Problem List (v1.1)

Problem A
Power Tower
Time limit: 1 second

In a futuristic city built on the concept of mathematical precision, there exists a grand tower with layers of cubes stacked into a perfect 3D matrix. Each cube, or "block," represents a crucial piece of the city's power grid. The matrix is initialized so that every block contains a power level 0. The first block is defined by the coordinates $(1, 1, 1)$, and the last block is defined by the coordinates (n, n, n).

The city's engineers perform two types of operations to maintain the power distribution across the tower:

- **UPDATE Task:** When a specific block at coordinates (x, y, z) requires a power adjustment, the engineers use the following instruction to set the power level of block (x, y, z) to a new value W: `UPDATE x y z W`
- **QUERY Task:** To assess the total power in a section of the tower, the engineers use the instruction `QUERY x1 y1 z1 x2 y2 z2`. The system needs to calculate the sum of the power levels of all blocks where the x-coordinates lie between x_1 and x_2 (inclusive), the y-coordinates lie between y_1 and y_2 (inclusive), and the z-coordinates lie between z_1 and z_2 (inclusive).

Using these two operations, the engineers ensure that the city's power tower remains balanced and operational, efficiently managing the power grid.

Input

- The first line contains an integer T, the number of test cases.
- For each test case:
  - The first line contains two space-separated integers, n and m, n defines the $n \times n \times n$ cube, m defines the number of operations.
  - The next m lines contain operations in one of the two forms:
    1. `UPDATE x y z W`
    2. `QUERY x1 y1 z1 x2 y2 z2`

Output

Print the result of each instruction query on a new line.

Sample Input	Sample Output
2 4 5 UPDATE 1 1 1 5 QUERY 1 1 1 3 3 3 UPDATE 1 1 1 23 QUERY 2 2 2 4 4 4 QUERY 1 1 1 3 3 3 2 4 UPDATE 2 2 1 1 QUERY 1 1 1 1 1 1 QUERY 1 1 1 2 2 2 QUERY 2 2 2 2 2 2	5 0 23 0 1 0

Constraints

- $1 \leq T \leq 50$; $1 \leq n \leq 100$; $1 \leq m \leq 10^5$; $-10^9 \leq W \leq 10^9$

Problem B
Connectopolis
Time limit: 1 seconds

In a bustling city known as **Connectopolis**, there exists a sprawling network of streets and avenues, forming a unique map of relationships between various locations. The city consists of n nodes, each representing a specific landmark, numbered from 1 to n. Each landmark i is assigned an integer c_i that denotes its significance in the city.

The city's street network is designed so that any landmark can be reached from any other landmark either directly or indirectly through a series of streets, and there are exact $n-1$ streets in the city. It is ensured that no landmark is isolated. This guarantee allows the city planners to traverse the entire city and explore its hidden connections easily.

The city planners often seek to uncover hidden connections within this urban landscape. They perform inquiries to explore the relationships between different landmarks and their significance. Each inquiry takes the form $w\ x\ y\ z$, and it is essential to find pairs of landmarks that satisfy specific conditions to uncover the true potential of Connectopolis.

To process an inquiry, you must count the number of ordered pairs of integers (i, j) such that the following conditions are satisfied:

1. $i \neq j$ (the landmarks must be different).
2. i is in the path from landmark w to landmark x (the route from w to x must traverse landmark i).
3. j is in the path from landmark y to landmark z (similarly, the route from y to z must pass through landmark j).
4. $c_i = c_j$ (the significance values at landmarks i and j must be equal).

As you navigate the city of Connectopolis, you will encounter a series of inquiries, each demanding your expertise to decode the secrets held within its streets.

Input

- The first line contains two space-separated integers n (the number of landmarks) and q (the number of inquiries).
- The second line contains n space-separated integers, representing the significance values of each landmark (i.e., $c_1, c_2, \ldots, c_n$).
- Each of the next $n-1$ lines contains two space-separated integers u and v, defining a bidirectional path between landmarks u and v.
- Each of the q subsequent lines contains an inquiry in the form of $w\ x\ y\ z$, as defined above.

Output

For each inquiry, print the number of valid pairs (i, j) that satisfy the conditions listed above on new line.

