Association for Computing Machinery
University of the Philippines Diliman Student Chapter, Inc.

ALGOLYMPICS 2024

BORDERLANDS

VENTURING BEYOND THE PANELS

PROBLEM SET

FINALS ROUND

APRIL 6, 2024

Access the contest website at the following url:

https://tinyurl.com/2024AlgolympicsFinals

Hidden Test Cases. Your solution will be checked by running it against one or more (usually several) hidden test cases. You will not have access to these cases, but a correct solution is expected to handle them correctly.

Strict Output Format. The output checker is **strict**. Follow these guidelines strictly:

- It is **space sensitive**. Do not output extra leading or trailing spaces. Do not output extra blank lines unless explicitly stated.
- It is **case sensitive**. So, for example, if the problem asks for the output in lowercase, follow it.
- Ensure that your output ends with a newline (by ending with `cout << endl` perhaps, in C++; this is already the default behavior of `print` in Python)
- Do not print any tabs. (No tabs will be required in the output.)
- Do not output anything else aside from what's asked for in the Output section. So, do not print things like "`Please enter t`".

Not following the output format strictly and exactly will likely result in the verdict "*Output isn't correct*".

Use Standard I/O. Do not read from, or write to, a file. You must read from the standard input and write to the standard output.

Submit Code Only. Only include **one** file when submitting: the source code (.cpp, .py, etc.) and nothing else.

No Java Package. For Java submissions, do not include a `package` line.

No Weird Filenames. Only use letters, digits and underscores in your filename. Do not use spaces or other special symbols.

Use Fast I/O. Many problems have large input file sizes, so use fast I/O. For example:

- In C/C++, use `scanf` and `printf`.
- In Python, use `sys.stdin.readline()`

Good luck and enjoy the contest! 😃

Problem A

4 ♦

Welcome to the Borderlands!

The difficulty of this game is 4 ♦, meaning it is an intelligence-based game.

This game is called ***Light Bulbs****.*

Before you is a closed door. Behind that door are three incandescent light bulbs, labeled `1`*,* `2`*, and* `3`*. In front of you are three light switches labeled* `A`*,* `B`*, and* `C`*. Each light switch controls a different light bulb, and your task is to determine which controls which. The light switches all initially begin in the "off" position.*

The catch is that once the door is opened, the states of the switches are frozen *and can no longer be toggled. Furthermore, you may only open the door once—once it is open, it cannot be closed again.*

Luckily, one of your teammates is a puzzle enthusiast, and recognizes this puzzle as *classic*. The following solution is written in your team notebook:

Choose two light switches and flip them on. Wait a few minutes, and then flip one of those two back off. Then, open the door.

- One of the light bulbs will be on. It must be controlled by the light switch that is currently flipped on.
- Touch one of the other light bulbs.
    - If it is **hot**, then it must have been on for a while before being turned off. Therefore, this light bulb is controlled by the light switch that was flipped off.
    - If it is **cool**, then it must never have been on (otherwise it would be hot). Therefore, this light bulb is controlled by the light switch that was never touched.
- In either case, the remaining light bulb's light switch can be determined by process of elimination.

Now, all you need to do is execute! Given a transcript of someone performing this solution, determine which light switch controls each light bulb.

Your time limit is 5 hours. All teams unable to solve this problem before the time is up will be eliminated*.*

Good luck and have fun!

Input Format

Input consists of exactly seven lines of input in the following format, where angle brackets <> mean the value of the enclosed variable is inserted there.

```
TURN ON <x>
TURN ON <y>
WAIT A FEW MINUTES
TURN OFF <x | y>
OPEN THE DOOR
LIGHT <r> IS ON
LIGHT <s> IS <'COOL' | 'HOT'>
```

Here,

- `x` is a light switch, so it is one of `'A'` or `'B'` or `'C'`
- `y` is a different switch, so it is one of `'A'` or `'B'` or `'C'` that is not equal to `x`
- `x | y` is either the value of `x` or the value of `y`
- `r` is a light bulb, so it is one of `1` or `2` or `3`
- `s` is a different light bulb, so it is one of `1` or `2` or `3` that is not equal to `r`
- `'COOL' | 'HOT'` is either the string `'COOL'` or the string `'HOT'`

When the value of a string is formatted into the input, the enclosing single quotes are omitted.

Output Format

Output three space-separated characters, some permutation of `A` and `B` and `C`, corresponding to the light switches that control light bulbs 1, 2, and 3, respectively.

Sample I/O

Input	Output
TURN ON A TURN ON B WAIT A FEW MINUTES TURN OFF B OPEN THE DOOR LIGHT 1 IS ON LIGHT 2 IS HOT	A B C

Problem B

BNA

As you know, a single strand of DNA can be represented as a string consisting of the letters `G`, `C`, `A`, and `T`, encoding a chain of the nucleotides glycine, creatine, adenosine, and thyme. The newly discovered BNA (Boxyribonucleic acid) can consist of up to twenty-six different types of nucleotides, each one represented by a distinct uppercase letter of the English alphabet.

