icpc international collegiate programming contest

ICPC Asia West Regional 2020

ICPC Asia Dhaka Regional Contest

Official Problem Set

14th August 2021

You get 24 Pages, 12 Problems & 240 Minutes

Problem A

The Revenge Of Anti Hash

Given a base **B** and a modulus **M**, the polynomial hash of a string **S**, consisting of only lowercase letters **(a-z)** is defined as below:

```
int GetHash(string str, int B, int M){
      long long hash = 0;
      for (auto chr: str)
            hash = (hash * B + chr - 'a' + 1) % M;

      return hash;
}
```

In other words, first, the letters of the string are replaced by numbers (equivalent to their position in the alphabet, **'a'** gets mapped to **1**, **'b'** to **2, ...** and **'z'** to **26**). This is then considered to be a number in base **B** (the rightmost number is the least significant digit), and the value of this number taken modulo **M** is called the polynomial hash of the string.

Limak the bear loves to hack other contestants in Codeforces. After the recent educational round, he came to know that his friend Swistak used the polynomial hash function stated above to solve the hardest problem! And believe it or not, he was the only one to solve that problem which eventually made him the round champion! Limak is very angry, how can Swistak solve a problem which Limak himself couldn't solve? And worst of all, Swistak used hashing to solve that problem. Limak believes people who use hashing have no real skill, getting 'Accepted' just implies getting lucky, nothing more.

Limak is just a little bear, he is not very good at solving problems. But after hours of scratching his head, he was able to come up with a solution involving *birthday attacks*. That should hack Swistak's solution since educational rounds allow hacking for 24 hours after the round ends. And voila! When he coded it later that night, he was finally able to come up with a case that broke Swistak's solution. And down he goes. From the top place to 153, even below Limak! Limak was overwhelmed with joy.

When Swistak woke up the next morning and casually checked the rank list he was furious. He could not believe what he saw, rolling his eyes in disbelief. He was the only contestant to solve the last problem and that earned him the top place. But alas! No more, because someone hacked his solution and he dropped down more than hundred fifty places. He clicked on the problem to see who hacked him, and his disbelief grew to anger and frustration when he realized it was his friend Limak! He couldn't believe his eyes. "I thought he was my friend, how could he do this to me?", he wondered. He vowed to take revenge. He modified his solution to use double hashing. But just to be extra sure so that Limak can never hack his solution ever again, he hashed

it a few more times resulting in **K** total hashes. Then he submitted his solution which passed the tests and Limak's initial hack as expected.

Afterwards, he rushed to Limak's place and challenged him to a duel. He claimed Limak was jealous and just got lucky while hacking his solution and has no real hacking skills. Feeling overconfident with his new solution, Swistak challenged Limak to hack his new solution and suggested he will retire from competitive programming if Limak can hack his new solution. But if however Limak fails, then Limak must retire instead!

Limak, being provoked like this, takes up the challenge without thinking it through. But he has no clue how to solve it, he is just a little bear after all. He thought about it throughout the whole day but has no idea on how to crack it. With just 4 hours left before the hacking phase ends, he desperately turns to you for help. He knows this isn't exactly fair, but nothing's fair in love and war as they say and he doesn't want to retire from competitive programming. Not now and not ever. Please help Limak solve the following problem and beat Swistak once and for all, thereby saving his career.

Limak will give you **K** pairs of numbers, **(B_1, M_1), (B_2, M_2), … , (B_K, M_K)**. Each pair consists of a base **B** and a modulus **M**. These are the numbers Swistak used to hash strings **K** times in his new solution. Limak needs you to find two **different** strings consisting of lowercase letters only. The strings must have the same hash value when hashed with each of the **K** base/mod pairs with the above described function. Since Codeforces will not accept just any string of arbitrary length as hack inputs, each of the strings also need to be **non-empty** and cannot exceed more than **65536** characters in length. They can be of different lengths though.

Input

The first line contains **T**, denoting the number of test cases. Then **T** test cases follow. The first line of each case contains an integer **K**. The next **K** lines consist of two integers **(B_i, M_i)**. These are the base and mod pairs Swistak used in his hash function.

Constraints

- **$1 \leq T \leq 20$**
- **$1 \leq K \leq 12$**
- **$32 \leq B_i \leq 256$**
- **$1 \leq M_i \leq 256$**

Output

For each test case, output the required two strings separated by a single space. If there is more than one solution satisfying all the above criteria then you may output any of them. You can be assured that there will always be at least one pair of strings.

Sample Input	Output for Sample Input
1 **1** **32 1**	**hello world**

Problem B

Password Shuffling

Working in the IT department of Reynholm Industries is surely hard work. Especially when they have profited 1800 Billion Billion Pounds. Moss is having doubts about the security of the computers, since the employees would just use passwords like their name or “password”. So he decided to generate passwords for everyone. The passwords have these properties:

- They are randomly generated.
- Their length is at least **6** and can be at most **50**. The length is not random.
- The password only contains numerical (0-9), lowercase (a-z) and uppercase (A-Z) letters. Each character of the password is chosen independently and each alphanumeric letter has the same probability to be chosen.

Roy came up with a game to play while they generate the passwords for all employees. He would take a password, split it into multiple groups, shuffle these groups and then concatenate them back together to make a new string. The shuffle is not random, as Roy can shuffle the groups any way he wants. Then he asks Moss to guess how many groups he can split the string into. Moss’ guess would be correct if he can tell the minimum number of groups required to end up with the new string. Can you help him write a program like that?

