The 2015 ACM ICPC Asia Regional Contest Dhaka Site

Sponsored by IBM

Hosted by NSU
Dhaka, Bangladesh

14th November 2015
You get 18 Pages
10 Problems
&
300 Minutes

acm International Collegiate Programming Contest

IBM event sponsor

Rules for ACM-ICPC 2015 Asia Regional Dhaka Site Onsite Contest:

a) Solutions to problems submitted for judging are called runs. Each run is judged as accepted or rejected by the judge, and the team is notified of the results.

b) Notification of accepted runs will **NOT** be suspended at the last one hour of the contest time to keep the final results secret. Notification of rejected runs will also continue until the end of the contest. But the teams will not be given any balloon and the public rank list will not be updated as usual in the last one hour.

c) A contestant may submit a clarification request to judges only through the CodeMarshal clarification system. If the judges agree that an ambiguity or error exists, a clarification will be issued to all contestants. Judges may prefer not to answer a clarification at all in which case that particular clarification request will be marked as IGNORED in the CodeMarshal clarification page.

d) Contestants are not to converse with anyone except members of their team and personnel designated by the organizing committee while seated at the team desk. **They cannot even talk with their team members when they are walking around the contest floor to have food or any other purpose.** Systems support staff or judges may advise contestants on system-related problems such as explaining system error messages.-

e) While the contest is scheduled for a particular time length (five hours), the contest director has the authority to alter the length of the contest in the event of unforeseen difficulties. Should the contest duration be altered, every attempt will be made to notify contestants in a timely and uniform manner.

f) **A team may be disqualified by the Contest Director** for any activity that jeopardizes the contest such as dislodging extension cords, unauthorized modification of contest materials, distracting behavior or communicating with other teams. The **judges on the contest floor** will report to the **Judging Director** about distracting behavior of any team. **The judges can also recommend penalizing a team with additional penalty minutes for their distracting behavior.**

g) Nine, ten, eleven or twelve problems will be posed. So far as possible, problems will avoid dependence on detailed knowledge of a particular applications area or particular contest language. Of these problems at least one will be solvable by a first year computer science student, another one will be solvable by a second year computer science student and rest will determine the winner.

h) Contestants will have foods available in their contest room during the contest. So they cannot leave the contest room during the contest without explicit permission from the judges. **The contestants are not allowed to communicate with any contestant (even contestants of his own team) or coaches when they are outside the contest arena.**

i) Teams can bring up to **200 pages of printed materials** with them and they can also bring five additional books. But they are not allowed to bring calculators or any machine-readable devices like CD, DVD, Pen-drive, IPOD, MP3/MP4 players, floppy disks etc. **Mobile phone MUST be switched off at all times and stored inside a bag or any other place that is publicly non visible during the entire contest time. Failure to adherence to this clause under any condition will very likely lead to strict disciplinary retaliation and possible disqualification.**

j) With the help of the volunteers, the contestants can have printouts of their codes for debugging purposes. **Passing of printed codes to other teams is strictly prohibited.**

k) The decision of the judges is final.

l) Teams should inform the volunteers/judges if they don't get verdict from the codemarshal within 5 minutes of submission. Teams should also notify the volunteers if they cannot log in into the CodeMarshal system. These sort of complains will not be entertained after the contest.

A Automatic Cheater Detection

Input: Standard Input
Output: Standard Output

In recent times a very bad incident is happening repeatedly in Wonderland (Poets claim that she is the queen of all countries), questions of many public examination and admission tests are being leaked before the exam. The effect is less terrible when the exam involved is a written public exam, because still one has to go to exam halls and write things correctly. But if the exam involved has only MCQs (Multiple Choice Questions) the situation becomes grave - anyone who has a little bit of memorization power can get good marks if he gets MCQ questions with solution the day before the exam.

For technical reasons and added expenditure, such an MCQ based exam cannot always be taken again. So you are given the responsibility of detecting cheaters of such an MCQ based exam. The details of this exam and cheater detection process is given below:

(a) There are at most **50000** MCQ questions in the exam.

