A

Germ Control

Input: Standard Input
Output: Standard Output

There has been a recent epidemic outbreak which is due to the spreading of a newly discovered virus X. The virus is extremely infectious and it has become essential to find a cure, otherwise humanity will cease to exist. Professor X (who named the virus), a renowned scientist, has made a miraculous discovery and has created the Anti-X serum. With this serum, he believes he might be able to produce a cure, however the procedure won't be trivial.

The biological structure of the virus consists of consecutive cells. Each cell can be of three types: **L, R** and **#**. The professor has discovered that it takes one day for a **#** to completely dissolve. He also discovered that, if there exists an **R** cell right after an **L** cell, i.e. **LR** (not **RL**), then both of those cells would take a day to dissolve completely. However, if there is no such case, i.e. if there is no immediate **R** after an **L**, then they won't get dissolved. After dissolving, the remaining cells comes together, forming a new virus structure, which starts dissolving itself again in a similar fashion. This process continues until it's dissolved completely or it's not possible to do so any further.

To create the cure, the professor needs to create an Anti-virus by dissolving each of the cells. Using the Anti-X serum, he can transform any of the cells to either an **L**, **R** or **#** depending on how much of the serum he would use per cell. He has a list of operations which describe which cell he wants to transform and to which type. After some transformations professor might ask, if the structure starts dissolving would it dissolve completely or not. That is, he is interested in the result of the simulation, he doesn't perform dissolving literally. So to sum up, after listing a certain number of operations, he might make a query to compute the minimum amount of days it would require to dissolve the virus completely (or report it will not dissolve completely) at the current state, if at all possible. Even though he is an extraordinary scientist, this is one task which he is having difficulty with, and so he's seeking your help. The fate of humanity rests in your hands!

Input

The first line contains **T (1<=T<=5)**. **T** test cases follow. Each test case consists of a string **S (1<=|S|<=10^5),** which represents the virus and each character of the string represent the cells. **S** will contain only **L, R** and **#**. The next line contains **Q (1<=Q<=10^5)**. The next **Q** lines describes the list, i.e. the operations or queries which the professor wants to perform. Each line can be either of these two types:

- **1 X C (0<=X<|S|, C={L, R, #})**
- **2**

The first type is the listed operation where the professor would want to apply a certain amount of Anti-X to the **X'th** (0-based) cell and transform it to **C**.
The second type is the query asking what the minimum number of days, it would require to completely dissolve the virus at its current state, if at all possible.

Output

For each case, first print the case number. Follow the format provided in the sample output. After that, for each query of the second type, print the minimum number of days it would take to dissolve the virus completely. If it's not possible to dissolve it, print **"We are doomed!"** (without the quotes). See the sample input/output and explanation for more clarification. **Warning**: Use faster I/O.

Sample Input	Output for Sample Input
1 LLLRLRRRL#R#LLRR 5 2 1 1 R 2 1 6 L 2	Case 1: 3 We are doomed! 2

Explanation

1. Initially the virus looks like **LLLRLRRRL#R#LLRR.**
    a. After the first day, it will transform into **LLRRLRLR.**
    b. After the second day, it will transform into **LR.**
    c. It will fully dissolve on the third day.
2. After performing the first operation **`(1 1 R)`** the virus will look like **LRLRLRRRL#R#LLRR.**
    a. After the first day, it transform into **LRRRLRLR.**
    b. After the second day, it transform into **RR.**
    c. It won't dissolve any further, so humanity will evidently become extinct.
3. After performing the second operation **`(1 6 L)`** the virus will look like **LRLRLRLRL#R#LLRR.**
    a. After the first day, it transform into **LRLR.**
    b. It will fully dissolve on the second day.

B

Trends

Input: Standard Input
Output: Standard Output

We'll talk about trends here. Not just any trend, but trends that are not downward. These trends are an indicator of steady progress. Mathematically speaking, if we denote a non-downward trend as a sequence **S** (of length **N**), for any pair of indices **(i, j)** where **0 <= i, j < N** and **i < j**, the condition $S_i <= S_j$ would hold. We don't like special cases, so the length would be at least 2.

