Problem List

Problem A: Two arrays 1
Problem B: Metro System Development Simulation 2
Problem C: Weighted Substring Queries 4
Problem D: The Alchemist's Diminishing Gold 6
Problem E: Delete sequence 7
Problem F: Power Absorption 8
Problem G: Electric Circuit 10
Problem H: Removing noisy data 12
Problem I: Draw a Polygon 13
Problem J: The most famous game 14
Problem K: Emma's Array Balancing Challenge 15
Problem L: Beautiful numbers 17

Problem A

Two arrays

Time limit: 1 second
Memory limit: 256 megabytes

You are given two integer arrays a and b, each containing n elements: $a_1, a_2, \ldots, a_n$ and $b_1, b_2, \ldots, b_n$. You are allowed to perform the following operation any number of times:

- Choose an index i $(1 \le i \le n)$, and **increase both** a_i and b_i by one.

Given two integers C and D, determine the **minimum number of operations** required to make the two arrays satisfy:

$$\max(a) - \min(a) \le C \quad \text{and} \quad \max(b) - \min(b) \le D.$$

Here, $\max(a)$ and $\min(a)$ denote the maximum and minimum elements of array a, respectively; similarly for $\max(b)$ and $\min(b)$.

Input

The first line contains an integer T, which is the number of test cases. For each test case:

- The first line contains three integers n, C, D $(1 \le n \le 10^5,\ 0 < C, D \le 10^9)$.
- The second line contains n integers $a_1, a_2, \ldots, a_n$ $(-10^9 \le a_i \le 10^9)$.
- The third line contains n integers $b_1, b_2, \ldots, b_n$ $(-10^9 \le b_i \le 10^9)$.

The sum of n over all test cases is at most 10^5.

Output

For each test case, print a single integer on a single line - the minimum number of operations required to make both arrays satisfy the given constraints. In case we cannot find a way to satisfy the condition, print -1.

Sample Input	Sample Output
2 4 2 3 -1 -2 -3 -4 -1 -2 -3 -4 5 2 1 -1 0 1 2 3 2 2 2 2 2	1 -1

Problem B
Metro System Development Simulation
Time limit: 2 seconds
Memory limit: 512 megabytes

The bustling metropolitan city of TTT has been experiencing unprecedented growth in its tech industry, leading to severe traffic congestion and transportation challenges. As the Chief Metro Planning Engineer for the TTT Metropolitan Transit Authority, you've been assigned to design and simulate the city's first comprehensive metro system.

Currently, TTT has n metro stations planned, connected by m uni-directional metro lines, with stations numbered from 1 to n. The transit authority has approved a multi-phase metro expansion project to create an efficient underground transportation network. Your simulation system needs to handle two critical types of metro development operations that will be implemented across different construction phases.

Your task is to simulate q metro development operations that represent real-world construction phases and route planning decisions. Each operation can have two types as described below:

Operation Type 1 - Station Expansion: $1\ x\ d$
A new metro station $n+1$ is constructed as part of the current expansion phase, and it is connected to existing station x with a new metro line to ensure network connectivity.

- If $d = 0$, the direction of the new metro line is from station x to the new station $n + 1$ (allowing metro travel from the existing station to the new station)
- If $d = 1$, the direction of the new metro line is from the new station $n + 1$ to station x (allowing metro travel from the new station to the existing station)

The value of n must be incremented by 1 after this operation as the metro system now has $n + 1$ stations to manage.

Operation Type 2 - Route Connectivity Check: $2\ x\ y$
The operations control center needs to verify if passengers can travel from station x to station y using the current metro network. Print `Yes` if it's possible to travel from station x to station y through the existing metro lines, print `No` otherwise.

As the metro planning engineer, you need to process these operations in real-time to help transit authority officials make informed decisions about TTT's metro expansion phases and route optimization.

Input

The first line of input contains two space-separated integers n (the initial number of metro stations) and m (the number of existing metro lines).

Next, m lines of input contain two space-separated integers u and v denoting that there is a uni-directional metro line from station u to station v.

Next line contains an integer q denoting the number of metro development operations to simulate.

Next q lines of input contain q operations, one per line, where each operation is one of the two possible types described above.

Constraints

- $1 \le n, m \le 5 \times 10^4$
- $1 \le q \le 10^5$
- $1 \le u, v \le n$
- $d \in \{0, 1\}$
- The values of x, y always correspond to an existing metro station in TTT at the time of the query.

Output

For each Operation Type 2 (route connectivity check), print `Yes` if it's possible to travel from station x to station y, otherwise print `No`.

