Think.
Create.
Solve.

2025 Asia-Manila Regional Programming Contest
9–10 December 2025
Ateneo de Manila University

Problem Set

Ateneo de Manila University
ARETE
icpc.foundation
HUAWEI
JETBRAINS
Bookmark
Ateneo Laboratory for the Learning Sciences
Smart
VÉRTERE
stratpoint

Problem A
Alphabet Chocolate

Alice and Bob decided to stop sending each other cryptic messages for once, and are now sharing a bar of chocolate. The bar is split into a row of n pieces, and as part of a cute gimmick from the chocolate-maker, each piece of the chocolate bar has a single letter written on it!

Alice and Bob agree on the following protocol: before they reach for a piece, they must first simultaneously declare if they want the *leftmost piece* or the *rightmost piece*. That way, they can confirm there won't be any conflict (but if they *do* end up wanting the same piece... oops, their protocol didn't plan for that).

Thankfully, Alice and Bob are simple predictable creatures and will always choose to get from the same side. The following procedure repeats until the chocolate bar is all eaten up (or until they fight):

- Alice says "I want to eat the leftmost remaining piece!" and
  Bob says "I want to eat the rightmost remaining piece!"
- If they want different pieces, then they each break off the piece that they want and eat it.
- If they want the same piece, they will fight over it. Oh no!

Please determine whether or not Alice and Bob will fight. And if they will fight, determine what letter is written on the piece that causes them to fight.

Input Format

Input consists of a single line, containing a single string s, which consists of n uppercase English letters—from left to right, these are the letters written on the pieces on the chocolate bar.

Output Format

If they will not fight, output the two characters `:)`

If they will fight, output a single letter, the one written on the piece that causes them to fight.

Constraints

Constraints
$2 \leq n \leq 2 \times 10^5$ s consists only of n uppercase English letters.

Sample I/O

Input	Output
`ICPCMANILA`	`:)`

Input	Output
`RACECAR`	`E`

Problem B
DJ Nicholas

DJ Nicholas is a popular DJ for Christmas and New Year events. DJ Nicholas has a battle-tested process for creating his playlists, but he needs help preparing them. His battle-tested process is as follows.

First, he has a stash of k short Christmas song snippets. Since k is at most 26, he assigns the first k letters of the alphabet to the songs in his stash (in the order they would first be played).

He creates an infinitely repeating master track t by repeating his song stash of k songs over and over again. So that the track doesn't get too stale, every time DJ Nicholas would repeat the song stash in the master track, he first *shifts* the order by taking what is currently the first song and then moving it to the end of the stash.

For example, if $k = 4$, then his song stash can be represented by the string `ABCD`, and the master track t can be represented as the infinite string "`ABCD BCDA CDAB DABC ABCD ...`" (the spaces are shown only for readability; they are not actually in t).

The playlist he prepares can be represented as a string of length n, initially filled with *blanks*. Next, he does q operations on the playlist, where each operation is in one of two forms:

- "`+ i a b`". This takes the segment of songs a through b (inclusive) in the playlist, and **replaces it** with the substring of t that starts at position i and has length $b - a + 1$.
- "`? a b`". This outputs k integers; the jth such integer is the number of appearances—from among songs a through b (inclusive) in the playlist—of the song represented by the jth letter of the alphabet.
  - DJ Nicholas needs this to keep track of royalties he needs to pay.

Here, we assume both the master track and the playlist are 1-indexed.

You can refer to the sample I/O to see DJ Nicholas' battle-tested process in action!

Help DJ Nicholas answer the above queries and ensure his audience leaves the event merry!

Input Format

The first line of input contains three space-separated integers k, n, and q.

Then, q lines follow, describing the operations. Each line is of either the form "`+ i a b`" or "`? a b`", corresponding to the operations described above.

Output Format

For every operation of the form "`? a b`", output a line of k space-separated integers, corresponding to the answer to that query.

Constraints

Constraints
$1 \le k \le 26$
$1 \le n \le 10^9$
$1 \le q \le 10^5$
$1 \le i \le 10^9$ in each operation
$1 \le a \le b \le n$ in each operation

Sample I/O

Input	Output

`4 10 4`
`+ 7 2 6`
`+ 2 8 10`
`+ 9 1 4`
`? 4 9`
	`1 2 1 1`

Explanation

Here, we have $k = 4$ and $n = 10$. So, $t =$ `ABCDBCDACDABDABC`...

The playlist is initially empty, so we represent it by this string (where blanks are represented by asterisks):

`**********`

If we are to perform the operation "`+ 7 2 6`", we would note that

- We need a substring of $6 - 2 + 1 = 5$ letters.
- Here we highlight in t the substring that starts at position 7 and is 5 letters long: `ABCDBCDACDABDABC`...
- Replace songs 2 through 6 in the playlist with this, so our playlist now looks like:

`*DACDA****`

You can then verify that after "`+ 2 8 10`", we have the string below:

`*DACDA*BCD`

Then, with the operation "`+ 9 1 4`", we *overwrite* some previous songs, ending up with:

`CDABDA*BCD`

Afterwards, we query "`? 4 9`", so we look at songs 4 through 9 in the current playlist:

`CDABDA*BCD`

The four integers we output should be the number of times that `A`, `B`, `C`, and `D` (respectively) appear in this range at this moment. So, the output should be "`1 2 1 1`".

