing two space-separated integers k_1 and k_2.

- $1 \le t \le 10^6$
- $1 \le k_1 \le k_2 \le 10^6$

Output

For each test case, print a single line containing the answer as described in the statement.

Example

standard input	standard output

3
1 2
1 10
15 2000
	3
	55
	736374621

Note

The first few Basic Blooms have the following number of petals:

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 20,
21, 22, 24, 26, 27, 28, 30, 31, 32, 33, 34, 35, 36, 39, 40, 42, 43,
44, 45, 48, 50, 51, 52, 54, 55, 56, 57, 60, 62, 63, 64, 65, 66, 68,
70, 72, 73, 75, 77, 78, 80, 84, 85, 86, 88, 90, 91, 93, 96, 98, 99,
...]
```

Problem B. Can We Still Qualify For Semifinals?

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

The Cricket World Cup is a highly anticipated event, captivating fans worldwide. During the tournament, a common curiosity arises: Can a specific team still qualify for the semifinals, i.e., the top 4? This problem aims to address such queries.

A total of 10 teams are participating in the Cricket World Cup: India, Australia, England, New Zealand, Pakistan, South Africa, Sri Lanka, Afghanistan, Bangladesh, and the Netherlands. We number the teams 1 to 10, with India being number 1.

The tournament follows a round-robin format, where each team plays against every other team exactly once, spread across a series of phases/rounds. In each round, every team participates in one game. The match selection process ensures that each team faces every other team exactly once, resulting in a total of 45 games.

The rounds of the tournament can be systematically structured using a mathematical approach, described below.

The method for generating the fixtures would be a round-robin method. In phase 1, the teams are initially arranged in ascending order (e.g., [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]). Matches are then formed by pairing the first team with the last team, the second team with the second-to-last team, and so on, until the middle teams meet. In other words, in this phase, the matches would be team 1 vs team 10, 2 vs 9, 3 vs 8, 4 vs 7, 5 vs 6

The subsequent phases involve a circular rotation of the team list, keeping the position of the first team fixed at the start. In each phase, teams are matched with their counterparts in a mirrored fashion. For example, in the second phase, the teams are arranged as [1, 10, 2, 3, 4, 5, 6, 7, 8, 9], where the first team faces the second-to-last team, the second team faces the second-to-second-last team, and so forth. In the third phase, the teams would be arranged as [1, 9, 10, 2, 3, 4, 5, 6, 7, 8].

The exact details of matches in various rounds can also be read from the table below.

Round 1	Round 2	Round 3	Round 4	Round 5	Round 6	Round 7	Round 8	Round 9
1 vs 10	1 vs 9	1 vs 8	1 vs 7	1 vs 6	1 vs 5	1 vs 4	1 vs 3	1 vs 2
2 vs 9	10 vs 8	9 vs 7	8 vs 6	7 vs 5	6 vs 4	5 vs 3	4 vs 2	3 vs 10
3 vs 8	2 vs 7	10 vs 6	9 vs 5	8 vs 4	7 vs 3	6 vs 2	5 vs 10	4 vs 9
4 vs 7	3 vs 6	2 vs 5	10 vs 4	9 vs 3	8 vs 2	7 vs 10	6 vs 9	5 vs 8
5 vs 6	4 vs 5	3 vs 4	2 vs 3	10 vs 2	9 vs 10	8 vs 9	7 vs 8	6 vs 7

The above matches are supposed to happen in order given.

Given the results of first k matches, your task is to determine whether the Indian team still has a chance to qualify for the semifinals.

A team can qualify to semifinals if and only if its number of wins is greater or equal to the fourth highest number of wins among the teams. (The fourth highest number of wins is obtained by sorting the list of 10 numbers of wins, and then taking the fourth one starting from the back.)

The results of first k matches would be given to you in a binary string of length k. Let the ith match be between the team x vs team y. Then, if ith (1-based indexing) character of string is 1, then team x wins, otherwise team y wins.

Input

The first line of the input contains a single integer t corresponding to the number of test cases. The t test cases follow.

Each test case consists of two lines. The first line of each test case contains the integer k. The next line contains a binary string of length k denoting the result of the matches.

- $1 \leq t \leq 10$
- $1 \leq k \leq 45$

Output

For each test case, output a single line containing `YES` or `NO` depending on whether India can still qualify for the semi-finals or not, respectively.