Let's look at couple of examples. By the definition (2, 3, 3, 5, 6) or (1, 1, 1, 1) are valid non-downward trends, whereas (1, 3, 3, 2, 1), (3, 3, 2) would not.

Okay, without further ado, we go straight down to business. ACME Corporation is the largest trend-seller in the world, not any trends but non-downward trends. But no, those are also mainstream, so how about a little special non-downward ones? So basically, some people like some particular numbers at particular positions of that sequence. One would wonder how many of valid such sequences could be made. Don't worry, to keep the answer finite we are about to put some restrictions.

Given the size of the sequence **N (2 <= N <= 100,000)**, arrays **P** and **C (|P|=|C|, 0 <= |P| <= N, 0 <= P_i < N, 1 <= C_i <= N)**, count the number of non-downward sequences **S** (as defined above), where $S_{P_i} = C_{P_i}$ and **1 <= S_i<= N**. The answer could be big, so print the answer modulo $10^9 + 7$.

Input

The first line of the input will denote the number of test cases **T (<=100)**. **T** test cases will follow. Each test case will start with two integers **N** and **|P|** denoting the desired sequence length and length of **P** and **C** arrays. **|P|** lines will follow, **i-th** line (0-based) will contain two integers P_i and C_i. You can assume that the array **P** will be increasing and **C** will be in non-decreasing order.

Output

You should print one line per test case, starting with the test case number and then the integer denoting the answer. Please refer to sample output for the format. **Warning**: Use faster I/O.

Sample Input	Output for Sample Input
3 5 2 0 1 2 4 13 4 7 1 8 5 11 11 12 13 7 6 0 2 1 3 3 4 4 5 5 6 6 7	Case 1: 12 Case 2: 28 Case 3: 2

Explanation (Fixed terms are shown in bold text)

First case:

1 1 **4** 4 4
1 1 **4** 4 5
1 1 **4** 5 5
1 2 **4** 4 4
1 2 **4** 4 5
1 2 **4** 5 5
1 3 **4** 4 4
1 3 **4** 4 5
1 3 **4** 5 5
1 4 **4** 4 4
1 4 **4** 4 5
1 4 **4** 5 5

Third case:

2 3 3 **4 5 6 7**
2 3 4 **4 5 6 7**

C	# Residential Area **Input:** Standard Input **Output:** Standard Output	

In Dhaka city, the Mayor is planning to make some major reforms. In the Mayor's plan, Dhaka city is represented as an **N** by **N** square array. Each cell of the array represents a small area of the city.

The Mayor has 10 services that needs to be provided to the citizens such as, Crime Prevention and Protection Service through Police, Fire Protection Service, Health Service, Education Service and etc. Let us represent each service using integer 1 to 10.

The Mayor is going to build various stations to provide these services (e.g, Police Station). According to the plan, there will be exactly one station in each of the small area of the city. Each area of the city, i.e, each cell of the 2D array will contain a single integer representing the station built in that area.

The blueprint for service station infrastructure is ready. The Mayor has already decided which stations are going to be built in which area. All that is left is planning for Residential Area. According to the Mayor's abstract idea, a Residential Area is a rectangular area made of one or more small cells of the 2D array such that it contains each of the services from 1 to 10 **exactly once.**

Before the Mayor finalizes the plan for Residential Area construction, he wants to know how many potential Residential Areas reside in the blueprint.

So given a 2D square array of length **N** representing Dhaka city, containing integer numbers from 1 to 10 denoting the services provided in each area, you need to find the total number of potential Residential Area. A area is different from another if it has at least one cell different from the other one.

Input

First line contains a single integer **T (T<=100)**, denoting test case. Next follows **T** test case. First line of each test case contains a single integer **N (N<=10)**, denoting the size of the 2D square array. Next **N** lines contain **N** space separated integers, denoting the elements of the 2D square array. The elements of the 2D square array are positive integers not exceeding the value of 10.

Output

For each test case print its case number, followed by number of potential Residential Area in the 2D square array. See sample I/O for more details.