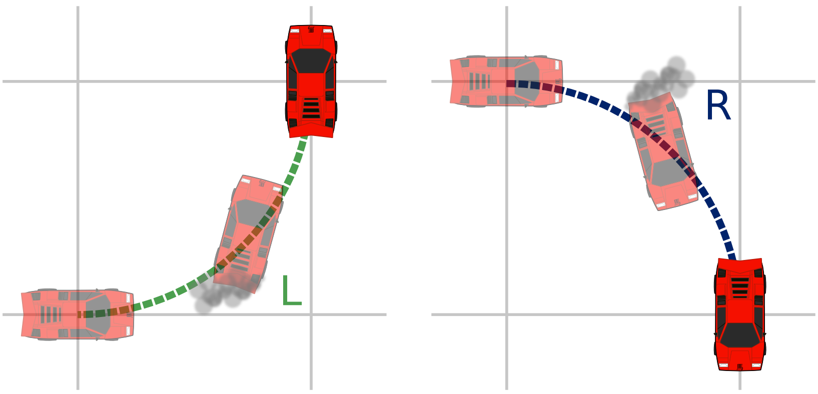
Problem C
The Drift King

There exists a cosmic being as old as the universe itself, one that they used to call *the Drift King* (back in college). The Drift King is simultaneously an entity of chaos and order. He drives his cosmic supercar[1] through space in a weird twisty-loopy-curvy manner that initially seems erratic... but in reality, the pattern is not so hard to understand!

Model the universe using the 2D Cartesian plane, and let the location of the Drift King's car be represented by a single point that is *oriented* so that it has a "front" facing a certain direction. The Drift King, naturally, only moves by drifting. His car can move in one of two ways, each represented by a certain symbol:

- `L` means: move counter-clockwise along a quarter-circular path that is centered around the point 1 unit to its left.
- `R` means: move clockwise along a quarter-circular path that is centered around the point 1 unit to its right.

Here is a picture of what those look like, if the Drift King's car is currently facing east.



These terms "left" and "right" are *relative* to where the car is currently facing. These quarter-circle turns will change the direction the car is facing at every step. The speed of the car is such that it completes one of these quarter-circle arcs in 1 unit of time.

The car's path is defined by a pattern formed by chaining these movements together, and then repeating that pattern infinitely (in both directions). We also know the location and orientation of the car at time $t = 0$, and thus its path is uniquely defined.

Formally, let s be a 0-indexed string $s = s_0 s_1 s_2 \ldots s_{n-1}$, where each character is `L` or `R`. This string s describes the pattern that extends infinitely in both directions. It performs these indicated turns in order, then loops back to the start when it reaches the end of the string (or vice versa).

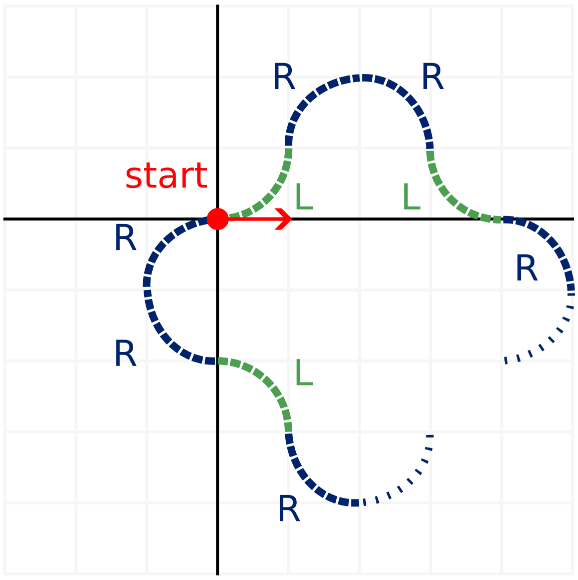
Precisely, the car's journey is defined to be the unique continuous path such that:

- at every integer time t (including the *negative* values), the car initiates a turn in the direction indicated by $s_{(t \bmod n)}$;
- at time $t = 0$, the car is at $(0, 0)$ and facing east.

We emphasize that even *in between* these integer timestamps, the car moves *continuously* through space as it performs these turns.

[1] a Lamborghini Countach LP1211-M

Here is what that path might look like, following the pattern `LRR`. Note how the car has already been moving in this pattern even before time $t = 0$.



Suppose that the Earth is located at the point (h, k). What is the closest (in terms of Euclidean distance) that this *menace* the Drift King will ever get (or has ever gotten) to our planet? Also, there will be T independent test cases per file.

Input Format

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

The first line of each test case contains the two space-separated integers h and k.

The second line of each test case contains the string s.

Output Format

Output a single decimal value, the closest the Drift King ever gets (or has gotten) to Earth.

Your answer will be accepted if it has an absolute or relative error of at most 10^{-8} from the judge's answer. In symbols, let ans_{you} be your answer, and let ans_{judge} be the judge's answer. Your answer will be accepted if

$$\frac{|ans_{\text{you}} - ans_{\text{judge}}|}{\max(1, ans_{\text{judge}})} \leq 10^{-8}$$

Constraints

Constraints
$1 \leq T$
$-10^9 \leq h, k \leq 10^9$
$1 \leq \lvert s \rvert$
$\sum \lvert s \rvert \leq 10^5$ across all test cases
Each character of s is `L` or `R`

Sample I/O

Input	Output
2 3 2 LRR 2 -1 LRR	0.41421356237309504876 1.82842712474619009753

Explanation

Both sample test cases use the pattern `LRR`, which is the example illustrated in the problem statement.

Problem D
Drinking Culture

Filipinos love alcohol. Those planning an *inuman*[2] have a plethora of options to choose from while in the Philippines: from the traditional palm liquor *lambanog*, to the ubiquitous *San Miguel Pale Pilsen* beer, to the classic *Tanduay Rum*.

Did you know that the logo on the bottle of *Ginebra San Miguel* (a popular local gin) was a commission done in 1917 by painter *Fernando Amorsolo*, who would later go on to be recognized as the Philippines' first National Artist? Drinking is a part of Filipino culture!

Bob fancies himself a mixologist. He has a collection of n bottles of alcoholic beverages, labeled 1 to n. He knows that bottle i contains exactly v_i units of liquid (measured by volume), exactly a_i units of which are alcohol. Note that $0 \le a_i \le v_i$ always.