Example

standard input	standard output
3 3 111 25 1000010101111111111010100 35 01111011110111101111011110111101111	YES YES NO

Note

In the first test case, India has got a really good start by winning its first match. India could qualify in the semi-finals. The coolest way to qualify would to be win all its upcoming matches :)

In the second test case, India has won all of its conducted matches already.

In the third test case, India has lost all of its already conducted 7 matches. India has now no way to qualify to the semis.

Problem C. Cheap Construction

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

Blackwater Industries plans on building a space colony in the forest moon of Alpha Centauri A called Pandora, and has hired you for help.

The colony will consist of n domes, each with a positive height. The n domes will be in a single sequence, and we label the positions 1 to n from left to right. Using the latest advances in wormhole technology, it is possible to insert a new dome between any two existing domes, even if there isn't any space available between them! Of course, it is also possible to insert a new dome before the first dome, or after the last dome.

The space colony will be built in n steps starting from an empty sequence. The ith step is described by two integers (p_i, h_i) and it means:

- Insert a new dome with height h_i, so that the newly-inserted dome lands at position p_i after insertion.

More formally, right before the ith step, there will be exactly $i - 1$ domes in the sequence, and the ith operation (p_i, h_i) must satisfy $1 \le p_i \le i$. Then the meaning of the ith operation is:

- If $p_i = 1$, then insert a new dome of height h_i before all currently-existing domes.
- If $p_i = i$, then insert a new dome of height h_i after all currently-existing domes.
- If $1 < p_i < i$, then insert a new dome of height h_i between the domes at locations p_{i-1} and p_i.

Thus, right after the ith step, there will be exactly i domes in the sequence.

Now, for a sequence $S = [p_1, h_1, p_2, h_2, \ldots, p_n, h_n]$ of $2n$ numbers, define $C(S)$ be the sequence of heights of the domes after performing the sequence of operations $(p_1, h_1), (p_2, h_2), \ldots, (p_n, h_n)$. Thus, $C(S)$ is a sequence of n numbers.

Blackwater Industries has drafted an initial plan. Their goal is to build a space colony whose dome heights are equal to $C(P)$, where P is a sequence of $2n$ numbers given as input. However, they're cheap, so they want to build the same space colony but with a lower **cost**!

In the age of space travel, cost is measured differently. For two given sequences P and Q of length $2n$, the sequence with lower cost is the one that is lexicographically smaller.

What is the cheapest Q which produces the same space colony as P? In other words, what is the lexicographically smallest sequence Q of $2n$ numbers such that $C(P) = C(Q)$?

Note: Let A of B be two distinct sequences. Then A is **lexicographically smaller** than B iff at least one of the following conditions hold:

- A is a prefix of B;
- A is not a prefix of B, and if i is the smallest index where they differ, then $A_i < B_i$.

Input

The first line of input contains a single integer t, the number of test cases. The descriptions of t test cases follow.

Each test case consists of multiple lines of input. The first line of each test case contains the integer n. Then n lines follow, where the ith line contains two space-separated integers p_i and h_i. The sequence P is now defined as $[p_1, h_1, p_2, h_2, \ldots, p_n, h_n]$.

- $1 \le t \le 1000$
- $1 \le n \le 500000$
- The sum of all n in a single test file is at most 500000
- $1 \le p_i \le i$
- $1 \le h_i \le n$

Output

For each test case, output n lines where the ith line contains two space-separated integers P_i and H_i. The sequence Q defined as $[P_1, H_1, P_2, H_2, \ldots, P_n, H_n]$ must be the lexicographically smallest sequence such that $C(P) = C(Q)$.

Example

standard input	standard output

1
3
1 1
2 2
1 3
	1 1
	1 3
	3 2

Note

The sequence $P = [1, 1, 2, 2, 1, 3]$ in the output corresponds to the 3 operations $(1, 1), (2, 2), (1, 3)$ which produces the following sequence of heights:

- Initially, the sequence is [] (empty).
- After $(1, 1)$, the heights are $[1]$.
- After $(2, 2)$, the heights are $[1, 2]$.
- After $(1, 3)$, the heights are $[3, 1, 2]$.

Thus, $C(P) = [3, 1, 2]$.

