Problem A

Almost Perfect Prime Factor!

A perfect prime factor is a prime factor that exists in all the elements of an array. An almost perfect factor is a prime factor that exists at least once in an element of the array, but not in all of the elements of the array.

You are given an array **A** of **N** positive integers and an integer **K**. You have to find the longest subarray of **A** that has exactly **K** almost prime factors and no perfect prime factor in that subarray.

Input

The first line will contain a single integer **T**. Each test case starts with two integers **N** and **K**. The following line will contain **N** space-separated integers $\mathbf{A_1}$, $\mathbf{A_2}$, **...**, $\mathbf{A_N}$ denoting the elements of array **A**.

Output

For each case, print one line with "**Case <x>: <y>**", where **x** is the case number, and **y** is the length of the longest subarray of **A** that has exactly **K** almost prime factors and no perfect prime factor.

Constraints

- $1 \le T \le 10^4$
- $1 \le N \le 10^5$
- $1 \le K \le N$
- $1 \le A_i \le 10^6$
- The sum of N over all test cases doesn't exceed 10^5

Sample Input

```
2
5 3
3 5 12 15 21
3 1
3 3 3
```

Output for Sample Input

```
Case 1: 4
Case 2: 0
```

Problem B

Bob and Numbers

Bob loves number theory. Up until now, he has solved countless number theory problems on several online platforms. One of those platforms is ForceCode. He especially likes to solve problems on ForceCode due its tremendous collections of number theory challenges.

One day while surfing through ForceCode he stumbled upon a unique number theory challenge which involves finding the modulo of some numbers by **A** and **A - 1**. As he dived into solving this intriguing problem, an idea struck him. What if he could create a new problem inspired by this mathematical concept? A problem that would not only test programming skills but also offer a unique twist to mathematics.

After spending several hours, he came up with the following problem. Given two positive integers **A** and **N,** you need to find:

$$\sum_{i=1}^{N} K(i) \text{ where } K(i) = |i \,\%\, A - i \,\%\, (A-1)|$$

Here, **a % b** is the remainder after dividing **a** by **b**. Moreover, **|a|** is the absolute value of **a.**

Even though Bob solved countless problems that involve mathematics and number theory, he finds this problem a bit challenging for his level of experience. Hence, he needs your help. Would you be able to help Bob solve this problem?

Input

Input will have multiple test cases. The first line of the input contains an integer, **T** ($1 \le T \le 10^5$), representing the number of test cases. Each of the next following **T** lines will have two positive integers **A** ($2 \le A \le 10^5$) and **N** ($1 \le N \le 10^{12}$).

Output

For each case, print the case number followed by the answer. Please, see the sample for details.

Sample Input

```
3
2 2
3 6
4 24
```

Output for Sample Input

```
Case 1: 1
Case 2: 5
Case 3: 28
```

Problem C

Counting Ones II

Bangladesh Association of Problem Setters

Given a **N x N** binary matrix and a positive integer **K**, you are tasked with performing scaling operations on this matrix by a factor **K** and counting '**1**'s within some specified ranges.

The scaling of the binary matrix by factor **K** means replacing each element with a **K x K** matrix filled with the same element. See the example below for better understanding.

Once the scaling is done, the resulting matrix is then merged into a binary string. More formally, the final string will be $S = r_1 + r_2 + r_3 + \dots \dots + r_{NK}$ where r_i is the ith row of the scaled matrix. The length of the final string is **|S| = $N^2 K^2$.**

Example of scaling:
For K = 2

Matrix	Matrix after scaling
0	00 00
01 10	0011 0011 1100 1100
010 101 010	001100 001100 110011 110011 001100 001100

You will then be given some ranges on the merged binary string. You need to answer how many 1's are there within those ranges.

Input

Input will have multiple test cases. The first line of the input contains an integer, **T (1 ≤ T ≤ 100)**, representing the number of test cases.

For each test case:

- The first line contains two integers, **N ($1 \le N \le 1000$)** and **K ($1 \le K \le 10^6$)** representing the size of the binary matrix and the scaling factor, respectively.
- The next **N** lines contain binary strings of length **N**, representing the rows of the binary matrix. Each string consists of '**0**' and '**1**'.
- The next line contains an integer, **Q ($1 \le Q \le 10^5$)**, representing the number of queries for this test case.
- Each of the following **Q** lines contains two integers, **L & R ($0 \le L \le R < N^2K^2$)**, representing the range of the final binary string.