(b) All questions have a difficulty, a status and a result associated with it. This difficulty is denoted with an integer **d** between **1** and **10** (Inclusive). The higher the value of d, the more difficult the question is. The status **s** can be either **1** (The question was leaked) or **0** (The question was not leaked). The result **r** is either **'c'** (The answer given by the student is correct) or **'i'** (The answer given by the student is incorrect). So a single MCQ question is denoted with two integers **d**, **s** and a character **r**.

(c) A suspicious activity is counted when someone answers a leaked question with difficulty $\mathbf{d_1}$ correctly but fails to answer a not leaked question with difficulty $\mathbf{d_2}$ correctly and ($d_1 > d_2$).

(d) Given the description of questions in a set (including the correctness of each question for a particular examinee) your job is to write a program to find the total number of suspicious activities for that examinee.

Input*

First line of the input file contains a positive integer **T (T ≤ 80)** which denotes the number of test cases. The description of each test case is given below:

Each test case starts with a positive integer **Q (1 ≤ Q ≤ 50000)** which denotes the number of questions in the exam and each of the next **Q** lines describes **1** MCQ question. Each of the line contains two integers $\mathbf{d_i}$ **(1 ≤** $\mathbf{d_i}$ **≤ 10)**, $\mathbf{s_i}$ and a character $\mathbf{r_i}$, here $\mathbf{d_i}$ denotes the difficulty, $\mathbf{s_i}$ denotes the status (1 means it was leaked and 0 means that it was not leaked) and $\mathbf{r_i}$ denotes whether the examinee answered the

question correctly **(c)** or incorrectly **(i)**. In reality a day long MCQ exam does not have **50000** questions but this is just being given as input to test the efficiency of your algorithm as your program will be used to find cheaters among ten million students in a very short time.

Output

For each set of input produce one line of output. This line contains an integer which denotes the total number of suspicious activities for the given input.

Sample Input	Output for Sample Input
2 4 2 0 i 3 1 c 4 1 c 5 1 i 4 2 0 i 4 1 c 3 0 i 5 1 c	2 4

*Warning: Large I/O file, user faster input output functions.

Illustration

In Sample 1, one suspicious activity is that the first not leaked question was answered incorrectly but the second leaked question was answered correctly (Although first one had a higher difficulty). The other suspicious activity is that first not leaked question was answered incorrectly but the third leaked question was answered correctly (Although it had a higher difficulty). So the total no of suspicious activities is 2.

Similarly, in 2nd sample there are four suspicious activities with the answering of the following question pairs (Q2, Q1), (Q4, Q1), (Q2, Q3), (Q4, Q3)

B Counting Weekend Days

Input: Standard Input
Output: Standard Output

The contestants probably don't know how eagerly problem-setters (The people who prepare problems for a programming contest) wait for the weekend to make problems that would terrorize contestants :-). So before a month begins, some problem-setters try to calculate the number of weekend days in that month and plans accordingly. Can you help them to calculate this?

There are seven days in a week namely Sunday (**SUN**), Monday (**MON**), Tuesday (**TUE**), Wednesday (**WED**), Thursday (**THU**), Friday (**FRI**) and Saturday (**SAT**). There are twelve months in a year, January (**JAN**), February (**FEB**), March (**MAR**), April (**APR**), May (**MAY**), June (**JUN**), July (**JUL**), August (**AUG**), September (**SEP**), October (**OCT**), November (**NOV**) and December (**DEC**). These months have **31**, **28**, **31**, **30**, **31**, **30**, **31**, **31**, **30**, **31**, **30** and **31** days respectively. In leap years, the month of February has **29** days. In the bracket the three letter code for each month and each day are shown. Unlike many countries of the world Friday (**FRI**) and Saturday (**SAT**) are considered weekend days in Bangladesh. Given a month and the name of the first day of that month, you will have to find out the total no of weekend days in that month.

Input

First line contains an integer **T** (**T≤100**) which denotes the number of test cases. The input for each set is given in a single line. This line contains two strings **MTH** and **DAY**, here **MTH** is the three digit code of the month and **DAY** is the three digit code for the name of the first day of that Month.