The sequence $Q = [1, 1, 1, 3, 3, 2]$ in the output corresponds to the 3 operations $(1, 1), (1, 3), (3, 2)$ which produces the following sequence of heights:

- Initially, the sequence is [] (empty).
- After $(1, 1)$, the heights are $[1]$.
- After $(1, 3)$, the heights are $[3, 1]$.
- After $(3, 2)$, the heights are $[3, 1, 2]$.

Thus, $C(Q) = [3, 1, 2]$, and we have $C(P) = C(Q)$. Furthermore, one can show that $[1, 1, 1, 3, 3, 2]$ is the lexicographically smallest such sequence.

Problem D. Expected Diameter

Input file:	`standard input`
Output file:	`standard output`
Time limit:	15 seconds
Memory limit:	256 megabytes

If you have some experience preparing problems for a contest, you might find the following fact counterintuitive: a random tree, chosen uniformly from all unrooted labelled trees with n nodes, has expected diameter $\Theta(\sqrt{n})$.

Why is this so unintuitive? Well, because if you've already prepared a tree problem before, then you might know that one of the simplest ways to generate a random tree (not necessarily uniformly random) is the following procedure, which we can call the "lopsided random generator":

- For each i from 2 to n:
  - Choose a j between 1 and $i-1$ randomly, and add the edge (i, j).
- Relabel the nodes by choosing a random permutation of $1, 2, \ldots, n$.

Although this is not at all a uniformly random choice of a unrooted labelled tree with n nodes—recall that there are n^{n-2} such trees by Cayley's formula—you might intuitively think that this is "close enough" to being uniformly random, and should give you the correct expected diameter. And the expected diameter of a tree generated with this procedure is $\Theta(\log n)$ (well, at least I think it is). However, this is incorrect! As it turns out, this procedure gives a highly lopsided probability distribution among the n^{n-2} trees, enough to change the expected diameter.

Let's put this newfound knowledge to the test. Suppose a random weighted, unrooted labelled tree with n nodes is chosen as follows:

- First, choose an unweighted unrooted labelled tree with n nodes uniformly randomly from the n^{n-2} such trees.
  - **Important Note:** The choice of unweighted unrooted labelled tree is **uniform** across the n^{n-2} distinct such trees; the "lopsided random generator" procedure described above will not be used.
- Next, for each edge, give it a weight of 1 with probability p_1, and 2 with probability p_2. Note that $p_1 + p_2 = 1$.

Given n, what is the expected diameter of a tree chosen this way? Find this number "modulo 998244353"; that is:

- Let $m = 998244353$.
- It can be shown that the answer is rational (assuming p_1 and p_2 are).
- Write the answer as u/v in lowest terms and with v positive.
- It can be shown (under the constraints of this problem) that there's a unique integer r such that $0 \le r < m$ and $rv \equiv u$ modulo m. Your goal is to find this unique r.

Notes:

- An **unrooted labelled tree with n nodes** is a connected acyclic undirected graph whose nodes are $1, 2, \ldots, n$.
- The **diameter** of a weighted graph is the largest weight of any simple path in it.
- A **simple path** is a path with no repeated nodes; that is, a sequence of distinct nodes $s_0, s_1, \ldots, s_k$ such that there is an edge (s_i, s_{i+1}) for each i.
- The weight of a simple path is the sum of the weights of the edges in it.

Input

The input consists of a single line containing three space-separated integers n, x and y. The probabilities p_1 and p_2 are now defined as:

$$p_1 = x/y$$
$$p_2 = 1 - p_1$$

- $1 \le n \le 2000$
- $0 \le x \le y \le 1000$

Output

Output a single line containing the integer denoting the answer.

Examples

standard input	standard output
2 1 3	665496237
3 2 3	665496238

Note

For $n = 2$, there is only $2^0 = 1$ unweighted tree, and 2 weighted trees, each with diameters 1 and 2, with probabilities $1/3$ and $2/3$, respectively. The expected diameter is then $5/3$, and the answer is $r = 665496237$ because it is the unique integer satisfying:

- $0 \le r < 998244353$
- $3r \equiv 5$ modulo 998244353.

For $n = 3$, there are $3^1 = 3$ unweighted trees, and 12 weighted trees.