Anyway, naturally, the first thing we're going to do with this discovery is start wantonly editing the BNA of some poor helpless alien creatures, because remember: bioethics doesn't apply to aliens! You will be given a string s of length n, encoding an alien creature's BNA (this string is 1-indexed, because aliens). Please process q commands, each being one of the following types:

- `SWAP <i> <j>` means you should swap the nucleotides at positions i and j of the BNA.
- `COUNT <x> <l> <r>` means you should count the number of times the nucleotide represented by x appears among positions ℓ to r (inclusive) of the BNA; then, output this count.

Please also answer T independent test cases per file.

Input Format

Input begins with a line containing a single integer T, the number of test cases per file. The descriptions of the T test cases then follow.

The first line of each test case contains two space-separated integers n and q.

The second line of each test case contains the string s, consisting of n uppercase English letters.

Then, q lines of input follow, each describing a command in one of the formats described above. These commands should be processed in order.

Output Format

For each test case, output a line containing some number of space-separated integers, the answers to this test case's `COUNT` commands in the order that they appear. Note that if there are no `COUNT` commands, then you should output a blank line for this test case.

This problem is sponsored by Samsung
and was contributed by Samsung R&D Institute Philippines.

Constraints

Let N and Q be the sum of n and q across all test cases.

Constraints
$1 \leq T \leq 100$ $2 \leq n \leq 10^5$ $2 \leq q \leq 10^5$ $N \leq 10^5$ $Q \leq 10^5$ $1 \leq i < j \leq n$ in all `SWAP` commands. $1 \leq \ell \leq r \leq n$ in all `COUNT` commands.

Sample I/O

Input	Output
2 3 3 ABA COUNT A 1 2 SWAP 2 3 COUNT A 1 2 13 7 ABDDACAADBEAA COUNT A 1 13 COUNT A 1 10 SWAP 2 12 SWAP 2 3 SWAP 6 11 COUNT A 3 7 COUNT E 1 7	1 2 6 4 3 1

Problem C
Deal Breaker

You may find it comforting (or depressing?) to learn that even with all the technological marvels in the year 30XX, humanity still has not "solved" romance. In any case, dating apps are still a hugely profitable market, and you're hoping to *break* into the scene with your startup's new app, `Dealbreakr`.

Your service is fundamentally built around the idea of a *deal breaker*, a flaw that is so non-negotiable that no matter any other positive traits of that partner, the relationship would never work out. Your goal is to pair **seekers** with **applicants** that have none of their listed deal breakers.

Your system has n applicants. Research has identified f of the most important flaws in dating, and each applicant must declare on their profile which of these flaws they think they have. Also, via an unrelated process, your app condenses their physical attractiveness, financial status, personality, etc., into a single positive integer a_i, that applicant's desirability rating.

Now, you must process the queries of q different seekers. Each one gives you some list of flaws which they determine to be *deal breakers*. For each one, output the maximum possible desirability rating among all applicants that have none of those deal breakers as a listed flaw.

Input Format

Input begins with a line containing a single integer f.

The next line contains f space-separated strings, the possible flaws.

The next line contains a single integer n, the number of applicants.

Each applicant is then described by two lines. The first line contains a single integer a_i, the desirability rating of this ith applicant. The second line contains a number of space-separated tokens, describing their flaws.

- The first token is `im`, and this is followed by the description of a flaw.
- Each flaw is described by two tokens. The first token is the actual flaw (one of those listed above). Then...
    - If the second token is `and`, the description of another flaw follows.
    - If the second token is `sry`, this signals the end of the line.

The next line contains a single integer q, the number of seekers.

This problem is sponsored by Insight Direct.

Each seeker is described in their own line, with each line containing a number of space-separated tokens, describing their deal breakers.

- The first token is `no`, and this is followed by the description of a deal-breaker.
- Each deal-breaker is described by two tokens. The first token is the actual flaw (one of those listed above). Then...
  - If the second token is `or`, the description of another deal-breaker follows.
  - If the second token is `pls`, this signals the end of the line.

Output Format

For each seeker, output the answer to their query in one line:

- If at least one applicant has none of their deal breakers, output the maximum desirability rating among all such applicants.
- If no applicant has none of their deal breakers, output the string "`LOWER YOUR STANDARDS`" (without the quotes) instead.

Constraints

Constraints
$1 \leq f \leq 20$
Each flaw consists of at most 5 lowercase English letters.
No two flaws have the same name.
$1 \leq n \leq 2 \cdot 10^5$
$1 \leq a_i \leq 10^9$ for all i.
Each applicant has at least one flaw.
No applicant lists the same flaw twice.
$1 \leq q \leq 2 \cdot 10^5$
Each seeker has at least one deal breaker.
No seeker lists the same flaw twice among their deal breakers.

Sample I/O

Input	Output
4 ugly lame crazy weird 4 100 im lame sry 200 im ugly sry 300 im crazy and lame sry 400 im crazy and weird sry 5 no crazy or weird pls no ugly or weird pls no ugly or crazy pls no weird or lame or ugly pls no weird pls	200 300 100 LOWER YOUR STANDARDS 300

Problem D
Look, Up in the Sky!