Sample Input	Sample Output
10 5 2 4 3 9 9 2 4 3 9 9 1 2 1 3 3 4 2 5 3 6 2 7 7 8 3 9 2 10 1 3 5 9 2 8 6 3 9 8 2 7 4 6 1 9 5 3 9 8	0 1 2 2 3

Constraints

- $1 \leq n \leq 10^5$; $1 \leq q \leq 50000$; $1 \leq c_i \leq 10^9$
- $1 \leq u, v, w, x, y, z \leq n$

Problem C
Marble Sorting
Time limit: 1 second

To develop logical thinking for prodigies, Bom has created a game with n marbles. The marbles are arranged in a single horizontal row, each with a unique color corresponding to an integer value from 1 to n. In other words, the color of the n marbles is a permutation of the numbers from 1 to n.

The game aims to reorder the marbles according to a required order by swapping any two marbles. To increase the difficulty, each marble of color i has a movement cost of w_i. The cost to swap two marbles of color i and j is $w_i + w_j$.

Find a way to reorder the marbles with the minimum total cost.

Input

- The first line contains an integer n ($1 \leq n \leq 10^6$).
- The second line contains n integers $w_1, w_2, \ldots, w_n$ ($1 \leq w_i \leq 10000$), representing the movement cost of marbles of each color.
- The third line contains n integers $a_1, a_2, \ldots, a_n$, representing the initial order of the marbles by their color.
- The fourth line contains n integers $b_1, b_2, \ldots, b_n$, representing the desired order of the marbles by their color.

Output

Output a single integer representing the minimum cost to reorder the marbles.

Sample Input	Sample Output
6 8 6 3 8 4 9 1 4 5 3 6 2 5 3 2 4 6 1	33

Explanation

- Step 1: Swap marble of color 2 and marble of color 5 with a cost of $6 + 4 = 10$.
- Step 2: Swap marble of color 3 and marble of color 4 with a cost of $3 + 8 = 11$.
- Step 3: Swap marble of color 1 and marble of color 5 with a cost of $8 + 4 = 12$.
- Total cost: 33.

Problem D
Robot
Time limit: 1 second

Given a grid of size $n \times m$, the rows are numbered from 1 to n, and the columns are numbered from 1 to m. There are k obstacles placed on the grid.

A robot is located on the grid, initially at position s. At any given moment, it can move in three ways:

- If the space in front of it is not blocked by an obstacle or does not go out of bounds, it can move forward by one unit in its current direction.
- The robot can turn 90 degrees to the left or right.
- The robot can turn 180 degrees (turn around to face the opposite direction).

Initially, the robot can choose any direction to move.

There are q queries, and each query asks for the minimum number of direction changes required for the robot to move from position s to position t_i. (Only actions 2 and 3 count as direction changes; choosing the initial direction does not count as a change). For each query, the robot can choose a new initial direction.

Input

- The first line contains four integers n, m, k, and q, representing the number of rows, columns of the grid, the number of obstacles, and the number of queries.
- The next k lines each contain two positive integers x_i, y_i, representing the positions of the obstacles.
- The following line contains two positive integers x_s, y_s, representing the starting position of the robot.
- The next q lines each contain two positive integers x_{t_i}, y_{t_i}, representing the destination coordinates for each query.

Output

Print q lines, each containing a single integer. This integer represents the minimum number of direction changes required to move from the starting position s to the destination position t_i. If reaching the destination from the starting position is impossible, output -1 for that query.

Sample Input	Sample Output
3 4 3 9 1 2 2 1 3 3 3 4 1 1 1 3 1 4 2 2 2 3 2 4 3 1 3 2 3 4	-1 1 0 1 1 0 3 2 0
3 3 0 9 2 2 1 1 1 2 1 3 2 1 2 2 2 3 3 1 3 2 3 3	1 0 1 0 0 0 1 0 1
5 7 4 6 1 4 1 6 2 5 5 4 1 1 1 5 2 4 2 7 4 1 4 6 5 7	-1 1 2 0 1 2

Constraints

- $1 \leq n, m \leq 10^9$; $0 \leq k \leq 50000$; $1 \leq q \leq 10^5$
- $1 \leq x_i, x_s, x_{t_i} \leq n$; $1 \leq y_i, y_s, y_{t_i} \leq m$
- The data guarantees that the starting and destination positions do not have obstacles, and the obstacle positions are distinct.

Problem E
Biology Test
Time limit: 1 second

Biologists have discovered some new information about species evolution, which supports Darwin's theory that each species has exactly one direct ancestor.

Species u is considered an ancestor of species v if there exists a sequence of species $u = x_1, x_2, \ldots, x_k = v$ such that x_i is the direct ancestor of x_{i+1} for all $i = 1, 2, \ldots, k-1$. There are n species, and biologists have collected $n-1$ pieces of ancestor information. All information is consistent with the rule that if u is an ancestor of v, then v cannot be an ancestor of u; in other words, there are no cycles in the ancestry relationships. And no two different species are the direct ancestor of the same species.