Problem E. Generalized Collatz Conjecture

Input file:	`standard input`
Output file:	`standard output`
Time limit:	12 seconds
Memory limit:	1024 megabytes

You are a bright-eyed young undergrad who has ~~watched a Youtube video about~~ done extensive research into the Collatz conjecture, and have been struck with inspiration for several *brilliant* ideas of how to solve it! You can picture it now—for your undergraduate thesis, you will solve the Collatz conjecture, cementing your name in the textbooks as a genius on the level of von Neumann, Terence Tao, and Ramanujan. Your thesis advisor thinks this is a bad idea; they warn you that many others before you have tried and failed taking this path. But they just don't get it—that's them, and you're you. This is *different. You're special.*

Due to your persistence, your thesis advisor reluctantly agrees to support you, but *on the condition* that you prove yourself by solving this problem featuring an **even more difficult and generalized version** of the setup in the Collatz conjecture!

You are given a set M of distinct integers and an integer n. Your goal is to *transform* n into 1 using only operations of the following two kinds:

1. Choose an $m \in M$, and replace $n \to mn + 1$
2. Choose a prime factor p of n, and replace $n \to n/p$

What is the minimum number of operations needed to transform n into 1? If it is impossible to transform n to 1, say so as well.

Input

The first line of input contains an integer t, the number of test cases. The descriptions of t test cases follow.

Each test case consists of one line containing $2 + |M|$ space-separated integers, where $|M|$ denotes the size of M. The first two integers are n and $|M|$, and the remaining $|M|$ ones are the elements of M.

- $1 \le t \le 262144$
- $2 \le n \le 2097152$
- $1 \le |M| \le 8$
- $1 \le m \le 64$ for each m in M
- No two cases in each file are exactly the same.
- The elements of M are given in increasing order.

Output

For each test case, output one line containing either:

- a single integer denoting the minimum number of operations needed to transform n to 1, or
- the string `FIELDS MEDAL` if it is impossible to transform n to 1.

Important Note: The output is **case-sensitive**, so you need to output in all-capital letters. Also, don't put leading or trailing spaces, two consecutive spaces, or tabs, in the output.

Example

standard input	standard output

2
84 2 3 6
18588 3 18 25 44
	3
	4

Note

- In the first test case, one possible sequence of 3 operations could be: $84 \to 12 \to 37 \to 1$.
- In the second test case, one possible sequence of 4 operations could be: $18588 \to 12 \to 301 \to 5419 \to 1$.

Problem F. Idola-Tree

Input file:	`standard input`
Output file:	`standard output`
Time limit:	9 seconds
Memory limit:	256 megabytes

The idol group Oshikoshi requests that you help design mathematical art for their new album cover.

The album cover is to be a **tree**—it will showcase the n idols, with $n-1$ curves that each bidirectionally connect a different pair of idols. A simple path is a sequence of two or more distinct idols, such that there exists a curve directly connecting any two adjacent idols in the sequence; as it is a tree, there exists a path between any two idols u and v in the tree, and we can show that such a path is unique. Thus, you can verify that any tree with n idols will have $n(n-1)/2$ distinct simple paths (if we count the path from u to v to be "the same as" the path from v to u).

Oshikoshi gives you some more definitions!

- Each curve's length is some positive integer.
- The "ink cost" of the entire tree is equal to the sum of the lengths of all of its $n-1$ curves.
- The length of a simple path is equal to the sum of the lengths of all curves along that path.
- The squared length of a path is the value you get by taking the length of a path and then squaring it
  - Note that the sum of the lengths of the curves along a path is squared, **not** the individual lengths (e.g. in a path with curves of length 3 and 4, what we want is $(3+4)^2$ and **not** 3^2+4^2).
- The "drama" of a tree is equal to the sum of the squared lengths of all $n(n-1)/2$ distinct simple paths in the tree.

Now, the "shape" of the tree has already been decided (i.e. which idols are connected by the $n-1$ curves), but the length to make each curve has not yet been set in stone.

Here was your original job: Given an integer c, consider all the different ways to assign a positive integer length to each curve in the tree, such that the ink cost is exactly equal to c. Among all such ways, find one which minimizes the drama of the tree; let this minimum drama value be denoted by $m(c)$.

But Oshikoshi wants to mess with you, so they ask you the following question instead: Given an integer C, what is the sum of $(m(c))^3$ across all integers c from $n-1$ to C (inclusive)? Find this number modulo 998244353.