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

D

Connecting To One

Input: Standard Input
Output: Standard Output

You will be given an edge weighted graph with **N** nodes numbered from 1 to **N**. If we want to drop all edges with weight lower than **C**, how many nodes will still be reachable from node 1?

Input

First line of the input is a positive integer **T (T <= 10)**. Then **T** cases follow. In each case, first line will be two integers, **N** and **E (1<= N <= 40,000, 1 <= E <= 150,000)**. Then in the next **E** lines, there will be three integers **U, V** and **W (1 <= U, V <= N, |W| <= 1,000,000,000),** which indicates there is an undirected edge between **U** and **V** with weight **W**. It will be followed by another line with an integer **Q (1<= Q <= 100,000),** then **Q** lines follow. Each of these lines will be a query. In each query there will be one integer **C (1 <= |C| <= 1,000,000,000)** for which you have to give the answer.

Output

For each test case, first line of the output will be **"Case I:"** where **I** is the test case number. Then in next **Q** lines print answer for each query.

Warning: Use faster I/O.

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

E

Anti Hash

Input: Standard Input
Output: Standard Output

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 Hash(string S, int B, int M){
      long long H = 0;
      for (int i = 0; i < S.length(); i++){
            H = (H * B + S[i] - 'a' + 1) % M;
      }
      return H;
}
```

In other words, first, the letters of the string are replaced by numbers (equivalent to their position, **'a'** gets mapped to **1**, **'b'** to **2, ...** and **'z'** to **26**). This is then considered to be a number in base **B** (rightmost number is the least significant digit), and the value of this number in base **10** 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! Limak is so 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 uses hashing have no real skill, getting 'Accepted' just implies getting lucky, nothing more!

Later that night, Limak realized that he can hack the solution if he is able to solve the following problem efficiently. Limak felt triumphant, he will teach Swistak and that stupid hash function of his a lesson! But Limak is just a little bear, he is not very good at solving problems. Please help Limak solve the following problem so that he can hack Swistak's solution. Limak will give you a string **S** of length **N**, consisting of only lowercase letters, a base **B** and a modulus **M**. Your task is to find another string **T,** satisfying all of the following constraints:

- Length of **T** is exactly **N**.
- **T** consists of only lower-case letters **(a-z)**.
- **T** and **S** are two **different strings**.
- **T** and **S** have the **same hash**, i.e. **Hash(S, B, M) = Hash(T, B, M)**.

Input

The first line contains **Q (1 ≤ Q ≤ 30)**, denoting the number of test cases. Each test case consists of two lines. The first line of each case contains three integers, **N, B,** and **M ($10^5 \le N \le 10^6$, $10^5 \le B < 2^{31}$, $10^5 \le M < 2^{31}$)**. The next line contains the string **S** of length **N**, consisting of only lower-case letters. **B ≠ M and both B and M are prime numbers. For 50% of the test cases, B, M <= 10^6**

Output

For each test case, output the string **T** in a single line. It is guaranteed that such a string will always exist for the given constraints. If there are many solutions, output any of them. **Warning**: Use faster I/O.

Sample Input	Output for Sample Input
1 38 666666667 1000000009 bbababbbbbbbaabaababaabbbabbbabababbb	hisotomeseemslikeanotoriouscoincidence

Note

The sample input contains a string of length **38** just for demonstration and clarity. There will be no such cases in the judge data, every case will strictly satisfy the constraints mentioned above.

F

Coldplay

Input: Standard Input
Output: Standard Output

Coldplay is my absolutely favorite band. I love Chris Martin. Not only as a singer, but also as a person. He is so down to earth despite being so famous. I think he is absolutely amazing.

The most amazing thing about Coldplay is that despite listening to them all the time, I often stumble on a song of them which I never heard and which turns out to be amazing as well.

Not only do they have great songs, their videos are great as well. You can feel that they put great thought behind each of them. I especially like the video of "Up&Up", "The Scientist", "Paradise" and "Yellow". Such nice concepts! Other videos are nice as well.

So you must have understood that I am a big fan of Coldplay. But what is it have to do with this problem?

Lately, I was wondering how many times I have listened to a song of Coldplay. Now there is no way to exactly calculate that number. But we can approximate. Say, in weekdays (Sunday to Thursday), due to various duties, I listen to Coldplay **P** times per day. But in weekends (Friday and Saturday) I listen to Coldplay **Q** times per day. Now I have been listening to Coldplay for exactly **Y** years now. So how many times have I listened to a Coldplay song? You can assume that a year has exactly 52 full weeks.

Try to find the number. In the meantime, I am going to listen to another Coldplay song. Cheers!

Input

Input will contain three integers, **P, Q** and **Y (0 < P, Q, Y < 10).**

Output

You need to output one integer. The number of times I have listened to Coldplay in **Y** years.

Sample Input	Output for Sample Input
2 3 2	1664

Hint: Code snippet in C/C++ to solve this problem. Just replace the line in comment.