The planet Xenon is a famous intergalactic tourist site due to its frequent meteor showers. Fun fact! Those meteor showers are actually usually fragments of nearby planets that had exploded due a buildup of pressure in their supermassive cores!

We can model Xenon on the 2D coordinate plane, where the x-axis is the "ground". There are two observation stations located at points A and B, where point A is the origin $(0, 0)$, and point B is at $(b, 0)$, where b is given.

There are n fragments from the meteor shower, and they can be modeled as n (not necessarily distinct) points above the ground between the two observation decks. Formally, the ith fragment is represented by the point (x_i, y_i), where it is guaranteed that $0 < x_i < b$ and $0 < y_i$.

When someone points at something in the night sky, people have a tendency to look in that direction. If *two* different people point into the night sky, people have a tendency to believe they must be pointing at the same thing, and so there must be something at the location pointed at by both of them (even if there's nothing really there). On Planet Xenon, the following event usually happens:

- A person at observation station A chooses a point C from among the meteor shower's fragments and points at it, so we draw the line $\overline{AC}$.
- A person at observation station B chooses a point D (it is permitted to have $C = D$) from among the meteor shower's fragments and points at it, so we draw the line $\overline{BD}$.
- Let E be the intersection of $\overline{AC}$ and $\overline{BD}$.

The **altitude** of point E is equal to its distance from the ground (the x-axis); more simply, it is its y-coordinate.

Please answer q queries of the following kind.

- Suppose that the points C and D are each inpendently uniformly randomly selected from the n possible fragments. Given positive integers s and t, find the exact probability that the altitude of point E is exactly equal to s/t.

Input Format

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

Then, n lines of input follow, each describing a fragment of the meteor shower. The ith line contains the space-separated integers x_i and y_i.

Then, a line follows containing a single integer q.

Then, q lines follow, each describing a query. The jth line contains the space-separated integers s and t for this query.

Output Format

Output q lines, each containing the answer to the respective query. We can show that the answer is a non-negative rational number, so write it in **lowest terms** as c/d (if $c = 0$, then $d = 1$). Then, output it in the format `<c>/<d>`

Constraints

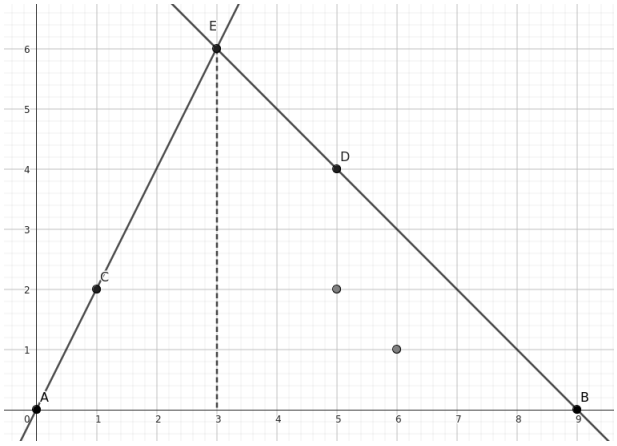
Constraints
$1 \leq n \leq 3 \cdot 10^5$ $1 \leq b \leq 3 \cdot 10^4$ $0 < x_i < b$ for all i $0 < y_i \leq 6$ for all i $1 \leq q \leq 3 \cdot 10^5$ $1 \leq s, t \leq 10^9$

Sample I/O

Input	Output
4 9 1 2 5 2 5 4 6 1 2 6 1 18 7	1/16 1/8

Explanation

If $b = 9$, $C = (1, 2)$ and $D = (5, 4)$, we see that E would have an altitude of 6.



Problem E
Minimum Cost Spanning Subgraph

The planet of Meeneecospanitri orbits the binary star system of Kruskalix and Primmcipia (near the Boruvkadon nebula). There are n countries in this planet, labeled 1 to n. Each country's technological progress is captured by a single number t_i, which we call that country's technology rating.

Due to issues with backwards compatibility and all that junk, two countries u and v can only establish an undersea internet cable directly connecting the two of them if $|t_u - t_v| \leq a$, where a is a to-be-decided **adaptor constant**. If two countries can establish an undersea internet cable between them, they will.

Two countries u and v can communicate with each other if and only if a message can be sent from country u to v through a connecting chain of undersea internet cables. Formally, there must exist a sequence of countries $c_1, c_2, \ldots, c_m$ such that...

- $c_1 = u$ and $c_m = v$
- For all i such that $1 \leq i < m$, there exists an undersea internet cable directly connecting countries c_i and c_{i+1}.

We must help Meeneecospanitri achieve world peace, and to do so, we must ensure everyone in the world can communicate with one another. Obviously it can be done if $a = 10^{10^{100}}$, but that's often overkill. What is the *minimum* adaptor constant a such that once the undersea internet cables are laid, any country can communicate with any other country in Meeneecospanitri?