The *alcohol content* of a liquid is equal to what proportion of that liquid (by volume) is alcohol. The alcohol content of the liquid in bottle i is exactly a_i/v_i. The contents of each bottle are completely homogeneous, i.e. any amount of liquid taken from bottle i will maintain that same alcohol content of a_i/v_i.

To mix a drink, Bob selects any subset of the bottles in his collection and takes *any* amount of liquid from each of those bottles *(they do not have to be integer values)*. He combines those samples into one glass, and mixes until homogeneous.

Bob's friend, a statistician, who's had a long and grueling day at work, wants him to mix her a drink that will render her unconscious. She issues Bob the following challenge:

- She generates a *real number* s uniformly randomly from the interval $[0, V]$, where $V = \sum_{i=1}^{n} v_i$ is the total amount of liquid across all bottles in Bob's collection.
- She also generates a *real number* f uniformly randomly from the interval $[0, 1]$.
- Bob's task is to use his collection to mix her a drink, such that
  - the total volume of liquid in the drink is exactly s, and
  - the alcohol content of the drink is exactly f.

What is the probability that Bob can successfully complete this task?

Input Format

The first line of input contains a single integer n.

The second line of input contains the n space-separated integers $v_1, v_2, v_3, \ldots, v_n$.

The third line of input contains the n space-separated integers $a_1, a_2, a_3, \ldots, a_n$.

Output Format

Output a single decimal value between 0.0 and 1.0, the probability Bob can complete this task.

Your answer will be accepted if it has an absolute or relative error of at most 10^{-8} from the judge's answer. In symbols, let ans_{you} be your answer, and let ans_{judge} be the judge's answer. Your answer will be accepted if

$$|ans_{\text{you}} - ans_{\text{judge}}| \le 10^{-8}$$

[2] TL note: a drinking session with friends

Constraints

Constraints
$2 \le n \le 2 \times 10^5$ $1 \le v_i \le 10^9$ and $0 \le a_i \le v_i$ for each i

Sample I/O

Input	Output
3 350 750 330 140 131 16	0.19356182654786474591

Explanation

Let's consider two concrete examples. Here, $V = 350 + 750 + 330 = 1430$.

Suppose $s = 500$ and $f = 3/13$. Then, the answer is yes! Bob could (for example) mix $97750/377$ units of the drink in bottle 1, and $90750/377$ units of the drink in bottle 3. The total amount of liquid in the mix would be

$$97750/377 + 90750/377 = 188500/377 = 500,$$

and the alcohol content would be

$$\frac{(97750/377)\frac{140}{350} + (90750/377)\frac{16}{330}}{500} = \frac{3}{13}.$$

However, suppose $s = 814$ and $f = 0.1234567$. Then unfortunately, the answer is no. It can be shown that such a drink cannot be mixed using Bob's collection of alcohol.

When considering the distribution of all possible pairs of values for s and f, the probability that the drink can be mixed is $\approx 0.19356182654786474591$.

Problem E

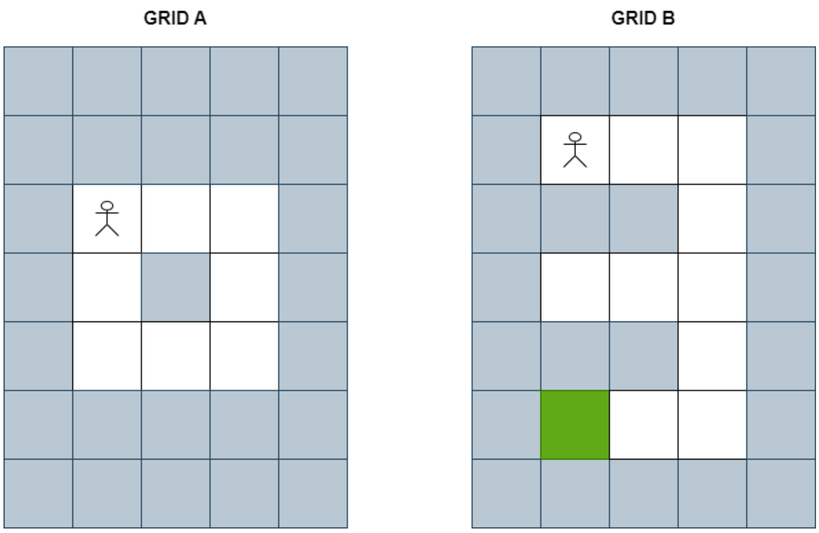
Long Distance Examination

Hero A (obviously an alias) is one of the applicants to the Super Resonant Psychic Hub (SRPH), the pre-eminent association of mentally-powered superheroes. After having passed the first round of exams, he is now currently on vacation in Vienna, Austria.

However, this vacation would soon be interrupted when he realizes that he misread the schedule for the final exam. It won't be taking place after he comes back from vacation... it's taking place *right now*!

Not wanting to spend the rest of his vacation reflecting on being rejected from his dream, he uses the full extent of his abilities to remotely conjure a clone of himself to take the exam, which he will control telepathically.

The exam requires applicants to utilize their psychic powers to navigate a maze, represented by an $r \times c$ grid (r rows and c columns) that we shall call Grid B. Hero A finds a room in Austria with the same $r \times c$ grid shape, and takes the examination from here; let's call this Grid A.



Hero A creates a clone (whom we'll call Clone B) that appears somewhere in Grid B. Whenever Hero A moves, Clone B attempts to mimic that motion as well (e.g. if Hero A takes a step to the right, Clone B will try to step to the right as well). The goal is to have Clone B navigate through the maze set up in Grid B and reach a marked destination.

Unfortunately, the setup isn't perfect. Although it is guaranteed that Grid A and Grid B have the same $r \times c$ dimensions, it's possible that Hero A and Clone B don't start at the same position in the grids. Also, both Grid A and Grid B have obstacles, but these obstacles could be placed in different locations!