Input

The first line of the input is an integer **T** (**1 ≤ T ≤ 100)**, denoting the number of test cases. The next **T** lines each contain two strings, $\mathbf{S_1}$ (**6 ≤ $|S_1|$ ≤ 50)** is the original password and $\mathbf{S_2}$ is the password string that is generated after Roy’s shuffle. It is guaranteed that you can always build $\mathbf{S_2}$ from $\mathbf{S_1}$.

Output

For each test, print one line in the format “**Case X: Y**”, where X is the case number and Y is the minimum number of groups the first string needs to be split into so that we can end up with the second string by shuffling the groups.

Sample Input	Output for Sample Input
4 **abcdAs sAdcba** **123abc 123abc** **RUeGz2tjSy0Sbrs3poVA rs3poVARUeGz2tjSy0Sb** **abcd1234 1234abcd**	**Case 1: 6** **Case 2: 1** **Case 3: 2** **Case 4: 2**

Please note that the sample input is for demonstration only, they are not randomly generated. The actual input will be randomly generated.

Sample Explanation

For the third input, the first string can be split into “RUeGz2tjSy0Sb” and “rs3poVA”, which would result in the second string after changing the order and merging them together.

Problem C

Chip's Challenge

Chip's Challenge is a classic top down puzzle game played on a **N x M** grid. Released in 1989, the protagonist of the game is the high-school nerd Chip McCallahan, who has met Melinda The Mental Marvel in the school science laboratory and fell in love at first sight! Chip must navigate through Melinda's clubhouse, consisting of several increasingly difficult puzzles, in order to prove himself and acquire Melinda's heart and also gain access to the very exclusive Bit Busters Club. Such is the premise of this vintage game, cliched, but who cares about the plot when the gameplay is so cool?

The game consists of more than one hundred two-dimensional levels. Gameplay involves using the arrow keys of the numeric keyboard or mouse to move Chip around each of the levels in turn. To progress through each level, Chip needs to interact with various game elements like computer chips, buttons, locked doors, water and lethal monsters. There are also some levels containing block-pushing puzzles and dodging enemies! Chip can move on to the next level by collecting enough chips to open the chip socket at the end of each level and get to the exit. Most levels have a time limit and levels can also be skipped by entering an appropriate four letter password.

Below is an example of a level from the actual game.

In this problem, we will attempt to solve a simplified variation of the game Chip's Challenge. You will be given a **N x M** grid with Chip's starting position and also the position of Melinda The Mental Marvel. The grid will consist of empty cells, water, lethal monsters and locked doors only. The rules of this game are a bit different than the original version, so pay close attention to the following details. Chip starts from his initial position and wants to travel to Melinda's location. To do so, he can navigate to any four of the adjacent cells from his current position using the arrow keys up, down, left and right. Provided the cell he wants to move to is empty and inside the grid, he will move to the cell and the whole process takes **exactly 1 second**. Otherwise, Chip is not allowed to make a move.

He can also click on any cell that is not empty and attempt to destroy it. Any cell that is not empty can be destroyed and once destroyed, the cell becomes empty for the rest of the game. Note that the cell does not necessarily have to be adjacent to Chip's current position while destroying it. Initially empty cells are marked using the character **'.'** (dot) and both Chip's starting position and Melinda's positions are initially empty cells and are different. Every non-empty cell has a power value and destroying a cell with power value **X** takes exactly **X** clicks, hence **X** seconds in total. That is, per click it takes 1 second. Keep in mind that the non empty cells can either consist of water, monsters or locked doors.

Cells containing water are marked using lowercase letters, therefore there can be **26** different types of water. The power value of a cell containing water is equivalent to its position in the alphabet. That means if a cell contains the character **'a'**, it's power value is **1**, if a cell contains the character **'b'** it's power value is **2**, and similarly for **'z'** it's power value is **26**. Monsters are marked with uppercase letters, and their power values are calculated by adding **26** with its position in the alphabet. So if a cell contains **'A'**, the power value of it will be **27**, for **'B'** it will be **28** and similarly for **'Z'** it will be **52**. Lastly, cells containing locked doors will be represented by the digits **0-9**. The power value of a cell containing a locked door will be the value of the digit plus **53**. If a cell contains a locked door labeled **'0'**, it's value will be **53**, for a cell having locked door labeled **'1'**, the power value will be **54** and for the locked door **'9'**, the power value will be **62**.

Chip can't wait to rejoin the love of his life and join the Bit Busters club with her and he wants to reach her in the shortest time possible. Given the description of the maze, the initial position of Chip McCallahan and Melinda The Mental Marvel, can you suggest how to play Chip's Challenge optimally so that Chip can reach Melinda as fast as possible?

Input

The first line will contain a single integer **T** and then **T** test cases follow. Every test case contains two lines. The first line contains 6 integers - **N, M, C_X, C_Y, M_X, M_Y**. The grid dimensions are denoted by **N** and **M**, **(C_X, C_Y)** indicates the starting position of Chip and **(M_X, M_Y)** the starting position of Melinda. The maze will be generated pseudorandomly. The second line of every case contains one integer called **seed**, and the maze is generated by following the C++ code snippet described below. Note that seed is a global variable that is initialized once at the start of every test case with the value given, and gets updated after each **update_seed()** call.

```
long long seed;

char maze[2021][2021];

void update_seed(){
    seed = (seed * 1000003 + 10007) % 1000000009;
}

void gen_maze(int N, int M, int Cx, int Cy, int Mx, int My){
    for (int i = 1; i <= N; ++i){
        for (int j = 1; j <= M; ++j){
            if ((i==Cx && j==Cy) || (i==Mx && j==My)){
                maze[i][j] = '.';
                continue;
            }

            update_seed();
            int power = seed % 63;

            if (power == 0)
                maze[i][j] = '.';

            else if (power <= 26)
                maze[i][j] = power - 1 + 'a';

            else if (power <= 52)
                maze[i][j] = power - 27 + 'A';

            else
                maze[i][j] = power - 53 + '0';
        }
    }
}
```

Constraints

- **$1 \le T \le 10$**
- **$1 \le N, M \le 1000$**
- **$1 \le C_X, M_X \le N$**
- **$1 \le C_Y, M_Y \le M$**
- **$(C_X, C_Y) \ne (M_X, M_Y)$**
- **$0 \le seed \le 1000003$**

Output

Print the case number followed by the shortest time to complete Chip's Challenge if played optimally. Check the sample for more details.