The biologists now want to check the ancestor relationship between some pairs of species. Given the ancestral information and a set of species pairs, for each pair (x, y), you need to determine if species x is an ancestor of species y.

Input

- The first line contains an integer n — the number of species.
- The next $n-1$ lines contain two integers a and b, indicating that species a is the direct ancestor of species b.
- The next line contains an integer T — the number of species pairs to check.
- The next T lines contain two integers x and y, representing the species pair to check if x is an ancestor of y.

Output

For each of the T pairs (x, y), output "Yes" if species x is an ancestor of species y, and "No" otherwise.

Sample Input	Sample Output
7 4 7 2 6 3 4 1 2 4 1 7 5 2 4 6 7 4	Yes No

Constraints

- $1 \leq n \leq 10^5$
- $1 \leq T \leq 10$

Problem F
Riband Game
Time limit: 1 second

Thuan and An are childhood friends who recently invented the following game: They have a sequence of n unit squares, numbered from 0 to $n-1$ from left to right. Each player has m ribands, each of length L and width 1 unit. Thuan's ribands are red, and An's ribands are blue.

The two players take turns performing $2m$ actions alternately, with Thuan going first, followed by An. On each player's turn, they choose a position x and place their riband in the sequence, with the following conditions:

- Position x is valid if, starting from x and extending to the right, no square contains ribands of different colors, and no two adjacent squares contain ribands of different colors.
- If position x is valid, the riband will be placed; otherwise, the riband will be discarded.

Task:

1. Calculate how many times each player selects an invalid position.
2. After performing all $2m$ actions, determine the maximum length of consecutive squares occupied by each color of riband.

Input

- The first line contains an integer C ($C = 1$ or $C = 2$).
- The second line contains three integers n, m, and L.
- The next $2m$ lines contain the positions x selected by Thuan and An.

Output

- If $C = 1$, output the number of invalid selections made by Thuan and An.
- If $C = 2$, output the maximum length of consecutive squares occupied by the red riband and the blue riband.

Constraints

- $1 \leq n \leq 10^9$
- $1 \leq m \leq 50000$
- $1 \leq L \leq 20000$

Sample Input	Sample Output
1 20 4 3 9 15 2 13 5 17 0 12	0 1
2 20 4 3 9 15 2 13 5 17 0 12	8 7

Explanation

In the first example:

- Turn 1: Thuan places a red riband on squares 9, 10, 11.
- Turn 2: An places a blue riband on squares 15, 16, 17.
- Turn 3: Thuan places a red riband on squares 2, 3, 4.
- Turn 4: An places a blue riband on squares 13, 14, 15.
- Turn 5: Thuan places a red riband on squares 5, 6, 7.
- Turn 6: An places a blue riband on squares 17, 18, 19.
- Turn 7: Thuan places a red riband on squares 0, 1, 2.
- Turn 8: An selected an invalid position.

  Result: Thuan made 0 invalid selections, while An made 1 invalid selection.

Problem G
Fibonacci Strings
Time limit: 1 second

The weight of a binary string x is defined as the length of x multiplied by the number of 1-bits in x. The beauty of a binary string s is the total weight of all its substrings. For example, the string 101 has a beauty of 16 because it has 8 substrings {empty string, 1, 0, 1, 10, 11, 01, 101} with corresponding weights {0, 1, 0, 1, 2, 4, 2, 6}. Compute the beauty of the n-th Fibonacci string, where the Fibonacci strings are defined as follows:

$$f_0 = \text{"0"}$$
$$f_1 = \text{"1"}$$
$$f_n = f_{n-2} + f_{n-1}$$

Some of the first Fibonacci strings are 0, 1, 01, 101, 01101, 10101101, ...

Input

- The first line contains a positive integer T — the number of test cases ($T \leq 10^5$);
- Each of the next T lines contains a non-negative integer n ($n \leq 10^9$).

Output

For each test case, print a single integer in one line, the beauty of f_n. Since the result may be large, print the result modulo 1000000007.

Sample Input	Sample Output
2 3 4	16 144

Problem H
Matrix Coloring
Time limit: 1 second

You are given a matrix of size $n \times m$ consisting only of the characters '.' and '#'. Each time a coloring action is performed, all consecutive '#' cells in a row (or a column) that have not yet been colored are painted. One coloring action stops when both directions of the row (or the column) reach the edge of the matrix, encounter a previously colored cell, or encounter a '.'.