Precisely, Clone B is controlled via the following manner.

- Hero A can only move horizontally and vertically, one step at a time.
- Neither Hero A nor Clone B can walk into obstacles, nor step out of the grid.
- When Hero A moves one step in a given direction, Clone B will *try* to move one step in the same direction.
  - In the case where Clone B's move is blocked, Clone B will not move.
  - This will not prevent Hero A from moving.

- If Hero A tries to take a step in a direction blocked by an obstacle or by a wall, neither he **nor Clone B** will move.

Additionally, Hero A needs to finish the exam as quickly as possible before the examiner realizes (they won't notice, right?) he was never there to begin with! Find the fewest number of steps Hero A needs to make in order for Clone B to reach the destination.

Input Format

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

The first line of each test case contains the two space-separated positive integers r and c—the number of rows and columns for Grids A and B.

Then, r lines follow, each containing a string of length c. This encodes the state of Grid A.

Then, another r lines follow, each containing a string of length c. This encodes the state of Grid B.

The possible characters that may appear in each grid are as follows:

- "`.`": Empty space
- "`X`": Obstacle
- "`S`": Starting point (of Hero A in Grid A, or of Clone B in Grid B)
- "`D`": Destination/Goal of the clone (only appears in Grid B)

Output Format

For each test case, output an integer corresponding to the fewest number of moves needed to reach the destination. If the destination is unreachable, output -1 instead.

Constraints

Constraints
$1 \leq T \leq 10$
$1 \leq r, c \leq 100$
$2 \leq r \times c \leq 1000$
"`S`" appears exactly once in Grids A and B.
"`D`" appears exactly once in Grid B.

Sample I/O

Input	Output
2 5 5 S.... ....S ....D 7 5 XXXXX XXXXX XS..X X.X.X X...X XXXXX XXXXX XXXXX XS..X XXX.X X...X XXX.X XD..X XXXXX	4 14

Explanation

In the first test case, though Hero A and Clone B have different starting points, Hero A only needs to move downward for Clone B to reach the destination.

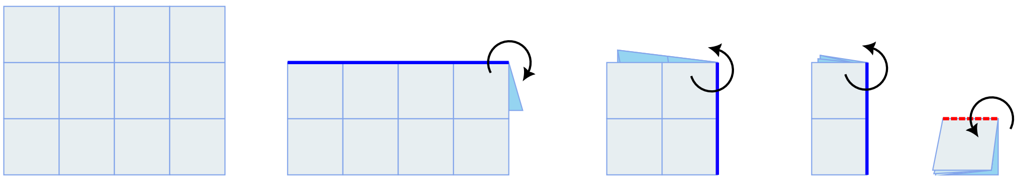
In the second test case, Hero A can have Clone B reach the destination in the optimal number of steps by walking "clockwise" in his grid, twice.

Problem F
Map and Fold

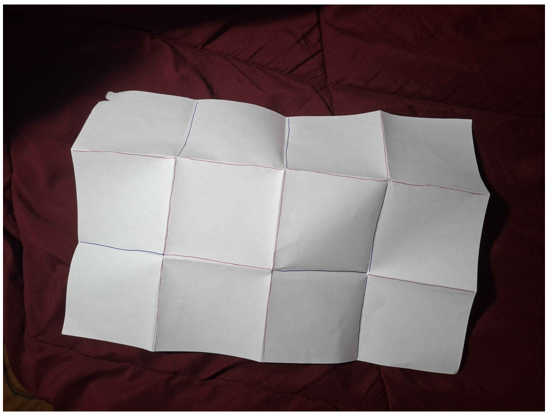
Victorino Mapa loves physical maps. His favorite part of a map is *folding* it up into a little square, and then seeing the crease patterns formed by the folds when he opens it up again. He loves it so much that he will create a cool new notation for folded maps.

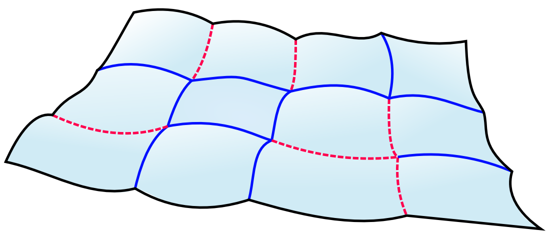
In some order, he performs some sequence of folds: some across a horizontal axis, and some across a vertical axis. Each horizontal fold is made an integer number of units away from the top of the paper, and the crease always spans the entire width of the paper (and similarly so for vertical folds). Victorino stops only when the paper has been folded into a unit square.

Here's an example of some sequence of such folds applied to a 3×4 map.



When Victorino opens this paper up again, he'll get a map that looks something like this:

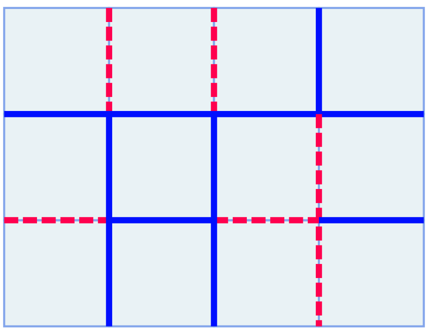




The crease lines create the shape of an $m \times n$ grid of unit squares! Victorino calls this process the *manifold method*, and the result of the manifold method is a *manifolded map*.

Let a *fold* refer to an edge (1 unit long) that is shared by two bordering cells. Each fold can be characterized as one of two types, depending on the *direction* of the crease. Suppose you are directly facing the surface of the map—folds that crease towards you are *mountain folds*, and folds that crease away from you are *valley folds*.

In the following diagram, we flatten the sheet of paper from the above example. Mountain folds have been marked with solid lines, while valley folds have been marked with dashed lines.



