A

Problem A

Input: Standard Input
Output: Standard Output

Kings in 2D Grid

There will be a 2-Dimensional RxC grid, where R is the row number and C is the column number. The rows of this grid are numbered from top to bottom starting with index 1, and columns are numbered from left to right starting with index 1. In this grid, two kings are placed where one king is white and the other king is black. The white king is placed on (R1, C1) coordinate and the black king placed on (R2, C2) coordinate. They are initially placed on two different cells of the grid in such a way that they are not adjacent to each other (two cells on the grid are adjacent if they share a side or a corner). Alice plays with white king and Bob plays with black king.

In each turn, one can move his/her king to any adjacent cell from the current cell, given that the new position of the king can not be adjacent to the other king. If there is no valid move for the player in turn, the game ends immediately. Otherwise, if the game continues for 10^9 moves in total (counting both Alice and Bob's move), the game is called to an end.

Alice and Bob play alternatively. Alice will move first. Alice will always try to minimize the different cell number visited by the black king of Bob. Bob will always try to maximize the different cell number visited by his black king. Both Alice and Bob will play optimally.
You have to calculate the different number of cell Bob's black king can visit before the game ends.

Input

Input starts with test case number **T** **(1 ≤ T ≤ 2500)**. Each of the following **T** Lines contains six space separated integers **R, C, R1, C1, R2, C2** where **1 ≤ R, C ≤ 14, R × C ≤ 14;** and **maximum of R and C will be greater than 2;** and **1 ≤ R1, R2 ≤ R, 1 ≤ C1, C2 ≤ C. (R1, C1)** and **(R2, C2)** will not be adjacent to each other.

Output

For each test case, output a single line in the format "**Case X: Y**" without the quotes. Here, **X** is the case number and **Y** is the maximum number of distinct cells visited by the black king.

Sample Input	Output for Sample Input
2 1 7 1 4 1 1 2 5 2 1 1 5	Case 1: 1 Case 2: 4

Easy

B Problem B

Mysterious LCM

Input: Standard Input
Output: Standard Output

You will be given an array **A** of **N** integers and another integer **X**. You have to find the minimum length of a subsequence from **A** such that, the LCM of all the elements in that subsequence is exactly **X**.

Input

The first line of the input contains a single integer **T**, which denotes the number of test cases. The first line of each test cases contains two integers **N** and **X**. The second line of a test case contains **N** space separated integers of the array **A**.

Constraints

- $1 \le T \le 50$
- $1 \le N \le 50000$
- $1 \le A_i, X \le 10^{18}$

The sum of **N** over all the test cases in an input file $\le$ **10^6**

Output

For each test case, output a single line in the format "**Case T: D**" without the quotes. Here, **T** is the case number and **D** is the desired answer denoting the minimum length of a valid subsequence whose LCM is exactly **X**. If there is no solution, then print **-1** instead.

Sample Input	Output for Sample Input
3 8 60 2 9 12 15 16 21 25 40 8 90 2 36 44 49 50 63 64 81 8 138600 40 2200 45 216 175 735 15 36	Case 1: 2 Case 2: -1 Case 3: 3

Explanation

In our first sample case, the LCM of the elements in the valid subsequence (12, 15) is 60. There is no way to select a valid subsequence of length less than 2.

Warning: Dataset is huge. Please use faster I/O methods.

C

Problem C

Input: Standard Input
Output: Standard Output

Swipe Your Time Away

Bangladesh Association of Problematic Society (BAPS) is about to release their new mobile game SYTA (Swipe Your Time Away). According to the opinion of several game specialists and psychologists, this is going to be one of the greatest addictive game in the history of mankind. This game will ruin the civil society. Students will drop out of schools, family bonding and relationships will be destroyed, people will only swipe their time away just playing SYTA day and night.

SYTA is a board game played on **N** by **M** board (**N** rows and **M** columns). The left uppermost cell is numbered as $(1,1)$ and the right lowermost cell is numbered as (N,M). Each cell contains exactly one ball. The colors of these balls are of different types. There are five colors and they are: 1. Red 2.Green 3.Blue 4.Yellow and 5. Purple. The goal of the game is to determine the largest **Cross** shape with the same color within the board. Formally, cell (x,y) is a center of a **Cross** shape if it has the following properties:

1. In x^{th} row all the balls from column y_0 to y_1 are of same color
2. In y^{th} column all the balls from row x_0 to x_1 are of same color
3. $x_0 < x < x_1$
4. $y_0 < y < y_1$
5. The size of that **Cross** shape is, $D = (x_1 - x_0) + (y_1 - y_0) + 1$

We are looking for someone who can crack down this devastating game. All you need to do is to find out the size of the largest Cross shape in shortest period of time. If you can do so, world will be saved, and in return we will gift you one colorful balloon.