You can safely assume that the sum of Q over all test cases will not exceed 10^5 and the sum of N over all test cases will not exceed 1000. Moreover, it is recommended to use a faster I/O method as the dataset can be huge.

Output

For each case, print the case number. Then print the answer to the queries associated with that case in separate lines. Please, see the sample I/O for details.

Sample Input	Output for Sample Input

```
2
1 1
0
1
0 0
2 2
01
10
5
0 4
3 11
6 9
4 5
0 15
```
	```
	Case 1:
	0
	Case 2:
	2
	5
	4
	0
	8
	```

Problem D

Sequence

You are given two strings **P** and **Q** having lower case english letters. A sequence $\{x_1, x_2, x_3 \ldots x_k\}$ is interesting if there is **any** sequence $\{y_1, y_2, y_3, \ldots y_k\}$ such that,

- for all $1 \le i \le k$, $P[1..x_i]$ appears as a substring of **Q** ending at y_i index. Note that, $P[1..x_i]$ is a string formed by taking the first x_i characters of **P**.
- $y_i < y_{i+1}$ for all $1 \le i \le k - 1$. Note that this condition is only considered for $k \ge 2$, for $k = 1$ first condition is sufficient.

How many distinct interesting non-empty sequences $\{x_1, x_2, x_3 \ldots x_k\}$ can you make?

Input

First line will contain a single integer **T** representing the test cases. For each test case there will be 2 lines. The first line will represent string **P** and second line will represent string **Q**.

Constraints

- $1 \le T \le 2 \times 10^4$
- $1 \le |P|, |Q| \le 3 \times 10^5$
- sum of $|P|$ overall test cases $\le 3 \times 10^5$
- sum of $|Q|$ overall test cases $\le 3 \times 10^5$

Output

For each test case print a single line representing the answer modulo **998244353**.

Sample Input	Output for Sample Input
2 aa aa abc abd	4 3

Explanation

For the first test case distinct interesting sequences are {1}, {1, 1}, {2}, {1, 2}.

- For sequence {1}, end indices are, {1} or {2}.
- For sequence {1, 1} end indices are {1, 2}.
- for sequence {2}, end indices are {1}.
- for sequence {1, 2} end indices are {1, 2}.

Problem E

Sub-Array

An array is Beautiful if all the elements in that array are identical.

You are given an array ***A*** consists of **N** integers. Now for the given array, you have to find the number of non-empty subarrays which are not Beautiful.

A subarray is a contiguous part of the array.

Example:

A = [1, 0, 1, 0] where N = 4

Possible six subarrays which aren't beautiful.

[1,0] , [1,0,1] , [1,0,1,0] , [0,1] , [0,1,0] , [1,0]

Input

Input starts with an integer T denoting the number of test cases. Each case starts with an integer N denoting the number of integers in the array. Next line will contain N integers of array A, separated by spaces.

Output

For each case, print the case number and the result in a single line.

Constraints:

- **$1 \le T \le 10$**
- **$1 \le N \le 10^5$**
- **$0 \le A[i] \le 2^{31} - 1$**

Sample Input

```
2
4
1 0 1 0
7
1 1 2 3 3 4 7
```

Output for Sample Input

```
Case 1: 6
Case 2: 19
```

Problem F

Mesmerizing Fireflies

One night, while camping in the woods, I found a mesmerizing sight, several fireflies floating in the air, without any movement. I was so mesmerized by the sight that I decided to put my tent over the fireflies. Geometrically speaking, we can consider the space to be 2 dimensional, the x axis being the ground and y axis being vertical direction. The *i*th firefly was at position (x_i, y_i) where $y_i > 0$. My tent is a triangle with two vertices on the x axis and a third vertex at a point above the x axis. I wanted to put the tent so that each firefly was either strictly inside my tent or on the border of my tent (i.e. the edge of the triangle). **Note that the tent must be stable**, meaning that if the x coordinates of the base points are $x_A < x_B$ and the x coordinate of the top point is x_C, then $x_A \le x_C \le x_B$ must be ensured.

Now being the competitive programmer I am, I started to think how I could put my tent up so that the area covered by the tent (i.e. the area covered by the triangle described) would be minimal. Of course, that is your job (because I don't have time to solve the problem, I just want to look at the fireflies).

