Topic 0: Introduction, Implementation

Handout

CS 41100 - CP3 Competitive Programming III (Spring 2025)

Purdue University

Instructor: Zhongtang Luo

Date: January 9, 2025

Problem: B Road Band
Link: `https://vjudge.net/problem/Gym-104757B`

Problem: Mathematical Problem
Link: `https://vjudge.net/problem/CodeForces-1916D`

Problem: Cyclic Substrings
Link: `https://vjudge.net/problem/Gym-104857C`

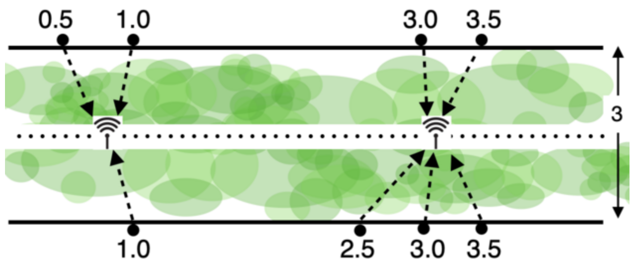
Problem: ICPC Ranking
Link: `https://vjudge.net/problem/HDU-4789`

Problem: Alice and Bomb
Link: `https://vjudge.net/problem/Baekjoon-13801`

Problem: Sumdoku
Link: `https://vjudge.net/problem/Baekjoon-15303`

B Road Band

All the residents of the rural community of Axes Point live on one of two parallel streets separated by a band of green park land. Recently, the local board of supervisors received a grant to (finally) bring wireless service to the town. The grant provides enough money for them to install k access points, and the supervisors have decided to place them in a straight line on County Road "B," which lies in the wooded band midway between the two residential streets. They want to place them in a way that minimizes the distance between users and their nearest access point. Specifically, they want to minimize the sum of the squares of the distances of each user from their nearest access point. For instance, Figure 1 shows two streets with eight customers and their locations along the streets (this is the first sample input). The streets are 3 units apart, and two access points have been placed at points midway between the two streets so that the sum of the eight squared distances is minimized.



Given the locations of all customers along each of the two streets, the distance between the streets, and the number of access points, help the local government determine the minimum sum of squared distances that can be achieved.

Input

There are three lines of input. The first line contains four integers m, n, k, s, where m and n ($1 \le m, n \le 1\,000$) are the number of customers along each of the two roads, k ($1 \le k \le \min(\max(m, n), 100)$) is the number of access points to be placed, and s ($1 \le s \le 50$) is the distance separating the two roads. The second line contains m floating-point values $x_1, x_2, \ldots, x_m$ ($0 \le x_i \le 1\,000$) giving the locations of the m customers along the first road. The third line is similar, containing n floating-point locations of customers along the second road. All values on each of the second and third lines will be distinct (but some values may appear in both lines). Customer locations will have no more than four decimal places.

Output

Output a single floating-point value equal to the minimum sum of squared distances for each customer from the closest of the k access points. Answers should be correct to within an absolute or relative error of 10^{-5}.

Examples

Input

```
4 4 2 3
0.5 1.0 3.0 3.5
1.0 2.5 3.0 3.5
```

Output

```
18.86666667
```

Source

2023-2024 ICPC East North America Regional Contest (ECNA 2023)

Mathematical Problem

The mathematicians of the 31st lyceum were given the following task:

You are given an **odd** number n, and you need to find n different numbers that are squares of integers. But it's not that simple. Each number should have a length of n (and should not have leading zeros), and the multiset of digits of all the numbers should be the same. For example, for `234` and `432`, and `11223` and `32211`, the multisets of digits are the same, but for `123` and `112233`, they are not.

The mathematicians couldn't solve this problem. Can you?

Input

The first line contains an integer t ($1 \le t \le 100$) — the number of test cases.

The following t lines contain one **odd** integer n ($1 \le n \le 99$) — the number of numbers to be found and their length.

It is guaranteed that the solution exists within the given constraints.

It is guaranteed that the sum of n^2 does not exceed 10^5.

The numbers can be output in any order.

Output

For each test case, you need to output n numbers of length n — the answer to the problem.

If there are several answers, print any of them.

Examples

Input

```
3
1
3
5
```

Output

```
1
169
196
961
16384
31684
36481
38416
43681
```

Source

Codeforces Good Bye 2023

Cyclic Substrings

Mr. Ham is interested in strings, especially palindromic strings. Today, he finds a string s of length n.

For the string s of length n, he defines its *cyclic substring* from the i-th character to the j-th character ($1 \le i, j \le n$) as follows:

- If $i \le j$, the cyclic substring is the substring of s from the i-th character to the j-th character. He denotes it as $s[i..j]$.
- If $i > j$, the cyclic substring is $s[i..n] + s[1..j]$, where $+$ denotes the concatenation of two strings. He also denotes it as $s[i..j]$.