Sample Input	Output for Sample Input
2 3 3 1 1 3 3 3 1 3 1 1 1 3 1	Case 1: 29 Case 2: 59

Explanation

For the first test case, the maze looks like:

.bl

JaS

Gv.

The shortest path from (1,1) -> (3,3) is the following:

- Destroy cell (1, 2) in **2** seconds
- Destroy cell (2, 2) in **1** second
- Move right from cell (1, 1) to cell (1, 2) in **1** second
- Move down from cell (1, 2) to cell (2, 2) in **1** second
- Destroy cell (3, 2) in **22** seconds
- Move down from cell (2, 2) to cell (3, 2) in **1** second
- Move right from cell (3, 2) to cell (3, 3) in **1** second
- Therefore the overall journey takes **2 + 1 + 1 + 1 + 22 + 1 + 1 = 29 seconds**

Problem D

Constructing Strings

Let's define a special string **S** which follows the following conditions :

1. **S** has length **N**
2. **S** consists of lowercase English letters only
3. It satisfies **Q** conditions of the following format :
   L R : Substring of **S** starting from **L** to **R** (both inclusive) must be palindrome (1-based indexing).

You have to count the number of **distinct special** strings. As the answer can be large, print it modulo **998244353.**

Input

The first line will contain a single integer **T(1 ≤ T ≤ 10)**, denoting the number of test cases.

The first line of each test case will contain 2 space separated integers : **N Q (1 ≤ N, Q ≤ 10^5).**

The next **Q** lines of each test case will contain 2 space separated integers : **L R (1 ≤ L ≤ R ≤ N)**, describing a condition.

Output

For each test case, first print the case number and then print the number of valid strings for that test case modulo **998244353**.

Sample Input	Output for Sample Input
2 2 1 1 2 12 4 1 3 10 11 1 5 4 6	Case 1: 26 Case 2: 45855352

Problem E

The Lieutenant's Exam

Sergeant Terry Jeffords is attending an exam where he has to answer only one MCQ question. There are **N** options for this question. He has to pick exactly one option as his answer. Some of them are correct and the rest are wrong. So if he picks any of the correct options, he will pass the exam. But unfortunately, Terry doesn't know which options are correct.

These **N** options for the question are actually **N** distinct binary strings. Terry suddenly notices that the teacher in the exam room is writing a binary string on the whiteboard. He suspects that the answer to the question is a subsequence of the string that is being written by the teacher.

The teacher is writing an infinitely long binary string. She starts with an empty string. Then in each step, she writes a digit or removes the last written digit. This is how she decides what to do in each step:

If this is not the first step and she has not removed a digit in the previous step:

i) She writes "0" with **X** probability

ii) She writes "1" with **Y** probability

iii) She removes the last written digit with (**1-X-Y**) probability.

Otherwise,

i) She writes "0" with $\frac{X}{(X+Y)}$ probability

ii) She writes "1" with $\frac{Y}{(X+Y)}$ probability

So the teacher never removes digits in two consecutive steps.

If after any step, the currently written string on the whiteboard contains any of the **N** options as a subsequence, Terry would pick it as the answer and end the exam. If there are multiple options that are subsequences of the current string, Terry will pick the option which has come first in the input options.

What is the probability of Terry picking a correct answer?

Input

The first line will contain a single integer **T** denoting the number of test cases. Each test case will have an integer **N** and two numbers **X** and **Y** in the first line. Next line will contain **N** space separated binary strings. The last line of the test case will contain a binary string of length **N.** If the **i^{th}** bit of this string is "1", then the **i^{th}** option is correct and if the **i^{th}** bit of this string is "0", then the **i^{th}** option is incorrect. It is guaranteed that there is at least one correct option.

Constraints

- **1 ≤ T ≤ 100**
- **1 ≤ N ≤ 5**
- **1 ≤ Length of the binary string in an option ≤ 8**
- **0 < X, Y**
- **(X+Y) < 1**
- **X and Y will have at most two digits after the decimal point**

Output

For each case, print the case number and the answer in a single line. Please see the sample for details. Your answer will be considered correct if its absolute error doesn't exceed **10^{-6}.**

Formally: If your answer is a, and the answer of the jury is b, the checker program will consider your answer to be correct, if $|a - b| \leq 10^{-6}$.

Sample Input	Output for Sample Input
5 2 0.3 0.45 1 0 10 2 0.57 0.25 10 0 10 2 0.57 0.25 0 10 01 5 0.74 0.16 10 00001 10111011 11 0101000 11100 5 0.46 0.14 00111111 11010000 01101110 10011000 10011010 10111	Case 1: 0.59999999999999997 Case 2: 0.26451612903225806453 Case 3: 0.00000000000001000000 Case 4: 0.90347426984368913979 Case 5: 0.72854717043475962230

Problem F

Fair Set

Given a list of numbers, Hermione wants to choose a collection of **K** numbers with minimum unfairness. Unfairness of a collection **A** is defined as $\sum_{i=1}^{|A|} \sum_{j=i+1}^{|A|} (A_i - A_j)^2$. You need to find out the value of the minimum unfairness and also the number of such collections with minimum unfairness. A collection **S** is different from another collection **T** if there exists at least one number whose frequency is different in **S** and **T**.