Sample Input	Sample Output
3 3 1 2 2 3 3 1 6 2 1 3 1 2 0 2 1 4 2 4 1 1 3 1 2 5 1	Yes Yes No Yes

Explanation

Initial State: Metro stations $1, 2, 3$ with lines $1 \to 2$, $2 \to 3$, $3 \to 1$ forming a circular route.

Operation 1: `2 1 3` - Check if passengers can travel from station 1 to station 3.
Route exists: $1 \to 2 \to 3$. Output: `Yes`

Operation 2: `1 2 0` - Build new metro station 4, connect station 2 to station 4 (metro line $2 \to 4$).
Now we have stations $1, 2, 3, 4$ with lines: $1 \to 2$, $2 \to 3$, $3 \to 1$, $2 \to 4$

Operation 3: `2 1 4` - Check if passengers can travel from station 1 to station 4.
Route exists: $1 \to 2 \to 4$. Output: `Yes`

Operation 4: `2 4 1` - Check if passengers can travel from station 4 to station 1.
No route exists from station 4 to station 1. Station 4 is a terminal station with no outgoing lines.
Output: `No`

Operation 5: `1 3 1` - Build new metro station 5, connect station 5 to station 3 (metro line $5 \to 3$).
Now we have stations $1, 2, 3, 4, 5$ with lines: $1 \to 2$, $2 \to 3$, $3 \to 1$, $2 \to 4$, $5 \to 3$

Operation 6: `2 5 1` - Check if passengers can travel from station 5 to station 1.
Route exists: $5 \to 3 \to 1$. Output: `Yes`

Problem C
Weighted Substring Queries
Time limit: 1 second
Memory limit: 256 megabytes

You are given N distinct strings. The i-th string is s_i and has weight c_i. You are also given a text string T.

For each query $[L, R]$ (1-indexed, inclusive), consider the substring

$$\mathrm{st} = T[L]\,T[L+1]\ldots T[R].$$

Define the *value* of the substring st as

$$\mathrm{val}(L, R) \;=\; \sum_{i=1}^{N} c_i \cdot k_i,$$

where k_i is the number of (possibly overlapping) occurrences of s_i as a **contiguous** substring of st. For every query, output $\mathrm{val}(L, R)$.

Input

The first line contains two integers N and Q ($1 \le N, Q \le 10^5$).

Each of the next N lines contains a string s_i and an integer c_i ($1 \le c_i \le 10^9$). All s_i are pairwise distinct.

The next line contains the text string T.

Each of the last Q lines contains two integers L and R denoting a query.

Additional guarantees:

- The total length of all input strings (all s_i plus T) does not exceed 10^5.
- The total number of occurrences of all s_i inside T is at most 10^5.

Output

Print Q lines. The j-th line contains a single integer — the answer to the j-th query.

Sample Input	Sample Output
6 3 she 3 he 2 her 5 sh 1 er 4 us 2 ushersheher 1 5 2 4 6 8	17 6 6

Note

For the first query $[1, 5]$, the substring is `usher`. Occurrences inside `usher`:

- `she` occurs 1 time: contributes 3
- `he` occurs 1 time: contributes 2
- `her` occurs 1 time: contributes 5
- `sh` occurs 1 time: contributes 1
- `er` occurs 1 time: contributes 4
- `us` occurs 1 time: contributes 2

Total: $3 + 2 + 5 + 1 + 4 + 2 = 17$.

Problem D
The Alchemist's Diminishing Gold
Time limit: 2 seconds
Memory limit: 512 megabytes

An alchemist has created a magical piece of gold with an initial **Essence Value** of n. He plans to perform a refining process exactly k times to purify it.

Each time the process is performed on a piece of gold with Essence Value v, the gold fractures into several fragments. The Essence Values of these fragments are **exactly the divisors of** v. Due to the chaotic nature of the magical reaction, the alchemist must **randomly pick one of the resulting fragments** to continue the process. Every fragment (i.e., every divisor) has an **equal chance** of being selected.

The alchemist wishes to forecast the outcome of his experiment. What is the **expected Essence Value** of the final piece of gold after k refining steps?

The answer can be represented as a fraction $\frac{P}{Q}$, where P and Q are coprime and $Q \not\equiv 0 \pmod{10^9+7}$. Output the result as $P \cdot Q^{-1} \bmod 10^9 + 7$.

Input

A single line containing two integers n and k $(1 \le n \le 10^{15},\ 1 \le k \le 10^4)$.

Output

Print a single integer - the expected Essence Value after k steps of refining, modulo $10^9 + 7$.