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

In the first test case, there are 2 Cross shapes.

cell (2,2) and (3,5) are the center of these Cross and the size of these cross shapes are 7 and 8 respectively.

So, the largest size is 8.

Input

Input starts with an integer **T** denoting the number of test cases. Each of the test cases starts with two integer **N** and **M** denoting the number of rows and columns in the board. Next, there will be **N** lines of inputs each containing **M** space separated integers. The j^{th} integer of the i^{th} line is C_{ij}, representing the color of ball in the cell numbered (i,j)

Output

For each test case, output a single line in the format "**Case T: D**" without the quotes. Here, **T** is the case number and **D** is the desired answer.

Constraints

- **$1 \le T \le 45$**
- **$3 \le N \le 1000$, $3 \le M \le 1000$** (In more than **90%** cases **$3 \le N, M \le 200$**)
- **$1 \le C_{ij} \le 5$**

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

Warning: Dataset is huge. Please use faster I/O methods.

D

Problem D

Input: Standard Input
Output: Standard Output

DarkCity, CrimsonCity of FlightLand

Flightland is a wonderful country. There are **N** cities in Flightland. A city's position can be determined by 2D coordinate system. You can go from any city to another city using an airplane.

Say an airplane is departed from **City P** and landed on **City Q**. **City P**'s coordinate is **(Xp, Yp)** and **City Q**'s coordinate is **(Xq, Yq)**. Here is some calculation for air travel costs for a single flight:

- The airplane chooses a path from **City P** to **City Q** according to your choice. But remember, you have to choose a path such that the length of the path is always an integer. It's for the safety of the passengers. You don't have to worry much about that. And you can ignore the altitude of the airplane. So, you can think this path to be a straight line, polyline or curve on a 2D plane. Say the length of the path is **L**.

  Example: Say, the airplane took-off from **(0, 0)** and landed on **(5, 5)**. You can not choose the straight line between these points, as the length of the path will not be an integer. Rather you can choose a path from **(0, 0)** to **(1, 0)** to **(1, 2)** to **(5, 5)** with straight lines. In that case, the length of the path will be **8**, which is an integer. But these turning points of the airplane can also be non-integer coordinates and also the paths can be arbitrary curves too. Like you can choose an arc from **(0, 0)** to **(5, 5)** such that the length of the arc is **11**, **10**, **9** etc. Simply, you can choose any route plan as your choice. This can be a straight line, polyline, curve etc. But the length of the path must be an integer.
- The airplane has a mass of **A**, including the mass of all passenger, the luggages and some other stuffs. It's basically the mass of the whole plane except its fuel's mass.
- The airplane starts with **F** amount of fuel, which is a positive real number. You can choose the amount of fuel for the journey. So, the total mass of the airplane is **T = A + F**. It is obvious that the airplane needs sufficient amount of fuel to complete the flight. Otherwise the airplane will crash and you are never going to meet your destination.
- For a unit distance of flight, Airplane consumes **TotalMass / D** amount of fuel. So, you can see the plane is constantly(in this case, per unit distance) losing its fuel thus its total mass. Here, **D** is the consumption rate which is constant.
- Price of a unit amount of fuel is **C**.
- There is some extra base cost for take-off, landing, airport fee etc. Say this extra cost is **B**.
- So, the total cost for a single flight is **F × C + B**.
- Size of airplane and airports are negligible. You can think them as points.
- You can not do a mid-air refueling. That means you start your flight with **F** amount of fuel and you have to finish your flight using that fuel only.

- Each flight is independent, and they will be operated with a brand new airplane. So, if you have some fuel remaining on your last flight, you can not use it in this flight. Besides, the initial amount of fuel used for each flight can be different too.
- You can take flights only from one city to another, and every city has an airport. Note that, airports are situated only in the cities.

Here is an example of fuel consumption. Say, **L = 5, A = 100, D = 100**.
If you choose **F = 10**, then here is the fuel consumption history:

When airplane travels **0** unit distance, means when the airplane will take-off, **TotalMass = 100 + 10 = 110**. So, in the next unit distance airplane will need **110 / 100 = 1.1** unit of fuel.
When airplane travels **1** unit distance, **TotalMass = 100 + 8.9 = 108.9**. So, in the next unit distance airplane will need **108.9 / 100 = 1.089** unit of fuel.
When airplane travels **2** unit distance, **TotalMass = 100 + 7.811 = 107.811**. So, in the next unit distance airplane will need **107.811 / 100 = 1.07811** unit of fuel.
When airplane travels **3** unit distance, **TotalMass = 100 + 6.73289 = 106.73289**. So, in the next unit distance airplane will need **106.73289 / 100 = 1.0673289** unit of fuel.
When airplane travels **4** unit distance, **TotalMass = 100 + 5.6655611 = 105.6655611**. So, in the next unit distance airplane will need **105.6655611 / 100 = 1.056655611** unit of fuel.
When airplane travels **5** unit distance, The airplane will land and **4.608905489** unit of fuel will remain the airplane.