Victorino devises the following scheme for encoding a manifolded map into a grid of ASCII characters. Before he describes it to you, let's look at a concrete example—he encoded the manifolded map from earlier.

```
.V.V.M.
M+M+M+M
.M.M.V.
V+M+V+M
.M.M.V.
```

Formally, let s be a $(2r-1) \times (2c-1)$ grid, where $s_{i,j}$ is the character in the ith row from the top and jth column from the left.

- If i and j are both odd then $s_{i,j}$ is `.` to represent a unit square in the grid.
- If i and j are both even then $s_{i,j}$ is `+` to represent a corner where four squares meet.
- Otherwise, $s_{i,j}$ represents the fold that is bordered by the two unit squares that are adjacent to this character; it is `M` if this is a mountain fold, and `V` if this is a valley fold.

Victorino realized that the fun could go the other way around. Given an ASCII grid, the challenge is to make the right choices while performing the manifold method in order to replicate the pattern described by the grid—that is, if it's possible at all.

Given such a grid, please determine (simply `YES` or `NO`) whether or not a manifolded map can be replicated in real life that would produce that pattern when represented as a grid using Victorino's encoding scheme. Also, there will be T independent test cases per file.

Input Format

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

The first line of each test case contains the two space-separated integers r and c.

Then, $2r-1$ lines follow, each containing a string of length $2c-1$.
This is the $(2r-1)\times(2c-1)$ grid whose pattern we wish to replicate using the manifold method.

Output Format

For each test case, output a single line containing either the string `YES` or `NO`

Constraints

Constraints
$1 \leq T \leq 80$ $1 \leq r, c \leq 50$ $2 \leq rc$ The input follows the format of Victorino's encoding scheme.

Sample I/O

Input	Output
2 3 4 .V.V.M. M+M+M+M .M.M.V. V+M+V+M .M.M.V. 2 2 .M. M+M .M.	YES NO

Problem G
Max Cut Min Flow

Oh no, it's flood season! Water from the mountaintop flows down and floods the town at the mountain's base. We need to do something about it!

The trajectory of the flood has been modeled as a straight path with n checkpoints, labeled 1 to n, where checkpoint 1 is the mountaintop and checkpoint n is the town. All water originates from checkpoint 1, and it proceeds through the checkpoints 2, then 3, then 4... and so on, in order, until it finally reaches the town at checkpoint n.

There are $n - 1$ possible *government projects* that you can choose to do, labeled 1 to $n - 1$. Project i involves building a barrier between checkpoints i and $i + 1$. If this project were to be completed, then any water that *would* flow between those checkpoints is instead **blocked**.

Your purse begins with b pesos. The costs of the projects are conveyed through an array of integers $x_1, x_2, x_3, \ldots, x_{n-1}$. Note that these integers could possibly be negative! Why? Well...

As you'd expect, these large government projects typically cost money. If $x_i \leq 0$, then completing project i results in $|x_i|$ pesos being subtracted from your purse (and you cannot choose to do this project unless you have at least $|x_i|$ pesos in your purse).

On the other hand, using *magic*, it is actually possible for a government project to earn you money instead! If $x_i > 0$, then completing project i actually results in x_i pesos being *added* to your purse! Amazing!

You can choose to do as many projects as you like, in any order, however each project may only be done at most once. You must make sure that water is not able to flow from checkpoint 1 to checkpoint n, but among all such ways to do that, what is the maximum possible value that your purse can end on?

If the task is impossible, please say so as well.

Input Format

The first line of input contains the two space-separated integers n and b, the number of checkpoints and the original amount of money in your purse.

The second line of input contains the $n - 1$ space-separated integers $x_1, x_2, x_3, \ldots, x_{n-1}$.

Output Format

If the task is possible, output a single non-negative integer—the maximum possible value of the contents of your purse, among all such ways that accomplish the task.

If the task is impossible, output -1 instead.

Constraints

Constraints
$2 \leq n \leq 10^5$ $0 \leq b \leq 10^9$ $-10^4 \leq x_i \leq 10^4$ for each i.

Sample I/O

Input	Output
`6 7` `3 -1 4 1 -5`	`15`

Input	Output
`5 4` `-6 -7 -6 -7`	`-1`

Explanation

In the first sample input, we choose to do projects 1 and 3 and 4, so our purse ends on a value of $7 + 3 + 4 + 1 = 15$.

In the second sample input, we don't have enough money in our budget to do any of the government projects. Oh no! I hope the people in the town understand :(

Problem H
Prime Topology

Alice was in her graduate-level topology class when she learned that there exists a *topological* proof of the fact that there are infinitely many prime numbers.

Topology, of course, is the mathematical study of topologies.

A topology can be defined as a collection of sets which are suitably well-behaved under set union and intersection—these are called the *open sets*. For the topological proof to work, we need to define a topology on the integers, called the evenly-spaced integer topology.

Anyway, Alice wasn't following along and couldn't understand any of this, so she decided to ignore all of it and doodle something completely unrelated in her notebook instead.

Let $U_n = \{1, 2, 3, \ldots, n\}$. A subset $S \subseteq U_n$ is called *prime-spaced* if it satisfies the ff.:

- For all *distinct* $a, b \in S$, we have that $|a - b|$ is prime.

An integer n is prime if $n > 1$ and n has no positive divisors other than 1 and itself.

For example, suppose we take $n = 6$, and so $U_n = \{1, 2, 3, 4, 5, 6\}$.

- The empty set is (vacuously) prime-spaced.
- $\{1\}$ is (vacuously) prime-spaced.
- $\{1, 6\}$ is prime-spaced, because $|1 - 6| = 5$, which is prime.
- $\{1, 5\}$ is **not** prime-spaced, because $|1 - 5| = 4$, which is **not** prime.
- $\{1, 3, 6\}$ is prime-spaced, because all of $|1 - 3|$ and $|1 - 6|$ and $|3 - 6|$ are prime.
- $\{1, 2, 4\}$ is **not** prime-spaced, because $|1 - 2| = 1$ which is **not** prime.