Input

The first line of input contains a single integer, **T (T ≤ 500)**. Then **T** test cases follow. Each of these **T** test cases starts with two integers **N (1 ≤ N ≤ 10,000) and K (1 ≤ K ≤ F).** Then **N** lines follow, where each line contains two integers, **v (1 ≤ v ≤ 10,000)** and **f (1 ≤ f ≤ 100),** which indicates the number **v** is present **f** times in the list of numbers. **F** is the summation of all **f**, i.e. the size of the list of numbers. The sum of **N** for all cases is less than **2,000,000**.

Output

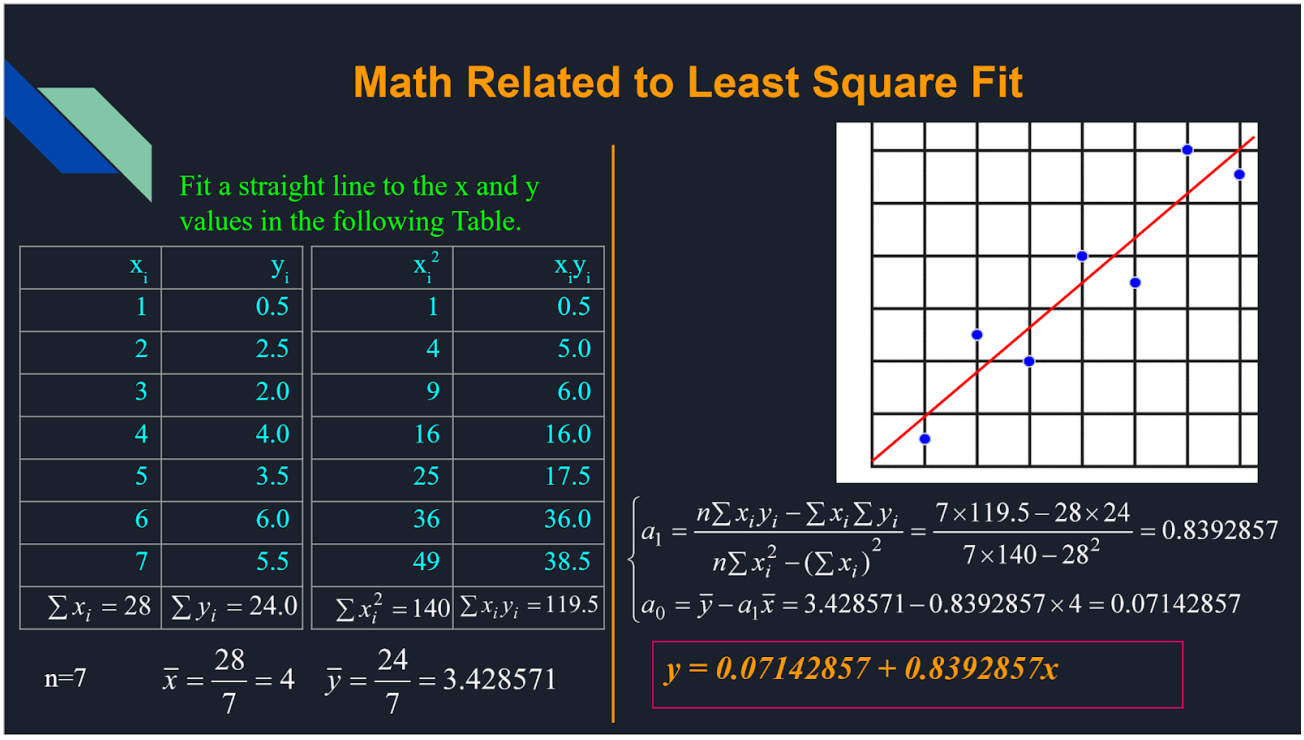
For each of the cases, output "**Case <x>: <y> <z>**" in a separate line, where **x** is the case number, **y** is the minimum possible unfairness of a collection of **K** numbers, and **z** is the number of such collections.

Sample Input	Output for Sample Input
2 5 3 3 1 5 1 6 1 1 1 9 1 4 2 234 2 1 1 5 1 3 1	Case 1: 14 1 Case 2: 0 1

Problem G

Creating Assignment

Tomisu teaches a course titled “Numerical Analysis” in Southeast University of Bangladesh. During Covid-19 pandemic he developed a practice to take online exams of students. He has given math to find the best fit line using least square method in linear regression analysis. The slides that he uses to teach in the class goes something like this:



x_i	y_i	x_i^2	$x_i y_i$
1	0.5	1	0.5
2	2.5	4	5.0
3	2.0	9	6.0
4	4.0	16	16.0
5	3.5	25	17.5
6	6.0	36	36.0
7	5.5	49	38.5
$\sum x_i = 28$	$\sum y_i = 24.0$	$\sum x_i^2 = 140$	$\sum x_i y_i = 119.5$

As it is an online exam, he gives different maths to different students. But after sending different PDF questions to all students he realized that he has made a mistake: He told his students that he will give all maths in such a way so that the regression line goes through the origin (0,0) (the value of a_0 is always zero). In this way the students will be able to catch calculation errors in almost all cases. But the value of a_1 will be different as points given in each math are different (So a_1 is the significant answer). But somehow he forgot to ensure that the value of a_0 is always zero. So for now for all math he has to add one point that will make the regression line go through the origin but the slope will be unchanged (So that if someone solves the math before getting the correction, his significant answer does not change).