Input Format

Input begins with a line containing a single integer T, the number of test cases per file. The descriptions of the T test cases then follow.

The first line of each test case contains a single integer n.

The second line of each test case contains the space-separated integers $t_1, t_2, \ldots, t_n$.

Output Format

For each test case, output a line containing a single integer a, the minimum value of the adaptor constant that allows all countries to communicate with one another.

This problem is sponsored by Samsung
and was contributed by Samsung R&D Institute Philippines.

Constraints

Let N be the sum of n across all test cases.

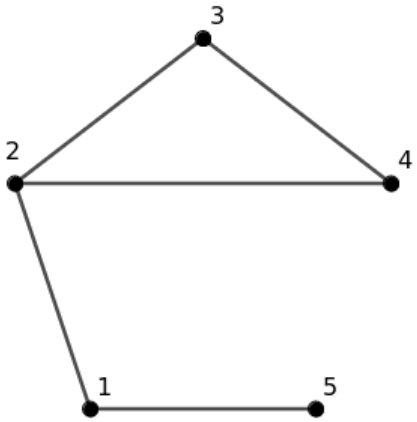
Constraints
$1 \leq T \leq 10^5$ $2 \leq n \leq 10^5$ $1 \leq t_i \leq 10^5$ $N \leq 10^5$

Sample I/O

Input	Output
2 3 3 1 2 5 9 5 1 4 10	1 4

Explanation

For the second test case, if $a = 4$, the following undersea internet cables are established.



Problem F
Meta Sudoku

In the year 2024, sudoku was a widespread pastime for human recreation. By the year 4024, sudoku strategies had become so streamlined and hyper-refined, that humans had to create a new game, **meta-sudoku**, in order to derive sufficient brain stimulation for enjoyment.

By the year 4024, the new popular sudoku mode had become on a 4×4 board, so we'll explain the rules in that context. A **sudoku** is a 4×4 grid of numbers, with each cell containing one of 1, 2, 3, or 4, such that:

- No row has a duplicate value.
- No column has a duplicate value.
- Subdivide the grid into four disjoint 2×2 "subgrids"—none of these subgrids should have a duplicate value.

A **puzzle** is a 4×4 grid such that some of its squares have been filled up (each containing one of 1, 2, 3, or 4)—such a square is called a **clue**. If a sudoku can be produced by filling in the empty non-clue squares of the puzzle (with 1, 2, 3, or 4), then that sudoku is called a **solution** to the puzzle. For example, the grid on the left is a sudoku, and it is a possible solution to the puzzle on the right (where `.` represents a square without a clue).

```
1234    ...4
3412    3...
2143    214.
4321    4...
```

In meta-sudoku, the player is presented with a puzzle and a sudoku. The player must erase *at most* k *clues* from the puzzle, with the goal of making the sudoku a solution to the puzzle.

But you are not a meta-sudoku player. You are a meta-sudoku problem setter! And a lazy one at that. You'd like to recycle the same starting puzzle as much as possible, but still present "technically different" problems by changing the target sudoku.

Given a positive integer k and a puzzle, please count the number of different sudokus that can be made a solution to this puzzle by erasing at most k clues from the puzzle.

Also, you will have to answer T different meta-sudoku instances per test file.

Input Format

Input begins with a line containing a single integer T, the number of test cases per file. The descriptions of the T test cases then follow.

This problem is sponsored by Old St. Labs.

The first line of each test case contains a single positive integer k.

Then, four lines follow, each containing a string of length 4, encoding the puzzle. The digits `1`, `2`, `3`, and `4` correspond to a clue with that value placed into it, and the character `.` corresponds to an empty square.

Output Format

For each test case, output a single line, the answer to that test case.

Constraints

Constraints
$1 \leq T \leq 5000$ $1 \leq k \leq 16$ At least one square in each grid is left empty (not a clue).

Sample I/O

Input	Output
2 1 12.. 34.. ..12 ..3. 4 1..3 .22. .22. 3..4	3 24

Explanation

These are the three possible sudokus for the first sample test case.

1243	1234	1243
3421	3421	3421
4312	4312	2314
2134	2143	4132

The first one is already a solution of the puzzle. The second one can be made a solution by deleting the puzzle's bottommost `3`. The third one can be made a solution by deleting the puzzle's rightmost `2`.

Problem G
Pandemic

You have done well to make it this far in the Goyo's Game Tournament![1] *Ah, but I shall not waste time. The game for this round is called* ***Pandemic****.*

You n *players have been labeled from* 1 *to* n*, and initially, each one of you has been secretly assigned the state of either* ***Healthy*** *or* ***Infected****. We'll say it right now—you* really *don't want to be Infected. Of course, to be fair, the ill-fated infected will have a chance to cure themselves by establishing* ***handshakes****. Each handshake takes place between two players, with the following effect:*

- *If one player is Infected and the other player is Healthy, the virus will be* transferred *from one player to another. Formally, the Infected player becomes Healthy, and the Healthy player becomes Infected.*
- *On the other hand, if both players in the handshake are Infected, their antibodies combine and both of them become Healthy!*
- *But be warned! The game masters have added a cruel twist... If two Healthy players shake hands, then* both *of them will end up Infected!*

Each pair of players should shake hands ***exactly once each*** *(so there will be* $n(n-1)/2$ *handshakes by the end of the game), but it is up to you players to negotiate the order of these handshakes. After all the handshakes are done, we shall check the status of all players: the Healthy players will advance to the next round, and the Infected players will be* eliminated.