Given positive integers n and k, how many prime-spaced subsets of size k are there in U_n (modulo 104206969)? Also, there will be T independent test cases per file.

Input Format

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

Each test case is described by a single line containing the two space-separated integers n and k.

Output Format

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

Constraints

Constraints
$1 \leq T \leq 2 \times 10^5$ $1 \leq k \leq n \leq 10^7$

Sample I/O

Input	Output
3 5 2 6 3 10000000 10000000	5 2 0

A completely useless fact

Let X be any (possibly empty) collection of prime-spaced subsets of U_n. Then, $\bigcap_i X_i$ is also guaranteed to be a prime-spaced subset of U_n.

The above fact has nothing to do with this problem, nor with topology. It is completely useless.

Problem I
Stone Steps

It is no surprise that many Filipinos are addicted to the brand new *animé* horse-racing game that has taken the world by storm. The name "Philippines", after all, could be translated to "land of the lovers of horses".[citation needed]

Many people can get quite obnoxious about new fandoms that they get into. Even if it's something completely unrelated, they'll find a way to make the conversation about horses. The worst part is, you know that this person didn't care at all about horse racing just a few months ago, and now they purport to be an expert!

For example, suppose you're in a brainstorming session for ICPC Manila 2025 problems, and then one guy randomly goes:

"Did you know that the Philippines' 2012 Triple Crown winner Hagdang Bato *is a descendant of* Northern Dancer*, the sire of the influential Japanese racehorse* Northern Taste*, who is widely theorized to be the inspiration behind-"*

And even after you shut them up and tell them to focus, they'll say:

"Well, Hagdang Bato can literally be translated to "stone steps". So, if n is a positive integer which doesn't have `0` as a digit*, we can define $H(n)$ to be the number you would get by "sorting" the digits of n in non-decreasing order. Like a staircase!"*

So, for example, $H(1971) = 1179$ and $H(3) = 3$. Hm, wait, this could be interesting...

Let s be a string of nonzero digits. Please compute the sum of $H(\text{int}(s[i...j]))$ across all contiguous non-empty *substrings* of s. Report the value of this sum (modulo 1000696967).

Do it for Hagdang Bato.

Let s be 1-indexed. If $1 \leq i \leq j \leq |s|$, then the substring $s[i...j]$ corresponds to the contiguous segment of characters $s_i s_{i+1} \ldots s_j$. The string s has $|s|(|s|+1)/2$ non-empty substrings.

Also, int(...) is just a function which takes a string as argument, and converts it to an integer (using the decimal system).

Input Format

Input consists of a single line containing the string s.

Output Format

Output a line containing a single integer, the desired sum (modulo 1000696967).

Constraints

Constraints
$1 \leq \|s\| \leq 5 \times 10^5$ s consists only of nonzero digits.

Sample I/O

Input	Output
`3141`	`1432`

Input	Output
`1`	`1`

Input	Output
`11234567891234567891`	`43138332`

Explanation

If $s =$ `3141`, then it has 10 non-empty substrings:

- `s[1...1]` is `3`, and $H(3) = 3$.
- `s[2...2]` is `1`, and $H(1) = 1$.
- `s[3...3]` is `4`, and $H(4) = 4$.
- `s[4...4]` is `1`, and $H(1) = 1$.
- `s[1...2]` is `31`, and $H(31) = 13$.
- `s[2...3]` is `14`, and $H(14) = 14$.
- `s[3...4]` is `41`, and $H(41) = 14$.
- `s[1...3]` is `314`, and $H(314) = 134$.
- `s[2...4]` is `141`, and $H(141) = 114$.
- `s[1...4]` is `3141`, and $H(3141) = 1134$.

Adding them together,

$$3 + 1 + 4 + 1 + 13 + 14 + 14 + 134 + 114 + 1134 = 1432.$$

Problem J
Tic-Tac-Toe on a Graph

Filipino schoolchildren love playing games on the classroom's blackboard or on little whiteboards (that they bring to school for math class) during recess. Alice and Bob used to enjoy playing Tic-Tac-Toe, but ever since they learned that it was a solved game, they got bored with it.

They now want to innovate on the game by playing it on a graph!

They draw n squares on the whiteboard (labeled 1 to n), and also m lines that each connect a different pair of squares (using jargon, they draw a simple undirected unweighted graph, where the squares are the vertices and the lines are the edges). Initially, all squares are empty.

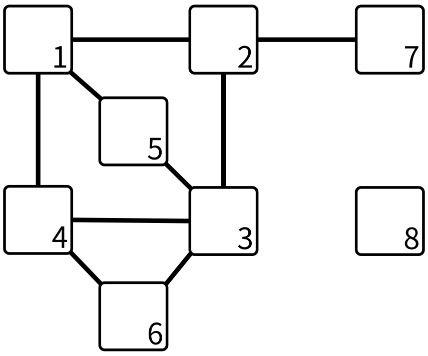
Alice and Bob will take turns; Alice always goes first. On Alice's turn, she chooses an empty square and draws an `X` in it. On Bob's turn, he chooses an empty square and draws an `O` in it.

They still get bored very easily, so **the game ends after five moves**. So, Alice always gets three turns, and Bob always gets two turns.

Alice wins if she can get "three Xs in a row". Formally, the three squares that Alice drew an `X` in must form a *path* of length 3 in the graph (though note that the order of the vertices in the path does not necessarily have to match the order in which she filled in those squares). Bob wins if he can prevent this.

Alice is interested in which starting moves allow her to always win, assuming that she and Bob perform perfect play (after that first move). Alice can "always win" if there exists a strategy she can use (after that first move) that leads her to victory, regardless of what Bob does.