Input

The input file contains at most **1000** sets of inputs.

Each set starts with an integer **N (3 ≤ N ≤ 1000)** which denotes the number of points. Each of the next **N** lines contain two floating-point numbers which denote the Cartesian coordinate of a point. You can assume that all

the coordinate values in the input will be non-negative and will not exceed **1000**. **You can also assume that the regression line for these given points will not go through the origin.**

Input is terminated by a single zero.

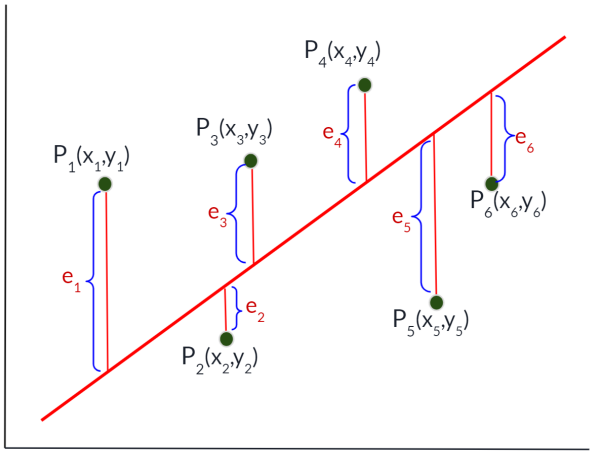
Output

For each case of input produce one line of output. This line contains the case number that is followed by two floating-point numbers. These two numbers denote the Cartesian coordinate of the newly added point that will ensure that the regression line passes through the origin. All these numbers should be rounded to exactly three digits after the decimal point. **You can assume that the inputs will be such that small changes in output values will not change the printed value. Look at the output for sample input for details.**

Sample Input	Output for Sample Input
3 1.0000 1.0000 2.0000 4.0000 5.0000 8.0000 7 1.0000 0.5000 2.0000 2.5000 3.0000 2.0000 4.0000 4.0000 5.0000 3.5000 6.0000 6.0000 7.0000 5.5000 0	Case 1: 2.667 4.641 Case 2: 4.000 2.857

Linear Regression (Least Square Method):

We will try to cover some basics of least-square regression here using the figure on the right just in case it helps. There are six points $P_1(x_1, y_1)$, $P_2(x_2, y_2)$, $P_3(x_3, y_3)$, $P_4(x_4, y_4)$, $P_5(x_5, y_5)$ and $P_6(x_6, y_6)$ on the right and suppose the long red line from left to right is a candidate for the regression line found using least square fit. The error for any point is the distance of it from the candidate regression line along the y-axis. So for points P_1, P_2, P_3, P_4, P_5 and P_6 the errors are e_1, e_2, e_3, e_4, e_5 and e_6 respectively. And if the red line is the regression line using least square linear regression then $e_1^2+e_2^2+e_3^2+e_4^2+e_5^2+e_6^2$ is minimum. If the equation of this regression line is $y = a_0 + a_1x$ then the value of a0 and a1 are found using the formulas on the right and they are derived using partial derivatives.



$$\begin{cases} a_1 = \dfrac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - \left(\sum x_i\right)^2} \\ a_0 = \bar{y} - a_1 \bar{x} \end{cases}$$

Problem H

XOR

You are given a sequence of **n** numbers **A** and **q** queries. Each query consists of two indices **l** and **r (1** $\leq$ **l** $\leq$ **r** $\leq$ **n)**. Find the value of the maximum subsegment XOR inside the subsegment **A[l...r].**

Note,
A subsegment is a contiguous sequence which is part of the original sequence.
Subsegment XOR is the XOR of all the numbers of the subsegment.

Input

First line of the input contains **T**, the number of the test cases. T test cases follow. Each case starts with 2 integers n and q. Next line contains n integers describing the sequence A. Next q lines contain 2 integers each: l and r.

Constraints

$1 \leq T \leq 10$
Sum of n over all test cases ≤ 100000
Sum of q over all test cases ≤ 100000
$0 \leq$ Each number of the input sequence $\leq 2^{20} - 1$

Output

For each case, print the case number. Then print the answer for each query in separate lines. See the sample input output for more details.

Sample Input	Output for Sample Input
1 5 2 1 0 1 0 0 1 3 4 5	Case 1: 1 0

Problem I

Hitmen

In order to save humankind, **n** potential criminals need to be taken down. The criminals are numbered from **1** to **n**. The government of Digital Utopia(DU) has announced different prize money for taking down each of them. The International Crime Prevention Corporation(ICPC) is planning to hire a bunch of hitmen to take down the criminals. There are **m** hitmen, numbered from **1** to **m**. But every hitman cannot take down every criminal. In fact, for the i^{th} criminal, there exists a range $[L_i, R_i]$ such that the hitmen whose ids are between L_i and R_i, can only take down that criminal. As expected, hiring a hitman will be costly. But once a hitman is hired, he will be obliged to take down as many criminals as ordered without any additional charge.

The chief recruiting officer of ICPC has hired a freelancer to decide which hitmen to buy and which criminals to take down so that their profit is maximized. While calculating the profit, prize money gained from taking down criminals should be added and expenses for hiring hitmen should be subtracted.

While solving the problem, the freelancer has discovered a surprising fact about the ranges of hitmen for the criminals: the given ranges never partially overlap, but one can be completely inside another range. Even after this discovery, the freelancer was unable to solve the problem. Can you help him to solve it?