Determine the minimum number of coloring actions needed to paint all the '#' cells in the matrix.

Input

- The first line contains two integers n and m, are the size of the matrix ($1 \leq n \leq 1000$; $1 \leq m \leq 10$).
- Each of the next n lines contains a string of m characters, each either '.' or '#'.

Output

Output a single integer representing the minimum number of coloring actions required.

Sample Input	Sample Output
4 4 #### #.#. #.## ####	5
5 4 #### .... ###. #### #.#.	5

Problem I
Building Hospitals
Time limit: 1 second

The country Ultrion Sovereign has n cities connected by $n-1$ roads, ensuring a path between every pair of cities. In other words, the cities and roads of Ultrion Sovereign form a tree.

The government plans to build hospitals to provide healthcare services for the citizens of these cities. To avoid overcrowding, the government has imposed the following requirements for hospital construction:

- Hospitals can be built in cities, and multiple hospitals can be built in a single city.
- Each city must belong to exactly one hospital's healthcare zone.
- Each hospital can have at most m cities in its healthcare zone.
- The distance from any city to the hospital whose healthcare zone contains the city cannot exceed k roads.

Find the minimum number of hospitals that need to be built to ensure all citizens can access a hospital.

Input

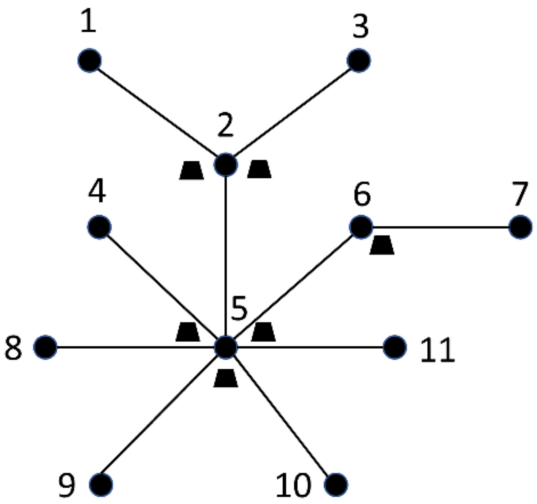
- The first line contains three integers n, m, k ($1 \le n \le 10^5$, $1 \le m \le n$, $1 \le k \le 20$).
- Each of the next $n-1$ lines contains two integers u, v ($1 \le u, v \le n$), describing a road connecting cities u and v.

Output

Output a single integer, the minimum number of hospitals that need to be built.

Sample Input	Sample Output
11 2 1 1 2 2 3 5 2 5 4 5 6 5 8 5 9 5 10 5 11 6 7	6

Explanation


1
3
2
4
6
7
5
8
11
9
10

Problem J
Beautiful Substring
Time limit: 1 second

You are given a string consisting of Latin letters. A substring is called "beautiful" if all distinct characters in the original string appear the same number of times in the substring.

A substring is a part of the string that consists of **consecutive** characters appearing in the same order as in the original string. The length of a substring can be from 1 to the entire length of the string.

Count the number of "beautiful" substrings in the string of length N. Substrings that are identical but appear at different positions are considered different.

Input

- The first line contains an integer N, the number of characters in the string S. ($2 \leq N \leq 10^5$).
- The next line contains the string S consisting of uppercase and lowercase Latin letters. Uppercase and lowercase letters are treated as distinct.

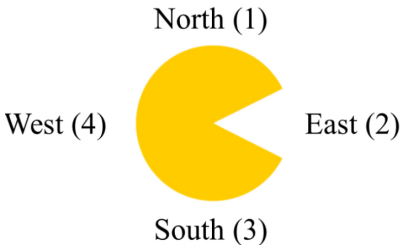
Output

Output a single integer representing the number of beautiful substrings in the string. The output can be large so it must be modulo by $10^9 + 7$

Sample Input	Sample Output
8 abccbabc	4
7 abcABCC	1

Problem K
Pacman Game
Time limit: 1 second

The screen of the Pacman game is represented as a matrix of N rows (numbered from 0 to $N-1$ from top to bottom) and N columns (numbered from 0 to $N-1$ from left to right). Pacman always faces one of four directions: North, East, South, or West — numbered from 1 to 4 by the game publisher, as shown in the figure below.