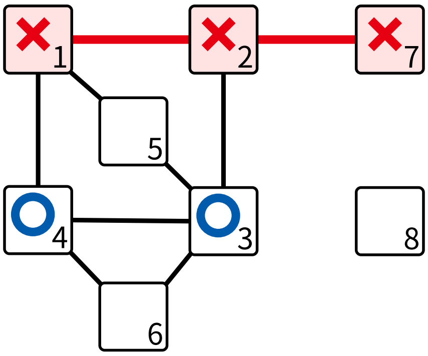
For example, consider the following graph.



If Alice draws an `X` in 1 on her first move, she can always win, regardless of what Bob does. Here is one possible game:

- Alice draws an `X` in 1
- Bob draws an `O` in 4
- Alice draws an `X` in 2
- Bob draws an `O` in 3
- Alice draws an `X` in 7

Alice wins, since she has a three-in-a-row through 1 and 2 and 7.



Note that demonstrating this one game is not sufficient to prove that Alice *always* wins; you would have to show that she has a path to victory *no matter what Bob does*. In this case though, it *is* true that Alice can always win after drawing an `X` in 1 as her first move. Trust us `;)`

We can show that Alice can always win if her starting move is among 1, 2, 3, 4, or 5. On the other hand, she is guaranteed to lose if her starting move is among 6, 7, or 8.

Given the graph, determine how many starting moves lead to a victory for Alice.

Input Format

The first line of input contains two space-separated integers n and m.

Then, m lines follow, describing the edges. Each line contains the two space-separated integers u and v, meaning that a line was drawn connecting squares u and v.

Output Format

Output a single integer, the number of possible starting moves from which Alice always wins.

Constraints

Constraints
$5 \le n \le 2 \times 10^5$ $0 \le m \le \min(2 \times 10^5, n(n-1)/2)$ $1 \le u, v \le n$ in each edge Each edge connects two different vertices, and no pair of vertices are connected by more than one edge.

Sample I/O

Input	Output
8 9 1 2 1 4 1 5 2 3 2 7 3 4 3 5 3 6 4 6	5

Problem K
Toxic Culinarity

The *International Culinary Preparatory College* welcomes a fresh batch of n students, labeled 1 to n. Each one has ambitions of becoming the best chef in the Philippines. At the start of the year, no one is friends yet, but as the year progresses, some of these students will become *"friends"* with one another.

There will be q events throughout the year. Each event specifies two different students u and v, and it means:

- If u and v are not yet friends, then they become friends.
- If u and v are already friends, then they **stop** being friends.

On the competitive cooking website Cookforces, each user is assigned an integer rating from 1 to c. Multiple users can have the same rating. Students in the ICPC are encouraged to aim for a high rating on Cookforces.

A student acts *toxic* if they are not friends with anyone with a strictly higher Cookforces rating than them. The *Ramsay number* of the school is equal to the number of toxic students it has at that particular moment.

One day, the creator of Cookforces got fed up with everyone's obsession over rating, and decided to go with the nuclear option. He announced that he has programmed the website to—without warning, at some unknown time in the future—*completely randomize all the ratings.* Each user's rating will just become an integer uniformly randomly selected from 1 to c.

This, of course, threw the school into complete chaos. After each of the q events described above, please answer the following question:

- If the ratings were randomized *immediately after this event*, what would be the *expected value* (modulo 1224736769) of the Ramsay number of the school?

Suppose we repeatedly perform the random experiment of "randomize all the users' ratings" and then record the Ramsay numbers we get. The expected value is equal to what our running average will converge to (with high probability).

*Formally, let $x_1, x_2, x_3, \ldots$ be an infinite sequence where x_t is the **average** of the recorded Ramsay numbers from the first t random experiments. We can prove that there exist positive integers p and q such that $\lim_{t \to \infty} x_t = p/q$ almost surely.*

Your task is to report, after each event, the value of $pq^{-1} \pmod{1224736769}$; more formally, output the value r such that $p \equiv qr \pmod{1224736769}$. We can show that such an r always exists and is unique (modulo 1224736769), given the constraints.

Input Format

The first line of input contains the three space-separated integers n and c and q.

Then, q lines follow, describing the events in the order they occur. Each line contains some two space-separated positive integers, the values of u and v for this event.

Output Format

Output q lines, each containing a single non-negative integer—the *expected value* (modulo 1224736769) of the Ramsay number immediately after each event.

Constraints

Constraints
$2 \leq n \leq 2 \times 10^5$ $2 \leq c \leq 10^9$ $1 \leq q \leq 3 \times 10^5$ $1 \leq u, v \leq n$ and $u \neq v$ in each event.

Sample I/O

Input	Output
3 2 4 1 2 2 3 1 3 1 2	612368387 1071644675 153092098 1071644675

Explanation

The expected values of the Ramsay number after each event are, respectively:

- 5/2,
- 17/8,
- 15/8,
- 17/8.

We say that, for example,

$$5/2 \equiv 612368387 \pmod{1224736769}$$

because

$$612368387 \cdot 2 \equiv 5 \pmod{1224736769}$$

.

Problem L
Trace of Product of Sparse Square Matrices

Alice was bored in her linear algebra class. She asked her teacher why they were learning any of this. Does linear algebra have *any* real world use whatsoever?

The teacher thought for a few minutes, and eventually came up with this scenario.

Suppose that a genie appears to you and says he will grant you *three wishes* if you can compute the trace of the product of two sparse square matrices A and B with integer entries (modulo 1006903069).

First, you will say, "I wish I remembered how matrix notation works!" and the genie will say that A and B are $n \times n$ grids of numbers. In A, we let $a_{i,j}$ denote the element in the ith row from the top and jth column from the left. The elements of B are similarly indexed $b_{i,j}$. The matrices are sparse, meaning (informally) that they only have a few nonzero entries, relative to their size.