You live In DarkCity and your dream girl/boy lives in CrimsonCity. You want to go from DarkCity to Crimson city. You have to bear the full cost for each flight. Can you calculate the minimum cost to visit your dream girl/boy?

Input

Input starts with an integer, **TC**, denoting number of test cases. For each test case, the first line contains 5 integers **N, A, B, C, D** denoting number of cities in Flightland, mass of airplane, extra base cost per flight, fuel cost, consumption rate. The next **N** lines contain 2 integers denoting the coordinates for each city. First city is always DarkCity and the last city is always CrimsonCity.

Output

For each test case print a single line in this format **Case X: Y.** Here **X** is an integer that denotes the test case number and **Y** is a real number that denotes the minimum cost.
Your answer will be accepted if its relative or absolute error does not exceed 10^{-9}.
Mathematically, if your answer is A and the jury's answer is B, then your answer is accepted if and only if $\frac{|A-B|}{max(1,|B|)} \le 10^{-9}$.

Constraints

- $1 \le TC \le 100$
- $2 \le N \le 200$
- $10^5 \le A \le 10^6$
- $1 \le B \le 10^7$
- $1 \le C \le 100$
- $10^9 \le D \le 2 \times 10^9$
- $0 \le$ coordinates $\le 10^9$

Sample Input	Output for Sample Input
2 2 787379 570 57 1071473853 0 0 100000 100000 3 695972 544 5 1044478508 0 0 100000 100000 500000 500000	Case 1: 6494.104906270 Case 2: 2900.646378366

E

Problem E

Input: Standard Input
Output: Standard Output

LICT IUPC 2019

Consecutive Letters

You are given a string S containing only uppercase English letters. There are Q queries. Each query can be of two types,

- **1 i**: Find the maximum size of the segment ***[b, e]*** where **$0 \le b \le i \le e < |S|$** and substring ***S[b...e]*** contains only the letter ***S[i]***. A Substring is a contiguous sequences of characters in a string.
- **2 i**: Change the character in index ***i*** with the character '**#**'.

For both type of queries, **S[i]** will not contain the character '#'.The characters of the string are indexed from 0.

Input

The first line contains number of test cases **T ($1 \le T \le 25$)**.
For each test cases, the first line contains the string **S ($1 \le |S| \le 200000$)**. The 2nd line contains number of queries **Q ($1 \le Q \le 100000$)**. Each of the next **Q** lines contains one query in the format mentioned in the problem statement.

Output

For each test case, first print the test case number and output of every query of type 1 in a single line.

Sample Input	Output for Sample Input
2 AABBBCCCC 5 1 0 2 1 1 0 2 2 1 3 XXYYY 3 1 3 2 3 1 2	Case 1: 2 1 2 Case 2: 3 1

Warning: The input file is huge, please use fast I/O.

F

Problem F

Input: Standard Input
Output: Standard Output

Palindromadness

You'll be given a string **S** of length **N**, containing lowercase English letters only. Let's define a function **f(x)** over this string.

f(x) denotes the number of quadruple of indices **(i, j, p, q)** where,

1. **A = S[i...j]** and **B = S[p...q]** are substrings of string **S** where $1 \le i, j, p, q \le N$, $i \le j$ and $p \le q$. That is, **A** and **B** can partially overlap, or contain each other, or be completely separate or be the same substrings in string **S**.
2. Both **A** and **B** are palindromes.
3. **A** is a substring of **B**
4. Length of **A** is equal to **x**.

Substrings are contiguous sequences of characters in a string. Palindromes are strings, that read the same forwards and backwards.

You'll also be given a base and a mod value. You'll have to print $(\sum_{i=1}^{n} f(i) \cdot base^{n-i}) \% mod$.

Input

First line of input will contain an integer **T** ($1 \le T \le 10^5$), denoting the number of test cases. First line of each test case will contain three integers **N, base** ($1 \le base \le 10^9$), **mod** ($1 \le mod \le 2 \times 10^9$). Following line will contain a string of length **N**, containing lowercase English letters only. Summation of **N** over all test cases will not be greater than 10^6.

Output

For each test case, first print the case number, starting with 1, followed by the answer for that case. Check the sample I/O for more details.

Sample Input	Output for Sample Input
5 13 100 999 welcometomist 7 1000 1000000000 topspot 11 23167 21898192 abbabobaxab 21 123456 123456789 amanaplanacanalpanama 32 72817 728917897 bobxyxthehtxyxbuildercantfixyxit	Case 1: 21 Case 2: 2000001 Case 3: 4104229 Case 4: 85459969 Case 5: 721428038

Explanation

The f(x) function(1 based) for the first three cases are given below.