For example, if $s =$ `12345`, then $s[2..4] =$ `234`, $s[4..2] =$ `4512`, and $s[3..3] =$ `3`.

A string t is *palindromic* if $t[i] = t[n - i + 1]$ for all i from 1 to n. For example, `1221` is palindromic, while `123` is not.

Given the string s, there will be many cyclic substrings of s which are palindromic. Denote P as the set of all **distinct** cyclic substrings of s which are palindromic, $f(t) (t \in P)$ as the number of times t appears in s as a cyclic substring, and $g(t) (t \in P)$ as the length of t. Mr. Ham wants you to compute

$$\sum_{t \in P} f(t)^2 \times g(t)$$

The answer may be very large, so you only need to output the answer modulo 998 244 353.

Input

The first line contains a number n ($1 \le n \le 3 \times 10^6$), the length of the string s.

The second line contains a string s of length n. Each character of s is a digit.

Output

Output a single integer, denoting the sum modulo 998 244 353.

Examples

Input

```
5
01010
```

Output

```
39
```

Input

```
8
66776677
```

Output

```
192
```

Source

The 2023 ICPC Asia Hefei Regional Contest (The 2nd Universal Cup. Stage 12: Hefei)

ICPC Ranking

There are so many submissions during this contest. Coach Pang can not determine which team is the winner. Could you help him to print the score board?

As we all know, the contest executes by the teams submit codes for some problems. Simply, we assume there are three kinds of results for every submission.

ERROR There was something wrong. The team didn't solve the problem but won't get any penalty.

NO Sorry, the code was not right. The team didn't solve the problem.

YES Yeah, AC. The team solved the problem.

To make the contest more exciting, we set a time called frozen time. If one team doesn't solve one of the problems before the frozen time and submits at least one submission on this problem after or exactly at the frozen time, this problem of this team is called frozen. For different team, the frozen problems will be different. For the frozen problems, the score board will only show how many submissions the team has been submitted but won't show the result of the submissions.

The rank is determined by the following factors. Remember, we only consider the unfrozen problems. The frozen problems will be ignored. The following factors are ordered with the priority from high to low.

Solved the team who solves more problems will place higher.

Penalty the team who gets less penalty time will place higher. Only solved problems will give penalty time. Every solved problem will give $T + 20X$ penalty time. T is the time of the first YES, X is the number of NOs before the first YES.

Last Solved the team who solved their last problem earlier will place higher. If there is a tie, we compare their second last problems, then their third last problems, etc.

Name The team whose name comes later in lexicographical order will place higher.

At the end of the contest, the score board will be unfrozen. First, choose the team which has frozen problems with the lowest rank. Then choose one frozen problem of this team. If the team has multiple frozen problems, choose their first frozen problem in alphabetic order. Then show the result of the problem, recalculate the rank, change the score board and make the problem unfrozen for this team. Repeat this procedure until no teams have frozen problems. Then we get the final score board.

Please help Coach Pang to print the initial score board, the final score board and the process of the unfreeze procedure.

Input

The first line contains an integer C, which indicates the number of test cases.

For each test case, the first line will have four integers n, m, T and t. n ($1 \leq n \leq 50000$) is the number of submissions. m ($1 \leq m \leq 26$) is the number of problems. T ($1 \leq T \leq 10000$) is the total time of the contest. t ($0 \leq t \leq T$) is the frozen time.

The following n lines, each line will be in the form "Name Problem Time Result". Name is the team's name which only contains letters and digits with at most 20 characters. Problem is the identifier of the problem which is a capital letter from A to them-th letter. Time is a integer indicates the submission time which greater than or equal to 0 and less than T. Result is one string which equal YES, NO or ERROR.

Every team will have at least one submission. If one team has multiple submissions at the same time, we consider the ERRORs come before NOs and NOs come before YESs.

Output

For each test case, first print "Case #x:", x is the case number start from 1.

Then print the initial score board (before the unfreeze procedure) ordered by the rank. For every team, print "Name Rank Solved Penalty A B C ...". Name is the team's name. Rank is the team's rank. Solved is the number of solved problems. Penalty is the team's penalty time.

A B C ... is the condition of every problem is the format below:

+x The problem is unfrozen and solved. x is the number of NOs before the first YES. If x = 0, print “+” instead of “+0”.

-x The problem is unfrozen but not solved. x is the number of NOs. If x = 0, print “.” instead of “-0”.

-x/y The problem is frozen. x is the number of NOs before the frozen time. y is the number of submissions after or exact at the frozen time. If x = 0, print “0/y” instead of “-0/y”.