Sample Input	Sample Output
7 1	4
7 3	750000007
77 7	642333990

Problem E
Delete sequence
Time limit: 1 second
Memory limit: 512 megabytes

You are given a sequence of n positive integers $a_1, a_2, \dots, a_n$ arranged from left to right. You may perform the following operations:

- **Operation 1:** If the first two numbers of the current sequence are **coprime** or one of them is a **multiple** of the other, delete both numbers.
- **Operation 2:** If Operation 1 cannot be applied to the first two numbers, choose an integer $x > 1$, increase the first number by x (i.e. $a_1 \leftarrow a_1 + x$), and append x to the **end** of the sequence.

Your goal is to make the sequence empty (removing both the original numbers and any appended numbers) using the **minimum** number of operations.

Input

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

For each test case:

- The first line contains an integer n ($1 \le n$).
- The second line contains n integers $a_1, a_2, \dots, a_n$ ($2 \le a_i \le 10^6$).

Global constraints:

- $1 \le t \le 10^6$,
- $\sum_{\text{all tests}} n \le 10^6$.

Output

For each test case, print a single integer — the minimum number of operations needed to delete the entire sequence.

Sample Input	Sample Output
2 2 28 30 1 30	4 2

Explanation

In the first test case, one optimal sequence of operations is:

$$(28, 30) \to (31, 30, 3) \to (3) \to (5, 2) \to ()$$

which uses 4 operations.

Problem F
Power Absorption
Time limit: 1 second
Memory limit: 256 megabytes

You are playing a very popular Marvel video game called *Dr. Strange*. In this game, you take on the role of a superhero who is saving the world by absorbing the powers of monsters.

You are given a list of n monsters. The i-th monster appears at time L_i and remains present through time R_i, inclusive. It has a power level P_i. Multiple monsters may be present at the same time.

To fight against the monsters, you will perform m power absorption moves in sequence. Before the first move, you start with an initial absorbed power value of $\text{Power}_0 = 1$ (absorb from yourself).

For each move $j = 1, 2, \ldots, m$, your available energy E_j at that moment is calculated as:

$$E_j = 1 + (D_j \cdot \text{Power}_{j-1} + A_j) \bmod F_j$$

Where D_j is the durability coefficient, A_j is the agility coefficient, F_j is the fatigue level at move j, $\texttt{Power}_{j-1}$ is the total power of the monsters absorbed in the **previous** move $j-1$.

Because you are Dr. Strange, you can travel through time. At time t_j of the j-th move, with this energy E_j, you absorb the powers of the E_j weakest monsters (those with the smallest power values) currently present. If there are fewer than E_j monsters at that time, you absorb all of them. Absorbing their powers only affects your energy - it **does not weaken or eliminate** any monsters.

Let Power_j denote the total power absorbed during the j-th move. Your task is to determine $\text{Power}_1, \text{Power}_2, \ldots, \text{Power}_m$.

Input

- The first line contains an integer n, m - the number of monsters and the number of absorption moves. $(1 \le n, m \le 10^5)$
- The next n lines each contain three integers L_i, R_i, and P_i - the appearance time, disappearance time, and power of the i-th monster. $(1 \le L_i \le R_i \le 10^5, 1 \le P_i \le 10^7)$
- The next m lines each contain four integers t_j, D_j, A_j, and F_j - the time of the j-th move, and the coefficients used to compute the energy for this move. Note that all t_j form a permutation of numbers from 1 to m. $(1 \le t_j \le m, 0 \le D_j, A_j \le 10^5, 1 \le F_j \le 10^5)$

Output

Print m lines. Each line should contain a single integer - the total power absorbed in the j-th move.

Sample Input	Sample Output
3 3 1 2 10 2 3 20 1 3 5 1 2 2 2 3 3 1 3 2 1 1 5	5 25 15

Explanation

- In the first move at time $t_1 = 1$, two monsters are present. You have $\text{Power}_0 = 1$, so your energy will be: $E_1 = 1 + (2 \times 1 + 2) \bmod 2 = 1$. The monster having power 5 is absorbed.
- Second move: $E_2 = 1 + (3 \times 5 + 1) \bmod 3 = 2$. So the total power absorbed is 25.
- Third move: $E_3 = 1 + (1 \times 25 + 1) \bmod 5 = 2$. Two monsters with powers 5 and 10 are absorbed, so the total power is 15.

Problem G

Electric Circuit

Time limit: 2 seconds
Memory limit: 512 megabytes

You are given an electric circuit consisting of a sequence of infinity switches, aligned in a row and labeled from left to right starting from 0.