What a gruesome game! Will trust and altruism win out in the end? Or will self-service push people to sow seeds of deceit and distrust?

The game is currently in progress. So far, exactly m of the $n(n-1)/2$ handshakes have already been completed, and our protagonist, a bio-engineered clone of Gregorio del Pilar, has been keeping track. Suddenly, the game masters give another plot twist. Surprise! The status of each player (Healthy/Infected) has been revealed, and is now public knowledge!

Gregorio knows it's now time to act. Each player in this competition is either his friend or his enemy, and his goal is to convince everyone to perform the remaining handshakes in some order, such that at the end of the game, all his friends advance and all his enemies are eliminated. Please *count* the number of different ways to order the remaining handshakes, such that all of Gregorio's friends end the game as Healthy, and all his enemies end the game as Infected.

Output the answer modulo $998244353 = 7 \times 17 \times 2^{23} + 1$. Two ways are considered different if there exist handshakes x and y such that handshake x happened before handshake y in one way, but vice versa in the other way.

Also, please answer T separate test cases per file.

[1] The future does sure have a lot of dystopian death game tournaments, doesn't it?

Input Format

Input begins with a line containing a single integer T, the number of test cases per file. The descriptions of the T test cases then follow.

The first line of each test case contains two space-separated integers n and m, the number of players, and the number of handshakes (out of $n(n-1)/2$) that have already been completed.

Then, m lines of input follow, describing the m handshakes that have already been completed, in order. The jth line contains two space-separated integers u_j and v_j, meaning that the jth handshake involved u_j and v_j shaking hands.

The next line of each test case contains a string of length n, encoding the statuses during the grand reveal, after the handshakes described above. The ith character is `H` if the ith player is currently Healthy, and is `I` if the ith player is currently infected.

The final line of each test case contains a string of length n, encoding Gregorio's relationships with the other players. The ith character is `F` if the ith player is a friend, and is `E` if the ith player is an enemy.

Output Format

For each test case, output a single integer, the number of different ways (modulo 998244353) for the remaining handshakes to be completed, such that all of Gregorio's friends end Healthy, and all his enemies end Infected.

Constraints

Let N and M be the sum of n and m across all test cases, respectively.

Constraints
$1 \leq T \leq 100$ $2 \leq n$ $0 \leq m < n(n-1)/2$ $N \leq 3 \cdot 10^5$ $M \leq 3 \cdot 10^5$ $1 \leq u_j < v_j \leq n$ in each handshake No (u_j, v_j) appears twice in the same test case.

Sample I/O

Input	Output
2 3 1 1 3 IHH FFE 5 0 IHIHI FEEEE	2 0

Explanation

In the first test case:

- There are $n = 3$ participants, and so $3(3-1)/2 = 3$ handshakes must be performed: between 1 and 2, between 1 and 3, and between 2 and 3.
- First, $m = 1$ handshakes have already taken place: the handshake between players 1 and 3.
- After that handshake, it is revealed that player 1 is Infected, and players 2 and 3 are Healthy
- Gregorio's friends include players 1 and 2, while player 3 is his enemy.
- The only handshakes left are between 1 and 2, and between 2 and 3. Gregorio must decide how to sequence them such that players 1 and 2 end the game Healthy, while player 3 ends the game Infected.

Let's suppose that Gregorio decides that the handshake between 2 and 3 should come before the handshake between 1 and 2.

- Both 2 and 3 are Healthy, so the game masters Infect both players.
- Player 1 is Infected, but now so is player 2, so after their handshake, both are cured.

It can be shown that performing handshake 1 and 2 *first* also yields the desired outcome. Thus, the answer is 2.

Problem H
Rainbow Energy

In the future, humanity is able to harness the power of rainbows as a renewable source of clean energy! Specifically, humans invented a so-called *light engine* that is highly dependent on the colors of light fed into it.

The light engine is powered by so-called light crystals. Each light crystal emits light of a certain color and does so with an indicated **radiation rating**, both values represented by integers. Suppose you have some collection of crystals, and wish to use it to fuel the light engine...

- Any number of times, you may choose two crystals that emit same-colored light, and combine then into a single crystal that also emits that color of light, but whose radiation rating is the *sum* of the radiation ratings of the two crystals used as input.
- This is important because if two different crystals fed into the light engine emit the same color of light, everything explodes. So make sure you consolidate your same-colored light crystals first!
- Then, the luminosity of the light engine is equal to the *product* of the radiation ratings of the distinctly-colored light crystals fed into it.