After the initial score board, print the process of the unfreeze procedure. During the unfreeze procedure, every time one team unfreezes one frozen problem and causes the change of its rank, print “Name1 Name2 Solved Penalty”. Suppose this team is team A. Name1 is the name of team A. Name2 is the name of the team which is overtaken by team A with the highest rank. Solved is the new number of the solved problems of team A. Penalty is the new penalty time of team A.

Finally print the final score board (after the unfreeze procedure) as the same format as the initial score board.

Examples

Input

```
20 12 300 240
Epic B 12 YES
Epic A 14 NO
Rivercrab E 25 YES
Two2erII B 100 NO
Epic A 120 YES
Rivercrab I 150 NO
Two2erII C 160 NO
Epic C 180 YES
Two2erII C 180 NO
Rivercrab F 226 YES
Two2erII C 230 YES
Two2erII L 241 YES
Epic F 246 YES
Epic G 260 YES
Rivercrab I 289 YES
Epic D 297 YES
Musou H 299 YES
Musou I 299 YES
Musou J 299 YES
Musou K 299 YES
```

Output

```
Case #1:
Epic 1 3 332 +1 + + 0/1 . 0/1 0/1 . . . . .
Rivercrab 2 2 251 . . . . + + . . -1/1 . . .
Two2erII 3 1 270 . -1 +2 . . . . . . . . 0/1
Musou 4 0 0 . . . . . . . 0/1 0/1 0/1 0/1 .
Musou Two2erII 2 598
Two2erII Musou 2 511
Musou Rivercrab 3 897
Rivercrab Musou 3 560
Musou Epic 4 1196
Epic Musou 4 629
Epic 1 6 1135 +1 + + + . + + . . . . .
```

```
Musou 2 4 1196 . . . . . . . . + + + + .
Rivercrab 3 3 560 . . . . + + . . +1 . . .
Two2erII 4 2 511 . -1 +2 . . . . . . . . +
```

Source

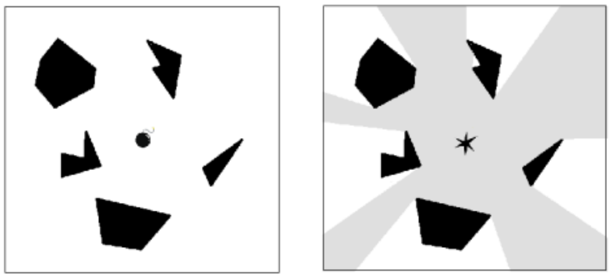
2013 Asia Chengdu Regional Contest

Alice and Bomb

Alice and Bob were in love with each other, but they hate each other now. One day, Alice found a bag. It looks like a bag that Bob had used when they go on a date. Suddenly Alice heard tick-tack sound. Alice intuitively thought that Bob is to kill her with a bomb. Fortunately, the bomb has not exploded yet, and there may be a little time remained to hide behind the buildings.

The appearance of the ACM city can be viewed as an infinite plane and each building forms a polygon. Alice is considered as a point and the bomb blast may reach Alice if the line segment which connects Alice and the bomb does not intersect an interior position of any building. Assume that the speed of bomb blast is infinite; when the bomb explodes, its blast wave will reach anywhere immediately, unless the bomb blast is interrupted by some buildings.

The figure below shows the example of the bomb explosion. Left figure shows the bomb and the buildings (the polygons filled with black). Right figure shows the area (colored with gray) which is under the effect of blast wave after the bomb explosion.



Your task is to write a program which reads the positions of Alice, the bomb, and the buildings and calculate the minimum distance required for Alice to run and hide behind the building.

Input

The input contains multiple test cases. Each test case has the following format:

The input contains multiple test cases. Each test case has the following format:

$$\begin{array}{l} N \\ b_x\ b_y \\ m_1\ x_{1,1}\ y_{1,1}\ \cdots\ x_{1,m_1}\ y_{1,m_1} \\ \vdots \\ m_N\ x_{N,1}\ y_{N,1}\ \cdots\ x_{N,m_N}\ y_{N,m_N} \end{array}$$

The first line of each test case contains an integer N ($1 \le N \le 100$), which denotes the number of buildings. In the second line, there are two integers b_x and b_y ($-10000 \le b_x, b_y \le 10000$), which means the location of the bomb. Then, N lines follows, indicating the information of the buildings.

The information of building is given as a polygon which consists of points. For each line, it has an integer m_i ($3 \le m_i \le 100$, $\sum_{i=1}^{N} m_i \le 500$) meaning the number of points the polygon has, and then m_i pairs of integers $x_{i,j}$, $y_{i,j}$ follow providing the x and y coordinate of the point.