```
#include<stdio.h>
int main(){
	int P, Q, Y, result;
	scanf("%d %d %d", &P, &Q, &Y);
	//put your result using the values of P, Q and Y
	printf("%d\n", result);
	return 0;
}
```

G	**Pattern Lock** **Input:** Standard Input **Output:** Standard Output	

More than 80 Million US Dollars was stolen from ABC Central Bank last year. The board of directors of Bank ABC investigated the whole thing but they ended up with no conclusion. “This was inevitable and it was due to the demerit of the Internet”- one of the directors had commented publicly. Being criticized by the investors after these entire unpleasant incidents, Bank ABC decided to introduce Password Protected Pattern Lock to add an extra safety mechanism.

Figure (a): Password Protected Pattern Lock

This lock has the following features:

- This is a one time use Pattern Lock. You need to set password, change the pattern and seal digits.
- There are **N** digits and one seal between each consecutive digit, so there are **N-1** seals.
- These seals are vulnerable. You can break a seal in a single unit time. Once a seal is broken, you can’t repair them.
- An unbroken seal glues two neighbor digits. So, when the seal is intact and you move one of them, the others will move in the same direction.
- Let the current pattern be {4, 3, 2, 1} and all seals are intact. If we move any of these digits in up direction, then it will be {5, 4, 3, 2} or if we move in the down direction, it will be {3, 2, 1, 0}.
- After breaking a seal, neighboring digits will become independent. Move in one will not affect other any more. For example, let the current pattern be {{4, 3}, {2, 1}}. The seal between second and third digit is broken. If we move first or second digit in up direction, new pattern will be {{5, 4}, {2, 1}}.
- In a single unit time, you can either move any digit upward or downward **once** or break an intact seal.
- When the pattern matches the password, the lock will open automatically, it does not matter if there is some unbroken seals.
- Pattern contains only digits from 0-9 and digits within a chain are also ordered from 0-9(in increasing order). Also, it’s cyclic so, 0 is connected with 1, 1 is connected with 2 … and 9 is connected with 0 again.

Now, the bank authorities will set a password, change the pattern and finally seal it. Now, the board of directors want to know the minimum amount of time it can take to unlock the lock.

Input

Input starts with an integer **T (<= 100)**, denoting the number of test cases. The first line of each test case contains the password **P** and second line of each test case contains the current situation of sealed pattern **S (1 <= |S|, |P| <= 50, |P| = |S|)**. The strings contain only the digits **0-9**. Initially all the seals in **S** are intact.

Output

For each test case print case number and the minimum amount of time it requires to unlock the lock.
Please follow the format provided in the sample input/output. **Warning**: Use faster I/O.

Sample Input	Output for Sample Input
3 1 2 111 123 121 111	Case 1: 1 Case 2: 4 Case 3: 3

Explanation

Case 1: Just Move Down once {2} and it will result in {1} in a single unit time.

Case 2:

It requires 4 unit time, there may be several ways of doing so.

Step 1: break seals between 1st and 2nd digit

Step 2: Move 2nd digit in downward once

Step 3: break seals between 2nd and 3rd digit

Step 4: Move 3rd digit in downward once.

Case 3:

Step 1: break seals between 1st and 2nd digit

Step 2: break seals between 2nd and 3rd digit

Step 3: Move Up 2nd digit once.

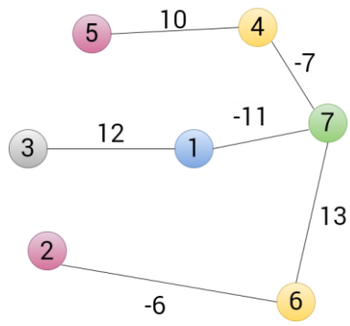
H

Maximum Subpath

Input: Standard Input
Output: Standard Output

Its ACM ICPC Dhaka Regional Preliminary 2017 and all the judges are busy with creating and solving problems. So, we couldn't find anyone to make a story for this one. As a result, this problem is going to be very much straight forward.