Input

The first line of input contains t, the number of test cases. The descriptions of t test cases follow.

The first line of each test case contains two space-separated integers n and C. Then $n-1$ lines follow, each containing two space-separated integers between 1 and n denoting a pair of idols directly connected by a curve. We number the idols 1 to n.

- $1 \le t \le 4$
- $2 \le n \le 3 \cdot 10^5$
- $n - 1 \le C \le 5 \cdot 10^7$

Output

For each test case, output a single line containing a single integer denoting the answer for that test case.

Example

standard input	standard output

2
4 3
1 4
1 3
2 1
4 4
1 4
1 3
2 1
	3375
	25327

Note

Both sample test cases feature the same tree.

- For $c = 3$, there is only one way to assign positive integer lengths to all 3 curves such that the ink cost is exactly 3, and that is to assign 1 to each of them.
  - The drama is then $m(3) = 1^2 + 1^2 + 1^2 + 2^2 + 2^2 + 2^2 = 15$.
- For $c = 4$, one way to achieve the minimum drama is to assign a length of 2 to the curve connecting idols 1 and 3, and assign a length of 1 to the remaining two curves.
  - The drama is then $m(4) = 1^2 + 2^2 + 1^2 + 3^2 + 2^2 + 3^2 = 28$.

Thus,

- the answer to the first sample case is $153 \bmod 998244353 = 3375$; and
- the answer to the second sample case is $(153 + 283) \bmod 998244353 = 25327$.

Problem G. Palworld

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	256 megabytes

There's a hot new game sweeping the world right now: Palworld. In it, your goal is to create the longest **pal**indrome. (It's a puzzle game!)

When the game starts, there is a string S consisting of n lowercase English letters. You are also given an integer k. As a player, you have the following action, which you must perform **exactly once**:

- Choose an index i ($0 \leq i \leq n$) and insert **at most** k characters after index i of S. Choosing $i = 0$ means you append up to k characters in front of S.

Your score in the game is equal to the length of the longest substring of the resulting string that is a palindrome.

What is the maximum possible score you can get?

Notes:

- A **palindrome** is a string that reads the same forwards and backwards.
- A **substring** of a string is a string obtained by deleting some number of letters (possibly none) in front and/or at the back.

Input

The first line contains t, the number of test cases.

Each test case consists of two lines.

- The first line contains n and k, the length of S and the number of characters you're allowed to insert, respectively.
- The second line contains the string S.

- $1 \leq t \leq 10^4$
- $1 \leq n \leq 2 \cdot 10^5$
- $1 \leq k \leq 100$
- S contains only the characters a through z.
- The sum of n across all test cases does not exceed $2 \cdot 10^5$

Output

For each test case, print a single line containing a single integer: the maximum possible length of the longest palindromic substring of S after the operation.

Example

standard input	standard output
4 1 3 a 4 1 icpc 4 2 icpc 8 4 icecream	4 5 5 11

Note

- In the first example, you can insert 3 characters to turn S into `"abba"`, which is a palindrome of length 4.
- In the second example, it's optimal to append i to S, forming `"icpci"`, which is a palindrome of length 5.
- In the third example, even though we're allowed to insert 2 characters, we're unable to attain a longer palindrome.
- In the fourth example, one optimal final string is `"imaercecream"`, where the substring `"maercecream"` is a length-11 palindrome.

Problem H. Passing Game

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

There are n footballers numbered $1, 2, \ldots, n$, on a line. The ith is on the position x_i. Initially, the footballer 1 has a football. The footballers are not allowed to move, but they can pass the football to one another. They know the positions of each other. Their aim is to make the ball reach the footballer n in minimum time possible. Different footballers have different kicking speeds. Formally, the ith footballer can kick the ball at a speed of $1/s_i$.

In other words, it takes $s_i \cdot |x_i - x_j|$ time for a direct pass from footballer i to reach footballer j. They are not allowed to change the direction of the ball more than k times. The direction of the ball is said to change in a pair of consecutive direct passes $a \to b$ followed by $b \to c$ if a and c are on the same side of b (both to the left or both to the right). Note that the first pass does not count as a change of direction.

What is the minimum time required for the ball to reach footballer n?

Input

The first line of input contains an integer T, the number of test cases. The descriptions of T test cases follow.