- Alice is initially located at $(0, 0)$.
- $N = 0$ denotes the end of the input.

Output

For each test case, output one line which consists of the minimum distance required for Alice to run and hide behind the building. An absolute error or relative error in your answer must be less than 10^{-6}.

Examples

Input

```
1
1 1
4 -1 0 -2 -1 -1 -2 0 -1
1
0 3
4 1 1 1 2 -1 2 -1 1
1
-6 -6
6 1 -2 2 -2 2 3 -2 3 -2 1 1 1
1
-10 0
4 0 -5 1 -5 1 5 0 5
1
10 1
4 5 1 6 2 5 3 4 2
2
-47 -37
4 14 3 20 13 9 12 15 9
4 -38 -3 -34 -19 -34 -14 -24 -10
0
```

Output

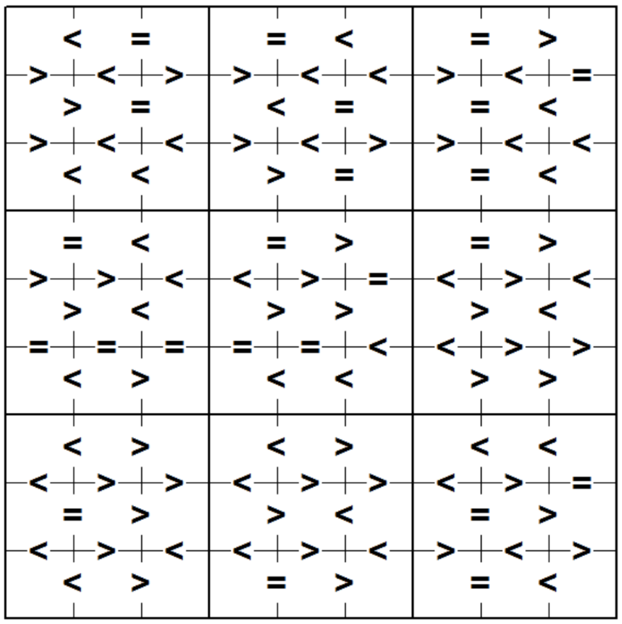
```
1.00000000
0.00000000
3.23606798
5.00000000
1.00000000
11.78517297
```

Source

JAG Practice Contest 2010

Sumdoku

Sumdoku is a variant of the game Sudoku. As in Sudoku, the aim is to fill in a 9-by-9 grid with the digits 1 through 9 so that each digit 1 through 9 occurs exactly once in each row, exactly once in each column, and exactly once in each of the 9 3-by-3 sub-squares subject to constraints on the choices. In Sudoku, the constraints are that certain squares must contain fixed values. In Sumdoku, the constraints are on the sum of adjacent squares within each 3-by-3 sub-square. In the illustration below, the symbols <, =, and > indicate that the sum of the values on either side (or above and below) the symbol must have a sum less than 10, equal to 10, or greater than 10, respectively.



Write a program to solve Sumdoku problems.

Input

The first line of input contains a single decimal integer P, $(1 \leq P \leq 10\,000)$, which is the number of data sets that follow. Each data set should be processed identically and independently.

Each data set consists of 16 lines of input. The first line contains the data set number, K. The following 15 lines consist of the characters <, =, or >. Rows 1, 3, 5, 6, 8, 10, 11, 13, and 15 contain 6 characters corresponding to constraints on the sum of values to the left and right of the symbol. Rows 2, 4, 7, 9, 12, and 14 contain 9 characters corresponding to constraints on the sum of values above and below the symbol. Note: Solutions to some problems may not be unique. The judging program will just check whether your solution satisfies the constraints of the problem (row, column, 3-by-3 box, and inequality constraints).

Output

For each data set, there are 10 lines of output. The first output line consists of the data set number, K. The following 9 lines of output consist of 9 decimal digits separated by a single space. The value in the jth position in the ith line of the 9 output lines is the solution value in column j of row i.

If there are multiple solutions, print the lexicographically smallest one if the answer is read by row-major order.

Examples

Input

```
1
1
<==<=>
><>><<><=
>=<==<
><<>><>><<
<<>==<
=<=>>
>><<>><><
><>>>><
=====<><>
><<<>>
><><<<
><><><><>=
=>><=>
><<<><><><
><>=>=<
```

Output

```
1
5 3 7 8 2 1 6 4 9
9 6 4 5 3 7 8 2 1
8 1 2 9 4 6 7 3 5
6 4 5 3 7 8 1 9 2
7 8 1 4 9 2 5 6 3
3 2 9 6 1 5 4 7 8
2 7 8 1 6 9 3 5 4
1 9 3 7 5 4 2 8 6
4 5 6 2 8 3 9 1 7
```

Source

2017 Greater New York Programming Contest