You will be given a tree with **N** nodes. Each node will be colored with one of the colors between 1 and **M**. Each edge of the tree has a weight.



Here is an example. You are given a tree of 7 nodes which are labeled with numbers from 1 to **N** and the nodes are colored with 5 different colors. So, **N** = 7 and **M** = 5.

Lets define a path from **U** to **V** as **(U, V)**. Two paths **(U,V)** and **(V,U)** are considered same as the edges of the tree are bi-directional. Now, we will discuss some terminology regarding our problem:

The cost of a path **(U, V)** is the summation of weights of its edges. The cost of (5, 4) is 10 in the above figure, cost of (1, 5) is -8 (-11 -7 + 10).

A path **(U1, V1)** will be called a subpath of path **(U, V)** if **U1 != V1** as well as **U1** and **V1** are both on the path **(U, V)**. As a result a path can have several subpaths. In the above figure, the subpaths of path (5, 1) are: (5, 4), (5, 7), (5, 1), (4, 7), (4, 1), (7, 1), (4, 5), (7, 5), (1, 5), (7, 4), (1, 4) and (1, 7)

Each pair of nodes in the tree has a beautiness value which is different from path cost. The beautiness of a pair of nodes **U** and **V** (where **U != V**) is equal to the maximum path cost among all the subpaths of path **(U, V)**. So, the beautiness value of path (5, 1) will be 10. Because, the subpath (5, 4) or (4, 5) has the maximum path cost among all the subpaths of (5, 1). The beautiness value of a pair **U** and **V** where **U and V are same is undefined.**

Now let's discuss what your task is. You will be given a tree. You will have to find a pair of nodes **U** and **V** such that the path **(U, V)** contains all the colors from 1 to **M** exactly once and their beautiness value is maximum. For making the problem simple, you will just have to print the maximum beautiness value of that pair.

Input

The first line of the input will be the number of test cases **T(**1 <= **T** <= 20). Each test case starts with two numbers, **N** and **M (**1 <= **N, M** <= 50,000). **N** is the number of nodes in the tree and **M** is the number of different colors the nodes are printed with. Next line contains **N** integers **col[1]**, **col[2]**, **col[3]**, … , **col[N] (**1 <= **col[i]** <= **M)**, the color of the nodes labeled with 1, 2, 3, … , **N**. Next, each of the following **N**-1 lines describes an edge with **U V W (**1 <= **U, V** <= **N**, **U != V,** -1,00,000 <= **W** <= 1,00,000). **U** and **V** are two nodes **(U != V)** connected with an edge and **W** is the weight of the edge. The summation of **N** for all the test cases in a file will not exceed 4,00,000.

Output

For each test case, you will have to print the case number and the maximum beautiness value of a pair **(U, V)** that contains all the colors from 1 to **M** exactly once between their path. If you can't find any such pair, just print -1 in the place of beautiness.

Warning: Use faster I/O.

Sample Input	**Output for Sample Input**
4 4 3 1 2 3 1 1 2 10 2 3 -10 3 4 11 3 2 1 2 2 1 2 -10 1 3 -11 3 3 1 2 2 1 2 -10 1 3 -11 2 1 1 1 1 2 -7	Case 1: 11 Case 2: -10 Case 3: -1 Case 4: -1

Explanation

Case 1:

There are two pairs, (1, 3) and (2, 4). The beautiness of (1, 3) is 10 because there is a subpath (2, 3) has a cost 10. The beautiness (2, 4) is 11 because the cost of subpath (3, 4) or (4, 3) is 11.

Case 2:

The pair (1, 2) has the maximum beautiness of -10.

Case 3:

There are no pairs containing all the three colors exactly once. So, the output is -1.

Case 4:

There is M=1, so, you will find a pair (1, 1) or (2, 2) which contains exactly one color. But according to our problem, the beautiness value of (1, 1) or (2, 2) should be undefined. So, we couldn't find such a valid pair.

I Repeated Digit Sum

Input: Standard Input
Output: Standard Output

You are given a function **f** which is defined as:

```
1. int f(n) {
2.    while (n >= 10) {
3.       n = sod(n);
4.    }
5.    return n;
6. }
```

Here, **sod(n)** indicates the sum of digits of **n**. So, **f(7689)** = 3 (7689 => 30 => 3).

Given, **a** and **b**, you need to find the value of **f(a^b)**.

Input

Input starts with an integer **T (≤ 50)**, denoting the number of test cases. Each case starts with two big integers **a b (0 ≤ a, b < $10^{50,000}$, a + b > 0).**

Output

For each case, print the case number and **f(a^b)**.

Warning: Use faster I/O.

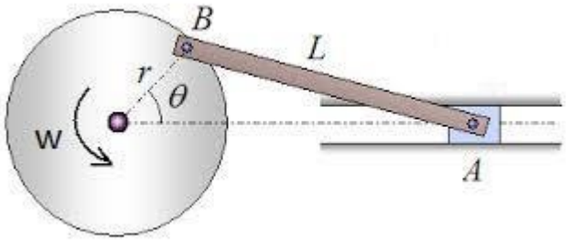
Sample Input	Output for Sample Input
4 2 5 7 2 8 1 1 123456789012345678901234 5	Case 1: 5 Case 2: 4 Case 3: 8 Case 4: 1

J

Rotation Motion

Input: Standard Input
Output: Standard Output

Have you ever looked at the gear/wheel system of rickshaw? Did you notice how the rotation of the paddle ultimately makes the rickshaw to go forward? It is always amazing to see how movement of one part causes different type of movement in another part. Take the following picture for an example.



Here rotation motion is converted to linear motion. Let, there is a disk of center **O** and radius **r**. A rigid stick of length **L** is attached at point **B** of the disk. The other end of the stick is at **A** which is pinned to a block. The block is tightly packed between two surfaces (you may consider it as cross section of an engine where the block is the piston and the two surfaces are actually a cylinder). So now when the disk rotates, the block moves horizontally. For convenience, we will consider the center **O** is at the origin of a 2D grid, the horizontal line on which **A** moves, is **x**-axis.

Given,
$\mathbf{B_0}$: Initial location of the point **B**
$\mathbf{A_0}$: Initial location of the point **A**
w: The counterclockwise rotation speed of the disk. Mathematically, in 1 second, the disk rotates by **w** radian in counter clockwise direction
t: Time of rotation

Find the final position of **A** after **t** seconds.

Input

First line of the input contains number of test cases, **T (T <= 1,000)**. Hence follow **T** cases.
For each case, there is one line of input. The line contains 5 integers: $\mathbf{B_0x}$ $\mathbf{B_0y}$ $\mathbf{A_0x}$ **w t**. You may assume that, the length of the stick is larger than the radius of the disk. The absolute value of all the inputs (including **t**) will be no greater than 10,000. Negative **t** means **t** second before now and negative **w** means rotation speed of **w** in the opposite (clockwise) direction.

To solve this problem, do not complicate your imagination with "how this is feasible in 3D" (i.e. how the stick goes through the cylinder, what happens when the block intersects the disk etc).

Output

For each case, print the case number followed by the **x** coordinate of the final location of **A**. Your solution will be accepted if your output is within 1e-6 of the correct answer.

Warning: Use faster I/O.

Sample Input	Output for Sample Input
2 3 4 20 1 1 2 3 100 -1 1	Case 1: 15.078963 Case 2: 101.650906