The first line of each testcase contains n and k, the number of footballers, and the maximum number of times they are allowed to change the direction of the ball.

The second line contains n space separated integers, $x_1, x_2, \ldots, x_n$, denoting the positions of the footballers.

The third line contains 'n' space separated integers, $s_1, s_2, \ldots, s_n$, where 's[i]' denotes the inverse of the speed with which the footballer 'i' can kick the football.

- $1 \le T \le 10^5$
- $1 \le n \le 3 \times 10^5$
- $0 \le k \le n$
- The sum of n over all test cases does not exceed 3×10^5.
- $1 \le x_i, s_i \le 10^9$ for all $1 \le i \le n$
- For $i \ne j$, $x_i \ne x_j$

Output

For each test case, print the minimum time required to make the ball reach the footballer n, on a new line.

Example

standard input	standard output

2
4 2
3 2 1 6
3 1 1 3
2 0
1 2
1 2
	7
	1

Note

In the first test case, it is optimal to pass the ball from player 1 to player 2, and then from player 2 to player 4. Note that, the direction of the ball is changed exactly once. The time taken is $3 \cdot (3-2) + 1 \cdot (6-2) = 7$.

In the second test case, there is only one solution, pass the ball directly from player 1 to player 2.

Problem I. Slothful Secretary

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	256 megabytes

A school has n classrooms, labeled 1 to n. Each pair of rooms has exactly one corridor directly connecting them, but for ease of traffic, these corridors are one-way for students.

The school's secretary has to deliver a memo to the class president of each classroom (each classroom has exactly one). He has a brilliant idea that helps him avoid work—he can give multiple memos to some class president, and task that class president with doing the delivery for him. A class president can deliver a memo to some other classroom if they can walk through the corridors to reach that other classroom and then walk through the corridors to return to their original classroom. A class president is willing to do as many deliveries as is requested of them—again, so long as it is possible for them to return to their original classroom when all is done.

A set of class presidents is called complete if the secretary can distribute the memos among the class presidents in this set, and task them with deliveries such that every classroom can receive a memo. Find the minimum possible size of a complete set of class presidents.

Actually, you must process q updates. In the initial setup, for each pair (i, j) such that $1 \leq i < j \leq n$, the one-way corridor connecting classrooms i and j is oriented from i to j (that is, you can travel from the lower-numbered room to the higher one). Then, for each update, the direction of a one-way corridor is reversed—give the minimum possible size of a complete set of class presidents after each update.

Input

The first line contains two integers n and q.

The kth of the next q lines contains two integers u_k and v_k. The kth update reverses the direction of the one-way corridor connecting u_k and v_k.

- $1 \leq n, q \leq 5 \cdot 10^5$
- $1 \leq u_k < v_k \leq n$

Output

For each update, print one line containing a single integer denoting the answer right after the update.

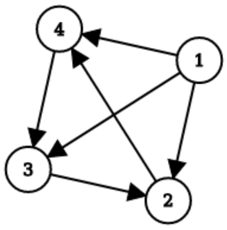
Example

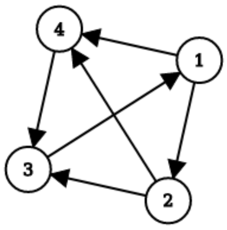
standard input	standard output

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

Note

Please refer following images to see how the classroom pathway looks like after second and fourth updates, respectively:





- After the second update, the secretary can deliver 3 memos to the president of class 3, and 1 memo to the president of class 1. The president of class 3 can deliver memos to the presidents of classes 2 and 4 and then come back to class 3.
- After the fourth update, the secretary can deliver 4 memos to the president of class 4. The president of class 4 can deliver memos to all the other presidents and then come back to class 4.

Problem J. Swirly Sort

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

You're given an array A containing n integers, and an integer k ($1 \le k \le n$). You would like to transform it into a sorted array.

You can perform the following operations on A any number of times:

- Choose k indices $1 \le i_1 < i_2 < i_3 < \ldots < i_k \le n$ and cyclically shift the values at these indices.
  - That is, $A[i_1]$ moves to index i_2, $A[i_2]$ moves to index i_3, $\ldots$, $A[i_k]$ moves to index i_1.
  - Note that the indices you choose for the cyclic shift must always be increasing.