The game has three basic operations:

- **A** K: Pacman moves in the direction it's currently facing by K cells ($1 \leq K \leq 2N$). If it moves out of the matrix, it will reappear on the opposite side. For example, in a 5×5 matrix, if Pacman is at position $(1, 3)$ and is facing North (direction 1), executing the command **A 2** will move it to position $(4, 3)$.
- **R** K: Pacman rotates clockwise by 90° K times, with $1 \leq K \leq 4$. For example, if Pacman is facing North (direction 1) and the command **R 1** is executed, it will rotate to face East (direction 2).
- **Z** K: Opens a portal that teleports Pacman to position $([\frac{K}{N}], K\%N)$, keeping the same direction ($0 \leq K < N^2$). For example, in a 5×5 matrix, executing the command **Z 7** will teleport Pacman to position $(1, 2)$.

Given the initial positions of P Pacman and a sequence of commands performed by Duy, where each command affects all Pacman, determine the final positions of the Pacman after executing M commands.

Input

- The first line contains three integers N, P, M, which represent the size of the matrix, the number of Pacman, and the number of commands Duy performs.
- The next P lines, each containing three integers u, v, d, describe the position (u, v) and the direction d that each Pacman is facing.
- The next M lines contain commands in the form C K, where $C \in \{A, R, Z\}$ and K is an integer.

Output

Print P lines, each showing the final position of each corresponding Pacman after executing all M commands.

Constraints

- $1 \leq N \leq 1000$
- $1 \leq P \leq 10,000$
- $1 \leq M \leq 100,000$

Sample Input	Sample Output
5 3 4 1 1 2 2 3 1 3 1 4 A 3 R 3 A 1 A 3	2 4 4 4 2 3

Problem L
Lights
Time limit: 1 second

Binh is playing a game involving n lights and n switches. Initially, the state of these n lights is given, numbered from 1 to n.

Each light has two states: either on or off. We use 1 to indicate that the light is on, and 0 to indicate the light is off. The goal of the game is to turn off all the lights.

However, when Binh operates switch i, the state of all lights that are divisors of i (including both 1 and i) is toggled. That means if a light is on, it will be turned off, and if a light is off, it will be turned on.

Binh realizes that this game is very difficult, so he comes up with a strategy: each time, he randomly selects one switch (with probability $\frac{1}{n}$) to operate, and continues to do so until all the lights are off.

Binh wants to know the expected number of operations required for this strategy. Additionally, Binh realizes that there may be a better approach: if, at the current time, all the lights can be turned off in no more than k operations, instead of continuing to select a switch randomly, he directly chooses the optimal method that requires the fewest operations (this approach ensures that no more than k steps are needed to turn off all the lights).

Binh wants to calculate the expected number of operations for this strategy (where he first randomly selects switches and then switches to the optimal method if the number of required steps is no greater than k).

The expectation can be very large, but Binh realizes that if multiplied by $n!$, the result will always be an integer. Therefore, he needs to find the result of the product of the expectation multiplied by $n!$, modulo 100003.

Input

- The first line contains two integers n and k.
- The next line contains n numbers, each either 0 or 1. The i-th number represents the initial state of the i-th light.

Output

Output an integer, the expected number of operations multiplied by $n!$, and then modulo 100003.

Sample Input	Sample Output
4 0 0 0 1 1	512
5 0 1 0 1 1 1	5120

Constraints

- $1 \leq n \leq 10^5$, $0 \leq k \leq n$

Problem M
Suffixes
Time limit: 1 second

Given a binary string S of length n consisting of the characters 0 and 1. An defines the function maxsufpre(u, v) as follows: Consider all suffixes of the string S whose starting positions are in the range $[u, v]$. The result of the function maxsufpre(u, v) is the longest common prefix's length between two suffixes with the longest common prefix.

An wants to know the value of D represented by the expression below, but due to the complexity of the formula, An needs your help to calculate it.

$$D = \sum_{i=L}^{R-1} \texttt{maxsufpre}(i, R) \tag{1}$$

However, An does not have just one pair $[L, R]$ but many such pairs, and An wants to calculate the value of D for each of them. Please help An.

Input

- The first line contains two integers n and Q, corresponding to the length of the binary string S and the number of pairs $[L, R]$.
- The second line contains the string S, consisting of n characters, either 0 or 1.
- The next Q lines, each containing two integers L and R.

Output

For each of the Q pairs, output the value D corresponding to the segment $[L, R]$.

Sample Input	Sample Output
6 3 010110 2 5 1 6 1 2	4 6 0

Constraints

- $1 \leq n, Q \leq 10^5$, $1 \leq L < R \leq n$