Input

First line of input will contain an integer **T**, denoting the number of test cases. Each test case will start with a line containing **n** and **m.** Each of the next **n** lines will contain **3** space-separated integers denoting the prize money for taking down a criminal and the lowest and highest id of the hitmen that can take down that criminal. The last line for a test case contains **m** space-separated integers, denoting the hiring costs of the hitmen in the order of their id.

Output

For each test case, print the case number and the maximum attainable profit. Check the sample i/o for details.

Constraints

$1 \leq T \leq 100$

$1 \leq n, m \leq 50,000$

$0 \leq$ prize money for taking down a criminal $\leq 10^9$

$0 \leq$ cost of hiring a hitman $\leq 10^9$

$1 \leq L_i \leq R_i \leq m$

$2 \leq \Sigma n + \Sigma m$ over all test cases $\leq 5 \times 10^5$

For any $1 \leq i, j \leq n$ and $i \neq j$, the following condition will never be true:

$L_i < L_j \leq R_i < R_j$

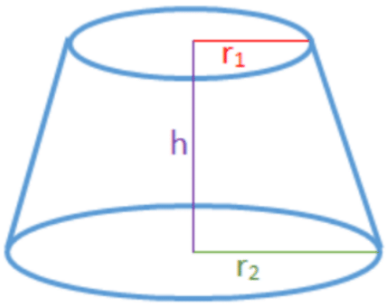
In other words, no two ranges will partially overlap.

Sample Input	Output for Sample Input
2 3 3 10 1 1 5 2 2 8 3 3 1 1 1 2 3 41 2 3 78 3 3 45 81 27	Case 1: 20 Case 2: 92

Problem J

Ant on the Conic

A teacher always gives easy exercises in the classroom but tougher in exams. Legend says, there was a student in the class who was an expert in mathematics. The student was way ahead of others in the class. Most of the others used to memorize the entire math book, while the student we are talking about never memorized, always solved math methodically. However, once before one exam the student did not understand one tough math from the book and thought: “Okay what’s the probability that this one will appear in the exam paper out of hundreds of the problems!”. And guess what exactly that one appeared. While he could not solve it, others in the class could solve it very easily!! Anyway the student is now the teacher. It’s now the payback time, but in his own way! In the class the students were given an easy problem: “Given two points on the surface of the earth, find the shortest distance between them and find the maximum latitude it will reach by a shortest path”. But in the exam, the following problem was given: “Given two points on a hill, find the shortest distance between them and also find the maximum height one can reach using a shortest path.” Let’s formalize the problem:



Given a cone-like solid shape as in the picture. The imaginary line going through the centers are perpendicular to both of the circular faces of the shape. The height is h, the radii of the two faces are r1 and r2. Two points are given on the slanted body. You need to find the shortest distance from one point to the other which wraps around the conic body at least n times (note this wrapping does not need to happen at the same height level). You will also need to find the maximum height that you can reach by one of the shortest paths. You can neither go through the circular surfaces nor go through the solid shape. You always have to be on the slanted body (it’s allowed to touch or go through the border of the slanted body.

Wrapping around the body is a vague definition. If you are familiar with the so-called winding number, you can think of it like a winding number around the vertical line through the centers of the circles. Another way to imagine it would be, to imagine yourself as a point on the vertical line through the centers. Now keep looking at the path one traverses from one point to the other. Once the path traversal finishes, count how many times you fully rotated around the vertical line. Please note, rotating 30 degree clockwise and then 390 degree counterclockwise means you rotated only 360 degree counterclockwise hence only one rotation. If the path ever comes to the vertical line (this may happen if one of the radii is zero) then you can consider the path wrapped around the vertical line (or the body) infinite times.

Input

First line of the input contains the number of tests, T. Hence T cases follow. Each case consists of 4 lines. First line describes the cone using three integers: h r_1 r_2 where h is the height, r_1 and r_2 are the radii of the upper and lower circles respectively. Second and third lines each describe a point using two integers: z theta. Here z is the height from the base and theta is an angle similar to longitude. That is, the described point is at z height from the base, and at theta angle around the vertical line through the centers of the circles (for a fixed 0 angle). The fourth line contains an integer n, the number of wrapping around the 3D body.

Constraint
$T \leq 15{,}000$
$1 \leq h \leq 10$
$0 \leq r_1, r_2 \leq 10$
$0 < r_1 + r_2$
$0 \leq z \leq h$
$0 \leq theta < 360$
$0 \leq n \leq 10$

Output

For each test case, output the case number followed by two floating-point numbers: the first one is the shortest distance and the second one is the maximum height it can reach. If the numbers are within 10^{-3} of the correct answer (by an absolute or relative) it will be accepted.

Sample Input	Output for Sample Input
2 10 0 10 0 0 5 0 0 10 0 1 0 0 9 0 10	Case 1: 7.071067811865 5.000000000000 Case 2: 11.054863183233 10.000000000000

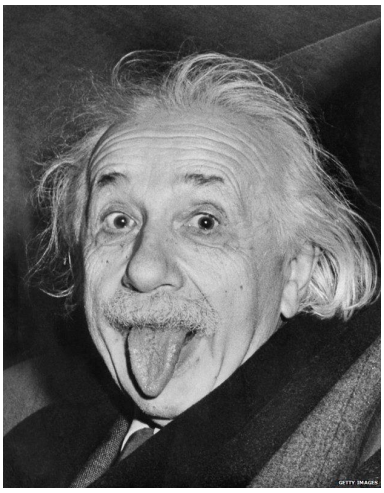
Explanation

I am a very good and student-friendly teacher, hence I gave you the second case!