  - This operation has cost 0.
- Choose an index i, and either increment or decrement $A[i]$ by 1.
  - This operation has cost 1.

Find the minimum total cost of operations to transform the array into a sorted array.

Notes:

- The array A is one-indexed.
- An array X is sorted if $X_1 \le X_2 \le X_3 \le \ldots$.

Input

The first line of input contains a single integer t, denoting the number of test cases.

Each test case consists of two lines of input.

- The first line contains two space-separated integers n and k.
- The second line contains n space-separated integers $A_1, A_2, \ldots, A_n$, the initial values of array A.

- $1 \le t \le 10^5$
- $1 \le n \le 3 \cdot 10^5$
- $1 \le k \le n$
- $1 \le A[i] \le 10^9$
- The sum of $n \cdot k$ across all test cases is $\le 3 \cdot 10^5$.

Output

For each test case, print a line containing an integer: the minimum cost to transform A into a sorted array.

Example

standard input	standard output

4
4 1
6 4 3 7
4 2
6 4 3 7
4 3
6 4 3 7
4 4
6 4 3 7
	3
	0
	1
	2

Note

In all samples, the initial array is $A = [6, 4, 3, 7]$.

- For $k = 1$, we subtract 2 from the first element and add 1 to the third element, turning the array into $A = [4, 4, 4, 7]$.
- For $k = 2$, we can choose $i_1 = 1$ and $i_2 = 3$, transforming the array into $[3, 4, 6, 7]$ which is sorted. This has a cost of 0.
- For $k = 3$, the following process is optimal:
- Choose $i_1 = 1$, $i_2 = 2$, $i_3 = 3$. The array becomes $[3, 6, 4, 7]$.
- Choose $i_1 = 1$, $i_2 = 2$, $i_3 = 3$ again. The array becomes $[4, 3, 6, 7]$.
- Subtract 1 from the first element to obtain $[3, 3, 6, 7]$.
- For $k = 4$, the following is optimal:
- Add 1 to the first element. The array becomes $[7, 4, 3, 7]$.
- Subtract 1 from the second element. The array becomes $[7, 3, 3, 7]$.
- Choose all four elements and cyclic shift to obtain $[7, 7, 3, 3]$.
- Cyclic shift again and obtain $[3, 7, 7, 3]$.
- Cyclic shift again and obtain $[3, 3, 7, 7]$.

Problem K. Three Person Tree Game

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

Alice and Bob are tired of playing 2-player games so they called their third friend Chaithanya and decide to play a 3-player game. We abbreviate their names with A, B and C respectively.

A, B and C are playing on a tree with N vertices (vertices are numbered from 1 to N). Recall that a tree is a connected graph with no cycles.

A, B and C are initially each standing on three **distinct** vertices. They take turns to play their move with A playing first, B playing second and C playing third.

When it's a players turn to move, they can do one of the following:

- Stay on the same vertex.
- Move to an adjacent vertex.

The game ends when one of the below conditions are met:

- C and A are in the same node. In this case A wins.
- A and B are in the same node. In this case B wins.
- B and C are in the same node. In this case C wins.

The **primary** objective of each player is to win the game. If that is not possible then their **secondary** objective is to not let anybody else win. All players play optimally.

Given the tree and the initial position of A, B and C, you have to decide if the game will continue forever or if there will be a winner. If there will be a winner then output the name of the winner, else output `DRAW`.

Input

The first line contains a single integer T denoting the number of test cases.

For each test case:

- The first line contains a single integer N denoting the number of people.
- The next line contains three space separated integers a, b, c denoting the vertex number of A, B and C respectively.
- The next $N - 1$ lines contains two space separated integers u, v denoting that there is an edge between vertex u and v. It is guaranteed that the edges form a valid tree.

- $1 \le T \le 3 \cdot 10^4$
- $3 \le N \le 2 \cdot 10^5$
- $1 \le u, v \le N$
- $1 \le a, b, c \le N$
- $a \ne b$ and $a \ne c$ and $b \ne c$ ($\{a, b, c\}$ are pairwise distinct)

- The edges form a valid tree.
- Sum of N over all test cases in a test file does not exceed $2 \cdot 10^5$

Output

For every test case:

- If A wins then output `A`.
- If B wins then output `B`.