Next, you will say, "That is great, but now I wish I remembered how to do matrix multiplication!" and the genie will remind you that the product AB is another $n \times n$ matrix C, whose elements $c_{i,j}$ are computed using the following formula:

$$c_{i,j} = \sum_{k=1}^{n} a_{i,k} b_{k,j}.$$

Finally, you will say, "Okay, but now I really wish I remembered what the trace was!" and the genie will remind you that the trace of a matrix C is equal to the sum of the elements along its main diagonal, i.e.

$$\mathrm{tr}(C) = \sum_{t=1}^{n} c_{t,t}.$$

Note that if all the entries of A and B are integers, then $\mathrm{tr}(AB)$ is an integer as well.

The genie points out that he now owes you -3 wishes, since he granted you three wishes which have yet to be paid for. Each wish was valued at PHP 6,264,067.84 at time of granting, with an additional 30% interest for each day it is not paid.

You dummy! You landed yourself in debt with a genie all because you didn't pay attention in linear algebra class! Your only way out of this mess is to compute $\mathrm{tr}(AB) \pmod{1006903069}$ and collect the original three wishes promised by the genie.

Alice's teacher apologized and said that he tried really hard, but this was truly the *only* real world application of linear algebra that he could come up with.

But it's good enough for Alice, who now desperately wants to solve this problem!

Input Format

The matrices are sparse, so they will be encoded by specifying the locations and values of their nonzero entries. Formally, the matrix A is encoded as follows:

- You are given k_a triples (i, j, x), meaning $a_{i,j} := x$.
- All given (i, j) index pairs are distinct, and all other elements of A not explicitly mentioned in the input are implicitly equal to 0.

The matrix B is encoded similarly, with its k_b nonzero entries explicitly enumerated.

The first line of input contains a single integer n, the size of the matrices.

This is followed by a line containing an integer k_a. Then, k_a lines follow, each describing a nonzero entry in A. Each line contains three space-separated integers i and j and x, meaning $a_{i,j} := x$. If some (i, j) is **not** mentioned in the input, then $a_{i,j} := 0$.

This is followed by a line containing an integer k_b. Then, k_b lines follow, each describing a nonzero entry in B. Each line contains three space-separated integers i and j and x, meaning $b_{i,j} := x$. If some (i, j) is **not** mentioned in the input, then $b_{i,j} := 0$.

Output Format

Output a single line, containing the value of $\operatorname{tr}(AB) \pmod{1006903069}$.

Constraints

Constraints
$3 \le n \le 10^5$ $1 \le k_a, k_b \le 1.5 \times n$ $1 \le i, j \le n$ and $1 \le x \le 10^9$ in each description. The (i, j) pairs are distinct across all descriptions for A, and similarly for B.

Sample I/O

Input	Output
3 4 2 1 6 2 3 6 3 1 4 3 2 3 4 1 1 2 1 3 3 2 3 3 3 2 1	27

Explanation

You can verify from the definitions that

$$\operatorname{tr}\left(\begin{bmatrix}0 & 0 & 0\\ 6 & 0 & 6\\ 4 & 3 & 0\end{bmatrix}\begin{bmatrix}2 & 0 & 3\\ 0 & 0 & 3\\ 0 & 1 & 0\end{bmatrix}\right) = \operatorname{tr}\left(\begin{bmatrix}0 & 0 & 0\\ 12 & 6 & 18\\ 8 & 0 & 21\end{bmatrix}\right) = 0 + 6 + 21 = 27.$$

Problem M
Web Delivery

Gagamboy, the Philippines' local Spider-Man[3], has a small issue. He's out of web fluid!

Gagamboy needs 1kg each of r different chemicals, labeled 1 to r, in order to synthesize his web fluid. He's too busy fighting crime in Metro Manila to buy these chemicals in person, so he plans to just order them online. The e-commerce app that Gagamboy is using has c sellers, labeled 1 to c, who cater to chemistry hobbyists.

All r chemicals he needs are stocked by all c sellers, although possibly at different prices. These prices can be encoded in a cost matrix A, where $a_{i,j}$ corresponds to the price of 1kg of chemical i if bought from seller j.

Also, for each seller, if Gagamboy orders *any* nonzero number of chemicals from that seller, he must also pay a flat *delivery fee* to have a box containing all those chemicals from that seller shipped to his house. The delivery fees are given in a vector d, where d_j is the delivery fee if *any* chemicals are bought from seller j.

Gagamboy wants to emphasize that this is a *flat* delivery fee. Regardless of how many chemicals you ordered from some seller j (could be one, or two, or all of them, etc.), the delivery fee for that seller will still always be just d_j.

Given A and d, determine the minimum cost for Gagamboy to purchase at least 1kg of each of the r types of chemicals. There will be T independent test cases.

Input Format

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

The first line of each test case contains the two space-separated integers r and c.

Then, r lines follow, each containing c space-separated integers. The value in the ith line from the top and the jth column from the left is $a_{i,j}$.

Finally, one last line follows, containing the c space-separated integers $d_1, d_2, \ldots, d_c$.

Output Format

Output a line containing a single integer, the minimum possible total cost.

Constraints

Constraints
$1 \le T \le 10$ $1 \le r, c$ $r \times c \le 250$ $1 \le a_{i,j} \le 10^{15}$ for each (i, j). $1 \le d_j \le 10^{15}$ for each j.

[3]TL note: *gagamba* is Tagalog for "spider".

Sample I/O

Input	Output
2 3 5 1 3 5 7 9 5 7 9 1 3 9 1 3 5 7 4 3 2 3 4 4 3 1 2 4 2 3 1 4 1 2 3 2 1 2 4 4	11 11

Explanation

In the first test case, here is an optimal solution.

- Buy chemicals 1 and 3 from seller 2. The prices are $a_{1,2} = 3$ and $a_{3,2} = 1$. The delivery fee is $d_2 = 3$.
- Buy chemical 2 from seller 4. The price is $a_{2,4} = 1$. The delivery fee is $d_4 = 3$.

In total, we spent $((3 + 1) + 3) + ((1) + 3) = 11$.