Problem K

Fare Thee Well, My Friend

Fare thee well,
my friend.
Fare thee well.
Keep this torrid land
in your heart,
wherever you are going.
May you find peace
and happiness within.
- Albert Einstein



Input

There is no input for this problem.

Output

In a single line, print the number of words on the poem (excluding the name of the poet) written above.

Sample Input	Sample Output
	28

Note:
The sample output is just to demonstrate the format, it may not be the actual word count of the poem.

Problem L

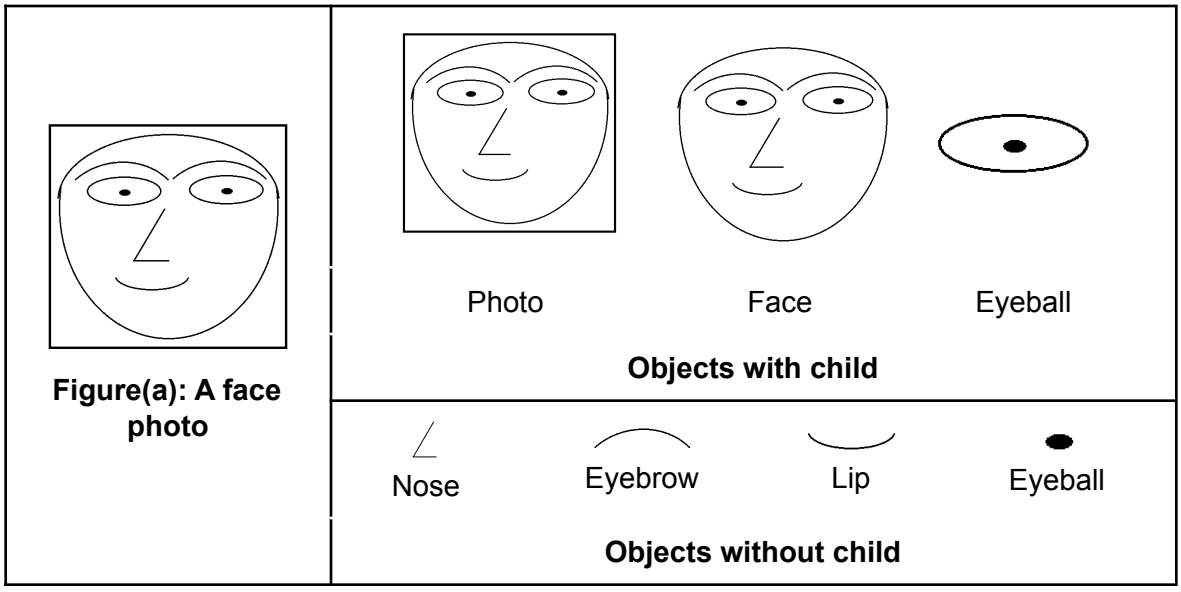
Object Detection

Detecting objects within an image is a very common research problem. An image can be represented as an $(N * M)$ matrix where each cell has a color intensity represented by character **'a'** to **'z'**, **'.'** and **'#'**. In an image, cells are considered as pixels. Pixel value **'.'** means it's blank or zero pixel (no ink). Otherwise, it's a non-zero pixel.

The following characteristics define an object within an image.

- Any two non-zero pixels sharing at least one common corner in the grid are considered to be part of the same object.
- An object can have multiple separate blank spaces inside it (see samples). Everything inside those blank spaces (if any) are part of that object.
- One or more objects may sit inside the blank space of another object. The outer object is called the parent. The immediate inner objects are called the child objects. Child objects of the same parent are called siblings.

To get a clear concept of parent, child and sibling relationships amongst objects, see the following pictorial example.



Figure(a): A face photo

The given Photo object has one child object named Face. The Face object has six child objects which are: two Eyebrow objects, two Eye objects, one Nose object and one Lip object. So, these six objects are siblings to each other. Each Eye object contains an Eyeball object as a child object. The Eyeball object of the left Eye and the Eyeball object of the right Eye are not siblings.

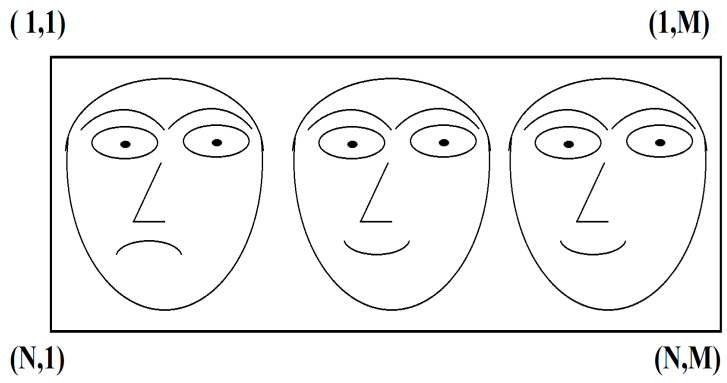
Two objects A and B are identical if all the following conditions are satisfied.

- They are siblings.
- Object A can be achieved after translating the axis of object B.
- All of the corresponding pixel **intensity** of A and B are equal, including pixels from their child objects.