On an expedition at an empty exoplanet, you discover n crystals, the ith of which has a color of c_i and a radiation rating of r_i. Your knapsack can only hold m of them though—if you choose the crystals optimally, what is the maximum luminosity of a light engine that you can power using your chosen crystals?

Input Format

Input begins with a line containing a single integer T, the number of test cases per file. The descriptions of the T test cases then follow.

The first line of each test case contains two space-separated integers n and m.

Then, n lines follow, describing the discovered light crystals. The ith line contains the two space-separated integers c_i and r_i.

Output Format

For each test case, output a single line containing the maximum luminosity possible if the m light crystals in this test case are optimally chosen.

This problem is sponsored by Samsung
and was contributed by Samsung R&D Institute Philippines.

Constraints

Constraints
$1 \leq T \leq 50$ $1 \leq m \leq n \leq 15$ $1 \leq c_i \leq n$ for each i. $1 \leq r_i \leq 5$ for each i.

Sample I/O

Input	Output
2 3 2 1 3 1 3 2 3 4 3 2 2 2 2 3 3 3 4	9 16

Explanation

Let (c, r) represent a light crystal with color c and radiation rating r.

In the first test case, the optimal choice is to pick a $(1, 3)$, and $(2, 3)$. Our crystals all emit distinct colors of light, so we feed both into the machine, getting a luminosity of $3 \times 3 = 9$.

In the second test case, the optimal choice is to pick $(2, 2)$, $(2, 2)$, and $(3, 4)$. First we combine the two $(2, 2)$ crystals into a $(2, 4)$ crystal. Now that our crystals all emit distinct colors of light, we feed them both into the machine, getting a luminosity of $4 \times 4 = 16$.

Problem I
Ready Player Juan

Sure, the future may be dystopian, but at least we have VR headsets! The game *Ready Player Juan* thrusts players into the Metaverse's rendition of Zihuatanejo, Mexico, a beautiful beach that serves as a metaphor for paradise.

Of course, even in paradise, death is there—in Ready Player Juan, this manifests as n super-powerful bosses that you need to defeat in a grueling boss rush (this is all a complicated metaphor for purgatory). These bosses must be defeated *in order*, although for each boss, you have two options on how you want to fight them.

- If you fight them head-on, you have to deal with them at their full power, represented by their **true strength** t_i.
- If you choose to ambush them, you only have to contend with their **ambushed strength** a_i, and it is guaranteed that $a_i < t_i$.
    - However, ambushing a boss makes *all* future bosses wiser to your tricks. After ambushing some boss, the true strengths *and* ambushed strengths of all succeeding bosses are **doubled**.

The challenge of the boss rush is equal to the sum of the strengths of each boss in your chosen method of engagement, across all bosses. You wish to make money off of Ready Player Juan by creating a fan-website that you can stuff with ads and maybe even a bitcoin miner. Thus, you must be able to answer: what is the minimum possible challenge of the boss rush, if a player were to choose optimally in how to engage each boss?

Also, please answer T separate test cases per file.

Input Format

Input begins with a line containing a single integer T, the number of test cases per file. The descriptions of the T test cases then follow.

The first line of each test case contains a single positive integer n.

The second line of each test case contains the n space-separated integers $t_1, t_2, \ldots, t_n$.

The third line of each test case contains the n space-separated integers $a_1, a_2, \ldots, a_n$.

Output Format

For each test case, output a line containing a single integer, the minimum possible challenge of the boss rush.

This problem is sponsored by Navitaire.

Constraints

Let N be the sum of n across all test cases.

Constraints
$1 \leq T$ $1 \leq n$ $N \leq 2 \cdot 10^5$ $1 \leq a_i < t_i \leq 10^9$ for all i

Sample I/O

Input	Output
1 3 4 9 6 1 3 5	17

Explanation

A challenge of 17 can be achieve by the following sequence.

- Fight the first boss head-on, whose true strength is 4.
- Ambush the second boss, whose ambushed strength is 3. All future bosses double in strength. The next boss (the only boss after it) now has true strength 12 and ambushed strength 10.
- Ambush the third boss, whose ambushed strength is 10.

A challenge of $4 + 3 + 10 = 17$ is optimal.

Problem J
Reservoir Doggos

Many celestial bodies in the solar-system were believed to not be worth terraforming due to being inhospitable to human life. However... that doesn't necessarily mean that *dogs* can't live there!

Titan (the moon of Saturn) hosts oceans of liquid hydrocarbons, surpassing even all the known natural gas and oil reserves on Earth. The oil extraction on Titan is managed by a pack of sapient dogs lovingly called the *Reservoir Doggos*. Technology may have granted these animals enough intelligence to perform skilled labor, but they're still adorable.

Unfortunately, disaster struck the Titan Oil Rig! A cosmic storm caused n holes to appear along the reservoir. Repairing the ith hole takes exactly t_i seconds of *continous* works, and it requires the combined efforts of the entire pack, so only one hole can be repaired at a time. Until the very *moment* that this hole is finished being fixed, it leaks oil at a constant rate of r_i units per second.