Due to a very difficult technical reason that you won't understand (and neither do I), just the value of the minimal area of the triangle won't be useful. Instead, you will have to also find a rectangle with minimal area having two adjacent points on the x axis such that each firefly is strictly inside the rectangle or on the border of the rectangle. If the minimal area of the triangle is **A** and the minimal area of the rectangle is **B** then I want to know the value of **A / (2 × B).**

Input

First line of input will contain a single integer **T** denoting the number of independent test cases. Then the descriptions of T test cases follow. Each test case will start with a line containing an integer **n** denoting the number of fireflies. Then **n** lines follow, the *i*th one containing two integers **x_i** and **y_i** denoting the position of the *i*th firefly.

- **$1 \le T \le 20$**
- **$2 \le n \le 50000$**
- **$-10^8 \le x_i \le 10^8$**
- **$0 < y_i \le 10^8$**
- **In a single test case, no two fireflies will be at the same position**
- **All fireflies will not have the same x coordinate in one test case**

Output

For each test case, output the value **A / (2 × B)** in a separate line as described above. **Print the value rounded to exactly 7 digits after decimal (even if they are all 0's). If the value is less than 1 then make sure to print a 0 before the decimal. Do not print extra space or extra leading 0's.**

Sample Input	Output for Sample Input

2
2
5 5
10 5
3
-10 10
0 2
20 25
	1.0000000
	0.4166667

In the first test case, the minimal area of the rectangle is 25 and the minimal area of the triangle is 50.

In the second test case, the minimal area of the rectangle is 750 and the minimal area of the triangle is 625.

Problem G

Bowling Figure

Bangladesh Association of Problem Setters

In the thrilling world of cricket, understanding a bowler's performance is crucial in evaluating a match. Bowling figures provide essential statistics that reflect a bowler's efficiency. The bowling figures consist of the number of overs bowled, runs conceded, and wickets taken. Let's unravel a problem that involves calculating these figures based on ball-by-ball information.

You are provided with a sequence of ball-by-ball information for a specific bowler during a cricket match. Each ball can result in various outcomes:

Wicket (W): The bowler takes a wicket.
Runs (0, 1, 2, 3, 4, 5, 6): The bowler concedes runs.

You may safely assume that there are no wide balls, no-balls, leg byes or any other extra runs in the provided information. Your task is to calculate the bowler's bowling figures based on the given ball-by-ball information.

Input

The first line contains a single integer **T (1 ≤ T ≤ 1000)**, the number of test cases.

For each test case, the second line contains a string **S, 1≤ |S| ≤ 60** and each character in the string represents a ball. The ball descriptions are either '**W**' for a wicket or a digit '**0**' to '**6**' representing the runs conceded in that ball.

Output

For each test case, output the bowler's bowling figures in the format: "**X.Y over(s) Z run(s) P wicket(s)**". Here, **X** is the total number of over(s), **Y** is the total number of ball(s), **Z** is the total run(s) conceded, and **P** is the total wicket(s) taken by the bowler. Use “s” in output due to plural form if over, run or wicket is greater than 1. See the sample input output below for more clarity.

Sample Input

```
3
123011121
W1W
111W11
```

Output for Sample Input

```
1.3 Overs 12 Runs 0 Wicket.
0.3 Over 1 Run 2 Wickets.
1.0 Over 5 Runs 1 Wicket.
```

Problem H

Filler Problem

Just like anime, contests also have filler. A problem low in quality, but necessary to increase the total number of problems. Here is one.

You are given a sequence of **N** non-negative integers $\mathbf{a_1, a_2, \ldots, a_N}$. For each prefix of this sequence, determine if the elements can be partitioned into 3 sets with equal sum.

Input

The first line will contain a single integer **T (1 ≤ T ≤ 5)**, number of test cases. First line of each test case will have a single integer **N (1 ≤ N ≤ 120)**, the size of the sequence. The second line has N space separated non-negative integers $\mathbf{a_1, a_2, \ldots, a_N}$. For one test case, $\sum_{i=1}^{N} \mathbf{a_i} \le \mathbf{15000}$ holds.

Output

For each test case, print the case number in a single line followed by N integers in separate lines. Print 1 if the prefix can be partitioned. Otherwise print 0.

Sample Input

```
2
8
2 1 4 2 3 2 3 4
7
0 0 0 0 2 2 2
```

Output for Sample Input

```
Case 1:
0
0
0
0
1
0
0
1
Case 2:
1
1
1
1
0
0
1
```

A valid partitioning of the 5th prefix in test case 1 is {2,2}, {1,3}, {4}.