- If C wins then output `C`.
- If there is no winner then output `DRAW`.

Note that the output is **case sensitive**.

Example

standard input	standard output

2
3
1 2 3
2 1
3 1
4
1 2 3
1 4
2 4
3 4
	A
	DRAW

Note

- For the first test case, A is at vertex 1, B is at vertex 2 and C is at vertex 3. There is an edge between vertex 1 and 3, so for the first move A will move to vertex 3 where C is located and win the game.
- For the second test case, again A is at vertex 1, B is at vertex 2 and C is at vertex 3. In this case every person has two options for the first turn: 1) either stay at the same vertex or 2) move to vertex 4. For the second option, if anyone moves to vertex 4 then in the next turn one of the other people will move to vertex 4 and win the game. Therfore no one will move to vertex 4 and the game will never end.
  - For example let's say A andB stay in the same vertex for their turns. C decides to move to vertex 4 on his turn. In the next turn it's A's move to play. A will definitely move to vertex 4 and win the game. According to the objective of every player, they don't want anyone else to win. So C will not move to vertex 4. Similar argument can be made for players A and B.

Problem L. Symphony in c++ major

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	256 megabytes

Making new songs is hard, so we asked an AI (ChatDoReMi) to create some new songs for us. The results... are pretty horrible! It turns out that LLMs are totally not-suited for composing music. Who could have guessed? But maybe we can use our own creativity and salvage something out of it.

A string is called **symphonic** if it can be produced by concatenating some (possibly empty) sequence of notes, where each note is one of “do”, “re”, “mi”, “fa”, “so”, “la”, “ti” (without the single quotes, of course). So, for example:

- “dodo”, “doremifasolatido”, “redososomiso”, “doremitiladoremisofadoredo” and “” (the empty string) are all symphonic;
- “dod”, “r”, “soda”, and “icpcamritapuri” are **not** symphonic.

Given a “reference string” s that was provided to us by ChatDoReMi, you must be able to measure how far from being symphonic certain substrings of s are — and actually, you should also be able to handle some updates, too, since we might ask ChatDoReMi to refine its response.

Formally, you must be able to process q queries, each of which is one of two kinds:

- “? i j” means that you must consider the substring that starts at position i and ends at position j, inclusive (this is 1-indexed). What is the minimum number of letters that must be deleted from this substring in order to make it symphonic? Note that each query is an independent and hypothetical question — no letters actually get deleted.
- “# i j *new*” means that you must replace the i-th through j-th characters of s (also inclusive and 1-indexed) with the string *new*; it is guaranteed that $|new| = j - i + 1$.

Input

The first line of input contains two space-separated integers n and q, where n is the length of s.

The second line contains the string s.

Then q lines follow, describing the queries. Each query consists of a single line that is of the form “? i j” or “# i j *new*”, as described above.

- $1 \le n \le 5 \cdot 10^5$
- $1 \le q \le 3 \cdot 10^5$
- For each query, $1 \le i \le j \le n$
- The total length of *new* across all “#” queries is $\le 10^5$
- Each string consists of lowercase English letters.

Output

For each “?” query, output a single line containing a single integer denoting the answer.

Example

standard input	standard output
8 10 eldorado ? 1 3 ? 1 8 # 6 7 it ? 1 8 # 3 3 t ? 1 8 # 1 8 streamer ? 1 8 # 1 8 symphony ? 1 8	3 4 6 6 6 6

Note

- For the first query, the substring in consideration is "eld". We must remove all three letters (turning it into the empty string) to make the string symphonic.
- For the second query, the substring in consideration is "eldorado". We must remove at least 4 letters to make the string symphonic: eldorado $\to$ eldordo $\to$ eldodo $\to$ ldodo $\to$ dodo.
- After the third query, s becomes "eldorito".
- For the fourth query, the substring in consideration is "eldorito". We must remove at least 6 letters to make the string symphonic: eldorito $\to$ eldorto $\to$ eldoro $\to$ eldoo $\to$ edoo $\to$ edo $\to$ do.
- After the fifth query, s becomes "eltorito".
- For the sixth query, the substring in consideration is "eltorito". We must remove at least 6 letters to make the string symphonic: eltorito $\to$ eltorio $\to$ eltrio $\to$ eltri $\to$ etri $\to$ tri $\to$ ti.