Oh no! The Reservoir Doggos aren't smart enough to perform operations research. Help determine the minimum amount of oil that will be lost from the reservoir if the dogs fix the holes in the optimal order. Also, please answer T separate test cases per file.

Input Format

Input begins with a line containing a single integer T, the number of test cases per file. The descriptions of the T test cases then follow.

The first line of each test case contains a single positive integer n.

The second line of each test case contains the n space-separated integers $v_1, v_2, \ldots, v_n$.

The third line of each test case contains the n space-separated integers $t_1, t_2, \ldots, t_n$.

Output Format

For each test case, output a line containing a single integer, the minimum amount of oil lost if the holes are fixed in the optimal order.

Constraints

Let N be the sum of n across all test cases.

Constraints
$1 \leq T$ $1 \leq n$ $N \leq 2 \cdot 10^5$ $1 \leq v_i, t_i \leq 10^9$ for all i

Sample I/O

Input	Output
2 1 123 456 3 6 24 30 14 36 60	56088 4404

Explanation

In the second test case, it is optimal to first fix the second hole, and then the third hole, and then the first hole.

Problem K
Space Colonizers

Goodness, space colonization sure is tough! You are the vanguard force of an expedition team whose goal is to colonize distant planets for human civilization. Except, you had sort of forgotten the part of "colonization" that entails a lot of genocide and oppression of the natives living in the *totally unoccupied land*. You're not happy about this part of the job, although admittedly the aliens you're slaughtering en masse are probably even less excited about it than you are.

The colonized planet can be represented as a grid with r rows and c columns. The rows are labeled 1 to r from top to bottom, and the columns are labeled 1 to c from left to right. Let (i, j) represent the cell in the ith row and jth column.

Your ship lands in the cell (i_s, j_s). The planet's Unobtanium mines are at (i_t, j_t). Of course, every other square besides the one you landed in (including the Unobtanium mines) is guarded by a native alien (an oxymoron?) who is none too happy about humanity's tendency to destroy ecosystems when harvesting natural resources. Each cell on the grid is represented by an integer $x_{i,j}$:

- If $(i, j) \neq (i_s, j_s)$, then $x_{i,j}$ is the **power rating** of the alien at that cell.
- If $(i, j) = (i_s, j_s)$, then $x_{i,j}$ is your own initial power rating.

You can move from your current cell to any adjacent cell (i.e. any of the cells that is exactly one step north, south, east, or west of your current cell). If this cell contains an alien, then after moving into the cell, the alien will try to fight you.

- If your power rating is less than or equal to the power rating of the alien there, you die. This is not ideal.
- If your power rating is strictly greater than the power rating of the alien there, these three things happen:
    - First, you kill the alien in that cell.
    - Second, your power rating increases by the strength rating of the alien you just defeated. Experience makes you stronger!
    - Third, any nearby aliens might try to take revenge. If any of the adjacent cells contains an alien with a strictly greater power rating than your own (after the increase), that alien attacks and kills you and you die; again, this is not ideal. Otherwise, nothing happens.

Can you make it to the Unobtanium mines without dying? Do it for humanity! Also, please answer T independent test cases per file.

Input Format

Input begins with a line containing a single integer T, the number of test cases per file. The descriptions of the T test cases then follow.

This problem is sponsored by Samsung
and was contributed by Samsung R&D Institute Philippines.

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

Then, r lines follow, each containing c space-separated integers, representing the grid, such that the jth integer in the ith row denotes the value of $x_{i,j}$.

Finally, a single line containing four space-separated integers will follow, giving the values of i_s, j_s, i_t, and j_t, in that order.

Output Format

For each test case, output a line containing `UNOBTANIUM` if the mines can be reached without dying, or `UNOBTAINABLE` if not.

Constraints

Let A be the sum of $r \times c$ across all test cases.

Constraints
$1 \leq T \leq 100$ $1 \leq r, c \leq 1000$ $1 \leq x_{i,j} \leq 10^9$ for all i, j $A \leq 10^5$ $1 \leq i_s, i_t \leq r$ $1 \leq j_s, j_t \leq c$ For all cells (a, b) adjacent to (i_s, j_s), we have $x_{a,b} \leq x_{i_s,j_s}$.

Sample I/O

Input	Output
3 3 3 3 1 5 2 4 10 4 8 20 1 1 3 3 3 3 3 1 5 2 4 12 4 11 1 1 1 3 3 1 7 8 3 1 2 2 5 13 1 4 1 7	UNOBTANIUM UNOBTAINABLE UNOBTANIUM

Problem L
SwapSwap++++

The programming language `Swap++` is very peculiar.

Every `Swap++` program begins by reading an array of n integers $a = [a_1, a_2, a_3, \ldots, a_n]$. It then executes each line in the program, one by one, in order from top to bottom. This isn't too complicated, because the `Swap++` language only has one kind of command, one on each line:

- `SWAP <i> <j>` which swaps the values of a_i and a_j in the array.