Each switch can be in one of two states: **ON (closed)** or **OFF (open)**. All switches are initially OFF.

You are told the following rule of interaction between adjacent switches:

- If a switch transitions **from OFF to ON**, it has **no effect** on other switches.
- If a switch transitions **from ON to OFF**, it will **cause the switch immediately to its right** to toggle its current state (i.e., ON $\rightarrow$ OFF or OFF $\rightarrow$ ON).

You plan to perform m operations. The i-th operation toggles the state of switch a_i ($0 \le a_i \le n$):

- If it is ON, it becomes OFF (and may cause a chain reaction to the right, per the rule above).
- If it is OFF, it becomes ON.

Let the **cost** of an operation be defined as the **number of switches whose state was changed** (including both the directly toggled switch and any affected ones to the right via propagation).

However, each operation is **executed with a certain probability**:

- The i-th operation is executed with probability $p_i = \frac{u_i}{v_i}$.
- Otherwise, it is skipped with probability $1 - p_i$.

You are required to compute the **expected total cost** of all m operations.

Let the expected value be $E = \frac{P}{Q}$, where P and Q are coprime integers. Output the value:

$$(P \cdot Q^{-1}) \bmod 998244353$$

where Q^{-1} denotes the modular inverse of Q modulo 998244353.

Input

The first line contains two integers n, m ($1 \le n, m \le 2 \cdot 10^5$) - the upper bound of a_i and the number of operations.

The next m lines each contain three integers a_i, u_i, v_i ($0 \le a_i \le n,\ 0 \le u_i < 998244353,\ 1 \le v_i < 998244353, u_i \le v_i$) - the position of the switch, and the probability of executing the i-th operation as a rational fraction $p_i = \frac{u_i}{v_i}$.

Note

The number of switches is infinite, n is just the upper bound of a_i in the input.

Output

Print a single integer - the value of $P \cdot Q^{-1} \bmod 998244353$, where $\frac{P}{Q}$ is the expected total cost of all operations.

Sample Input	Sample Output
3 3 0 1 2 1 1 2 2 1 2	499122178
100 5 0 100 333 0 333 666 0 666 1234 1 1234 1235 2 1235 1236	610769569

Problem H

Removing noisy data

Time limit: 1 second
Memory limit: 512 megabytes

During a break time in the "Noisy Data Processing" class, teacher gives each student a sequence of integers of length $n \geq 2$. The teacher asks them to compute the product of all numbers in the sequence.

Being top students, Luna and Thana could easily do this with a single line of code. However, the teacher wants them to apply the knowledge from the lecture on noise reduction. The challenge is that each student must remove **exactly one number**, can be considered as noisy data, from the original sequence so that the product of the remaining numbers is **as large as possible**.

If there are multiple choices of which number to remove that lead to the same maximal product, then:

- Luna will remove the **largest** such number.
- Thana will remove the **smallest** such number.

Your task is to help Luna and Thana determine which numbers they should remove.

Input

- The first line contains an integer n ($2 \leq n \leq 10^5$).
- The second line contains n integers, not necessarily distinct, each having an absolute value not exceeding 10^5.

Output

Print two integers — the numbers that Luna and Thana should remove, respectively.

Sample Input	Sample Output
4 0 1 2 3	0 0

Sample Input	Sample Output
4 0 0 -1 2	2 -1

Explanation

In the first example, removing any number other than 0 results in a zero, so both of students should remove 0. In the second example, since there are two zeros, removing either does not change the product (it remains 0); thus, Luna removes the largest possible number (which is 2), while Thana removes the smallest possible number (which is -1).

Problem I

Draw a Polygon

Time limit: 1 second
Memory limit: 256 megabytes

Your task is to draw a simple polygon with exactly n vertices. A *simple polygon* is a non-self-intersecting polygon without holes, and no three **consecutive** vertices are collinear. All vertex coordinates must be integers within the range $[-10^9, 10^9]$.

In addition, the number of *diagonals lying entirely inside* the polygon must be exactly k. (A diagonal of a polygon is a segment connecting two non-adjacent vertices.)

Input

The only line contains two integers n and k ($4 \le n \le 100$, $0 \le k \le \frac{n(n-3)}{2}$).

Output

If it is impossible to draw such a polygon, print a single word `No`. Otherwise, print `Yes` on the first line, followed by n lines containing integer pairs (x_i, y_i) ($-10^9 \le x_i, y_i \le 10^9$) — the vertices of the polygon listed either clockwise or counterclockwise.

The polygon must not intersect or touch itself, no two vertices may coincide, and no three consecutive vertices may lie on a common line.