Case 1: { 21, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 }

Case 2: { 28, 0, 3, 0, 2, 0, 1 }

Case 3: { 97, 2, 5, 1, 2, 0, 0, 0, 0, 0, 0 }

G

Problem G

Input: Standard Input
Output: Standard Output

Decode The Alien Message

Professor Neo has established contact with an alien civilization! His team have been intercepting encoded messages since then from the alien colony. Messages contained chunk of random integers. After getting no clue for weeks finally they were able to decode it.

The random integers are to be decoded according to their parity. An even integer correspond to 1, and an odd integer corresponds to 0. So, every chunk of integers can be converted to a string containing 0 and 1. As there are lots of messages, you came into play!

You have to write a code that decodes the messages. There are two points you have to be careful about:

1. There will be no chunk where all of the integers are odd. That means, in a chunk there will be at least one even integer. For example, there will be no chunk like: 5 5 5.
2. After the message is decoded, you have to skip the leading zeros before printing. For example, consider a chunk of three integers: 5 5 4. After decoding you will get 001, but you need to print 1.

Input

The first line will contain an integer **T ($T \le 100$)**.
Each test case will contain an integer **N ($1 \le N \le 50$)** in the first line, and the second line will contain **N** space separated positive integers within the range **[1, 1000]**.

Output

For each test case, output a single line in the format "**Case T: D**" without the quotes. Here, **T** is the case number and **D** is the desired decoded message.

Sample Input	Output for Sample Input
3 1 4 3 4 4 5 3 5 5 4	Case 1: 1 Case 2: 110 Case 3: 1

H

Problem H

Input: Standard Input
Output: Standard Output

Triangle Inside Rectangle Inside Pentagon

So last day I was teaching geometry to my cousin. He is interested in learning geometry but could not keep his attention. So I tried to make things interesting by asking him to give me any geometric challenges he can think of.

All was going good, but then I started to teach him regular convex polygon. A regular convex polygon is a polygon that is equiangular (all angles are equal in measure), equilateral (all sides have the same length) and all interior angles are strictly less than 180 degrees. Like equilateral triangle, square and so on. But then he gave me a challenge:

- *Brother, can you fit a equilateral triangle in a square*
- Ok, here it is.
- *Ok, now fit the square in a regular convex pentagon.*
- Umm, ok, this might take some time!
- *Then fit that in a regular convex hexagon.*
- Stop!
- *Then fit that in a regular convex heptagon.*
- Please Stop!!
- *Also make them as small as possible*
- ಠ_ಠ

You will be given the length of side of the equilateral triangle **s** and **N**. You have to print the area of minimum regular convex **N**-gon, which will contain a regular convex **N-1**-gon(if **N** > 3), which will contain a regular convex **N-2**-gon(if **N** > 4), which will contain a regular convex **N-3**-gon(if **N** > 5) and so on. Finally, a square(**4**-gon)(if **N** > 3) will contain the triangle.

Input

The first line contains an integer **T** ($T \le 10^5$), denoting the number of test cases. Each test case contains two integers **N** ($3 \le N \le 10^5$) and **s** ($1 \le s \le 10^3$).

Output

For each test case, print a single line in this format **Case I: A.** Here **I** denotes the test case number and **A** denotes the minimum area. Your answer will be accepted if its relative or absolute error does not exceed 10^{-4}. Mathematically, if your answer is A and the jury's answer is B, then your answer is accepted if and only if $\frac{|A-B|}{max(1,|B|)} \le 10^{-4}$.

Sample Input	Output for Sample Input
3 3 1 5 4 10 1	Case 1: 0.433012702 Case 2: 22.542684260 Case 3: 3.484763487

I	Problem I	Input: Standard Input Output: Standard Output

Fibonacci Power Sum

The fibonacci series is defined as below:

fib(0) = 0, fib(1) = 1

fib(n) = fib(n-1) + fib(n-2) for n > 1

Given three integers **N**, **C** and **K**, find the summation of the following series:

$\mathbf{fib(0)^K + fib(C)^K + fib(2*C)^K + fib(3*C)^K + \dots + fib(N*C)^K}$

Since the answer can be huge, output it modulo **1000000007** **(10^9+7)**.

Input

The first line contains an integer **T**, denoting the number of test cases. Each test case contains three space separated integers in the order: **N**, **C** and **K**.

Constraints

- $1 \le T \le 100$
- $0 \le N \le 10^{15}$
- $1 \le C, K \le 10$

Output

For each test case, output a single line in the format "**Case T: D**" without the quotes. Here, **T** is the case number and **D** is the desired answer denoting the sum of the series.

Sample Input	Output for Sample Input
5 10 1 1 5 2 2 3 3 4 1000000007 7 9 996969696969696 9 6	Case 1: 143 Case 2: 3540 Case 3: 1340448 Case 4: 880410497 Case 5: 689328397