When it is done executing all commands in order, the program ends by printing out the value of the array.

For example, suppose that $a = [5, 6, 7]$. Then, the following `Swap++` program:

```
SWAP 1 2
SWAP 2 3
```

prints out $[6, 7, 5]$.

Now, let's get funky. We create a new programming language `SwapSwap++++`, which is barely even a language if we're being completely honest, because it only does one thing. A `SwapSwap++++` program consists solely of a single permutation of the integers from 1 to c, which we denote by $p_1, p_2, \ldots, p_c$.

When executed, the `SwapSwap++++` program reads, as input, a `Swap++` program with c commands. Then, it produces a *new* `Swap++` program by permuting the order of the commands in the input program: line p_1 comes first, and then line p_2, and so on, ending with line p_c.

For example, let the `SwapSwap++++` program be `2 1`, and we feed it the `Swap++` program above. Its output is thus:

```
SWAP 2 3
SWAP 1 2
```

If you feed $a = [5, 6, 7]$ into *this* program, the output is instead $[7, 6, 5]$.

You are given an array $a = [a_1, a_2, \ldots, a_n]$ as input, as well as a `Swap++` program S. Determine if there exist two *different* `SwapSwap++++` programs such that...

- Let P and Q be the respective `Swap++` programs created by feeding S as input into the two given `SwapSwap++++` programs.
- Then, P and Q should produce the same output when the given array a is passed into each of them as input.

Input Format

Input begins with a line containing a single integer n.

This problem is sponsored by Expedock.

The next line containing n space-separated integers, $a_1, a_2, \ldots, a_n$.

The next line contains a single integer c, the number of commands in the `Swap++` program.

Then, c lines follow, with each line describing a command in the program, formatted as described above. These lines are indexed 1, 2, 3, ..., c in the order they are given in the input.

Output Format

First, output a line containing `YES` if the task is possible, or `YES` if it is not.

If `YES`, also output two more lines, each containing a space-separated permutation of the integers from 1 to c. Let's label these $p_1, p_2, \ldots, p_c$ and $q_1, q_2, \ldots, q_c$. These correspond to the two `SwapSwap++++` programs you are proposing. It must **not** be the case that $p = q$, i.e. there should exist some i such that $p_i \neq q_i$.

If there are multiple possible answers, any will be accepted.

Constraints

Constraints
$1 \leq n \leq 2 \cdot 10^5$ $1 \leq a_k \leq 10^9$ for each k. $2 \leq m \leq 2 \cdot 10^5$ $1 \leq i, j \leq n$ and $i \neq j$ for each i, j in each command

Sample I/O

Input 1	Output 1
5 3 1 4 1 5 7 SWAP 1 2 SWAP 1 3 SWAP 1 4 SWAP 1 5 SWAP 2 3 SWAP 2 5 SWAP 3 4	YES 1 6 7 4 5 3 2 5 7 3 6 4 2 1

Input 2	Output 2
3 5 6 7 2 SWAP 1 2 SWAP 2 3	NO

Explanation

These are the two `Swap++` programs produced by the proposed `SwapSwap++++` in the first sample output:

```
SWAP 1 2
SWAP 2 5
SWAP 3 4
SWAP 1 5
SWAP 2 3
SWAP 1 4
SWAP 1 3
```

```
SWAP 2 3
SWAP 3 4
SWAP 1 4
SWAP 2 5
SWAP 1 5
SWAP 1 3
SWAP 1 2
```

When $a = [3, 1, 4, 1, 5]$ is fed into them, both output $[5, 1, 4, 3, 1]$.

Problem M
Yet Another Arbitrary Polynomial Problem

This problem has no standard input. Whaaaaat? That's crazy!

Alice and Bob were strolling along the beach one moonlit night, when Bob gasped and pointed at the star-filled sky. "How many stars do you think are in the sky, Alice?" he asked.

Alice squinted at the sky for a few seconds, before saying, "I think there are 426969 stars visible in the sky tonight."

"Wow!" Bob said. "That reminds me of the fact that there exists at least 426969 different positive integer solutions (a, b, c) to the equation

$$2023ab^3 + 1858abc + 1084670664a^2 = 2024b^2c^2 + 1966b^4c + 36051705c^3$$

where all of a, b, and c are $\leq 10^{18}$."

"Wow, what an awesome and natural story!" Alice said. "It sure would be awesome if someone could write a program that enumerates such solutions to that polynomial. That person would be my hero!"

Output Format

Output 426969 lines, each one containing three space-separated integers a, b, c. Each triplet should be a valid solution to the given polynomial, with each a and b and c being a positive integer no more than 10^{18}.

Any 426969 solutions will be accepted, so long as all outputted triplets are distinct and have all values within the bounds.

Sample O

These are not actual solutions to the above polynomial. This section only showcases the output format (although the ellipsis is a stand-in for the other 426965 solutions).

Output

```
1 2 3
4 5 6
7 8 9
...
427000 31415926 1000000000000000000
```