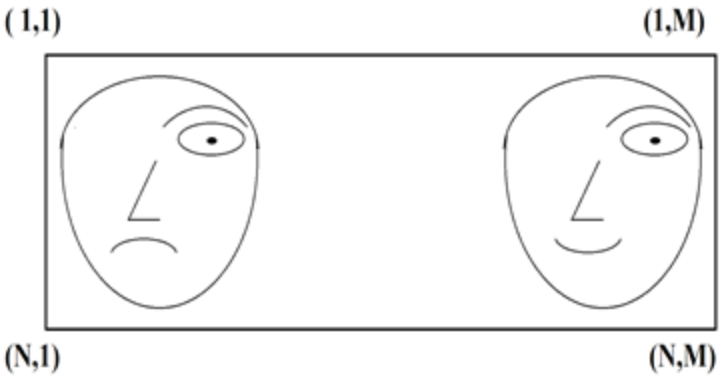
In the previous figure, the two Eyebrow objects are identical, so are the two Eye objects. As the Eyeball objects are not siblings, they are not considered to be identical.

In this problem, you have to identify all the distinct objects in an image and print their rightmost upper pixel position. The rightmost upper pixel position of an object is the cell (x,y) belonging to the object such that y is maximized, and then x is minimized. If the object is a parent, then you have to print all the child objects based on their rightmost upper pixel position. The child objects must be printed in order of decreasing y of the rightmost upper pixel. If y of the rightmost upper pixel of two objects are the same, then print the one having smaller x first.

An illustration is provided in the figure (b). In figure (b) we can see it has three faces numbered as A, B and C (from left to right). Face B and C are identical. So we output Face C by comparing the rightmost upper pixel of B and the rightmost upper pixel of C. Also, the Eyes and Eyebrows inside each face are identical.



Figure(b): Photo consists of three faces ***A, B*** and ***C*** (from ***left to right***)



Figure(c): Output image derived from above figure.

Distinct objects of Figure(b) in a tree structure according to parent-child relation:

```
|---Photo Frame
    |---Face C Border
        |---Eyebrow of C
        |---Eye of C
            |---Eyeball of C
        |---Lip of C
        |---Nose of C
    |---Face A Border
        |---Eyebrow of A
        |---Eye of A
            |---Eyeball of A
        |---Lip of A
        |---Nose of A
```

Given an image as the input, you will need to output the tree structure. See the Sample I/O for understanding.

Input

The first line of the input contains an integer **T(1≤T≤100)**, denoting the number of test cases. Each of the test cases starts with two integers, **N, M (3≤N, M≤1000)** denoting the size of the image. Next, there will be **N** lines of input. The **y-th** character of the **x-th** line represents the cell **(x,y)**. You may safely assume first row, first column, last row and last column of the input image has character **'#'** representing the photo frame of the image and no object touches the image frame. Other characters of the input are **'a'** to **'z'** and **'.'**. Character **'.'** represents a blank cell and character from **'a'** to **'z'** represents pixel intensity. **Note that, in a test file** $\sum(N \times M) \leq 2 \times 10^7$**.**

Output

For each test case, print the case number in the form of **"`Case K:`"** in a single line, where **K** is the case number. From the next line, print each of the distinct objects in the form of **"`|---(x,y)`"** in a single line where **(x,y)** stands for the rightmost upper pixel position. If the object is composed of one or more child objects, then also print them from the next line adding four leading spaces.

Note that distinct objects will be printed based on their rightmost upper pixel positions. You should not print any trailing space. See the sample I/O for more clarity.

Sample Input

```
3
20 35
###################################
#.................................#
#....oooooooooooooooooooooooooo...#
#...o........................o..#
#..o...ooooooo......ooooooo....o..#
#..o..x.......x....x.......x...o..#
#..o....ooo............ooo.....o..#
#..o...o...o..........o...o....o..#
#..o...o.x.o....l.....o.x.o....o..#
#..o...o...o....l.....o...o...o...#
#..o....ooo.....l......ooo...o....#
#...o...........l...........o.....#
#....o.......jjjj..........o......#
#.....o...o........o......o.......#
#......o...oo....oo......o........#
#.......o....oooo.......o.........#
#........o............o..........#
#.........voooooooooooov..........#
#.................................#
###################################
15 17
##################
#................#
#.aaaaa..aaaaa..#
#.a...a..a...a..#
#.a.b.a..a.c.a..#
#.a...a..a...a..#
#.aaaaa..aaaaa..#
#................#
#.abc....aaaaa..#
#........a...a..#
#.a..c...aaaaa..#
#.b..b...a...a..#
#.c..a...aaaaa..#
#................#
##################
13 24
########################
#......................#
#.bbbbbbbbb..bbbbbbbbb.#
#.b...b...b..b...b...b.#
#.b...b.a.b..b...b.a.b.#
#.b...b...b..b...b...b.#
#.bbbbbbbbb..bbbbbbbbb.#
#.b...b...b..b...b...b.#
#.b.a.b.a.b..b.a.b.a.b.#
#.b...b...b..b...b...b.#
#.bbbbbbbbb..bbbbbbbbb.#
#......................#
########################
```

Output for Sample Input

```
Case 1:
|---(1,35)
    |---(4,32)
        |---(6,28)
        |---(8,27)
            |---(9,25)
        |---(14,20)
        |---(9,17)
Case 2:
|---(1,17)
    |---(3,14)
        |---(5,12)
    |---(9,14)
    |---(3,7)
        |---(5,5)
    |---(11,6)
    |---(9,5)
    |---(11,3)
Case 3:
|---(1,24)
    |---(3,22)
        |---(5,20)
```

Explanation

In the third sample case, the ‘a’s in (5, 9), (9, 5) and (9, 9) are siblings with each other because they are inside the blank spaces of the same parent. Similarly, the ‘a’s in (5, 20), (9, 16) and (9, 20) are also siblings with each other.