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

Problem J
The most famous game
Time limit: 1 second
Memory limit: 256 megabytes

Two players play a single round of *Rock–Paper–Scissors* game. Each player chooses one of the following moves:

- `Rock`
- `Paper`
- `Scissors`

The rules are standard:

- Rock beats Scissors.
- Scissors beats Paper.
- Paper beats Rock.
- Same moves result in a draw.

Task. Given the two moves, determine the outcome of the game.

Input

The input contains two strings a and b (`"Rock"`, `"Paper"`, or `"Scissors"`), representing the moves of Player 1 and Player 2 respectively.

Output

Print one of the following:

- `Player 1` — if Player 1 wins.
- `Player 2` — if Player 2 wins.
- `Draw` — if both moves are the same.

Sample Input	Sample Output
`Rock Scissors`	`Player 1`
`Paper Paper`	`Draw`
`Scissors Rock`	`Player 2`

Problem K
Emma's Array Balancing Challenge
Time limit: 1 second
Memory limit: 512 megabytes

Emma is a data analyst working for a tech startup that specializes in creating balanced datasets for machine learning applications. Her company has developed a revolutionary algorithm that requires perfectly balanced numerical arrays to function optimally.

Emma has an array $[a_0, a_1, a_2, \ldots, a_{n-1}]$ of size n where n is an even number representing critical sensor data from IoT devices. An array is balanced if the sum of the left half of the array elements is equal to the sum of right half - what her team calls *"balanced arrays."*

For example:

- **Balanced Array**: $[1, 2, 5, 2, 4, 2]$ where left half sum $= 1 + 2 + 5 = 8$ and right half sum $= 2 + 4 + 2 = 8$
- **Unbalanced Array**: $[1, 2, 1, 2, 1, 3]$ where left half sum $= 1 + 2 + 1 = 4$ and right half sum $= 2 + 1 + 3 = 6$

Unfortunately, the raw sensor data rarely comes in balanced form. Emma has discovered that she can fix unbalanced arrays by adding non-negative values to any elements in the array. However, her company wants to minimize data modification to preserve the integrity of the original sensor readings.

To balance an array, Emma can increase exactly one element a_i by a non-negative integer x. Your task is to help Emma find the smallest value of x that makes the array balanced, ensuring the AI algorithm can process the data while maintaining maximum data authenticity.

Input

The first line contains an even integer n.

The second line contains the n integer elements of the array $a_0, a_1, a_2, \ldots, a_{n-1}$.

Constraints

- $1 \le n \le 100000$
- $0 \le a_i \le 1000$
- n is an even number.

Output

Print the minimum value of x on a single line.

Sample Input	Sample Output
6 1 2 1 2 1 3	2

Explanation

To balance the array, Emma needs to add 2 to a_2: For example:

- Adding 2 to a_2: $[1, 2, 3, 2, 1, 3]$ gives left sum $= 1 + 2 + 3 = 6$ and right sum $= 2 + 1 + 3 = 6$

The array is now balanced with minimum addition of $x = 2$.

Problem L

Beautiful numbers

Time limit: 1 second
Memory limit: 512 megabytes

For each positive integer x, let $S(x)$ denote the sum of digits of x in decimal representation. For example, $S(2025) = 2 + 0 + 2 + 5 = 9$.

A positive integer n is called "beautiful" if it satisfies both of the following conditions:

- There exist positive integers a, b such that $n = a + b$ and $S(a) = S(b)$.
- There exist positive integers c, d, e such that $n = c + d + e$ and $S(c) = S(d) = S(e)$.

You are asked to answer q queries. In each query, you are given two integers L and R ($1 \le L \le R \le 10^{18}$). For each query, determine how many beautiful numbers are contained in the interval $[L, R]$.

Input

The first line contains a single integer q ($1 \le q \le 10^5$) — the number of queries. Each of the next q lines contains two integers L and R ($1 \le L \le R \le 10^{18}$).

Output

For each query, output a single integer — the count of beautiful numbers in the interval $[L, R]$.

Sample Input	Sample Output
2 1 10 2024 2026	1 1

Explanation

In the first test case, the only beautiful number in $[1; 10]$ is 6, since $6 = 3 + 3 = 2 + 2 + 2$, because $S(3) = S(3) = 3$ and $S(2) = S(2) = S(2) = 2$. In the second test case, the only beautiful number in $[2024; 2026]$ is 2025, because $2025 = 2016 + 9 = 2013 + 6 + 6$ and $S(2016) = S(9) = 9$, while $S(2013) = S(6) = S(6) = 6$. No other numbers in these intervals satisfy both given conditions.