Output

For each line of input produce one line of output. It contains a single integer which denotes the number of weekend days (Fridays and Saturdays) in that month. You must do your calculation assuming that the year is not a leap- year.

Sample Input	Output for Sample Input
3 JAN SUN FEB SUN OCT THU	8 8 10

October

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

Illustration of third sample input:

In the third sample input we are asked to count the number of weekend days of a month October whose first day (October 1) is Thursday. The calendar on the left depicts this and it can be seen that there are 10 weekend days (colored red) in this month.

C Toll Management

Input: Standard Input
Output: Standard Output

Another attempt of revising toll system is rejected by the people of Byteland. The Government is in big trouble now. They are seeking help of Hashmat, the brave warrior and the great programmer of Byteland to solve the problem. Hashmat has come up with a new idea to fix this toll management problem.

There are **N** cities and **M** bidirectional highways among these cities. The government proposed some toll (may be different toll for different highways) for each of the highways. But people think they are being over charged. Hashmat's idea was to make both the people and the Government happy. In his plan he decided to keep the toll of the highways as it is in Government's proposal but he would publish a special set of **N - 1** highways. There are two conditions for a set of **N –** 1 highways to be special. These **N – 1** highways have to connect all the cities and a person will be able to go from a city to any other city spending minimum toll in any of these highways. Please note, using these special highways does not guarantee minimum sum of toll but it guarantees you minimum individual toll.

People of Byteland are happy with the idea of special highway, but the Government is not happy as they want more toll from the highway sector. They called up a meeting and formed a committee to find two values for all the highways. Let these values be $\mathbf{A_i}$ and $\mathbf{B_i}$ for the **i**'th highway and defined as follows:
1) $\mathbf{A_i}$: The maximum amount of toll they can add to the **i**'th highway so that Hashmat's set remains special.
2) $\mathbf{B_i}$: The maximum amount of toll they can decrease from the **i**'th highway so that Hashmat's set remains special.

In other words, if $\mathbf{C_i}$ is the current toll of the i'th highway, then if the Government updates the toll of the highway to $\mathbf{C_i + A_i}$ **(or** $\mathbf{C_i - B_i}$**)**, Hashmat's set remains special. Please note, while finding out $\mathbf{A_i}$ and $\mathbf{B_i}$ other tolls remain unchanged.

This time Hashmat does not want to help the Government. He thinks this is a conspiracy against the people of Byteland. So they came to you. Will you help them to find out $\mathbf{A_i}$ and $\mathbf{B_i}$ for all the highways?

Input*

First line of the input contains a positive integer **T (T < 25)**, denoting the number of test cases.

First line of each test contains two integer numbers **N** and **M (1 ≤ N ≤ 10000, N-1 ≤ M ≤ 100000**), denoting the number of city and number of highway respectively. Each of the next **M** lines contains the description of a highway, where the **i-th** line contains three integer numbers $\mathbf{U_i}$, $\mathbf{V_i}$ and $\mathbf{C_i}$ **(1 ≤** $\mathbf{U_i, V_i}$ **≤ N,** $\mathbf{U_i}$ **!=** $\mathbf{V_i}$**, 0 ≤** $\mathbf{C_i}$ **≤ 1000)**, that means there is a highway between city $\mathbf{U_i}$ and city $\mathbf{V_i}$ and the toll of the highway is $\mathbf{C_i}$. You may consider the highways to be bidirectional. Note that the first **N-1** highways in the input are the special highways. You may assume that there will be no invalid data in the input file.

Output

For each test case, output the test case number and a single integer **S**, where

$$\mathbf{S} = \sum_{i=1}^{M}(i * A_i + i^2 * B_i)$$

If the value of $\mathbf{A_i}$ or $\mathbf{B_i}$ is infinite, replace the value with **-1**.

Sample Input	Output for Sample Input
2 3 3 1 2 5 2 3 7 1 3 10 4 5 1 3 1 3 4 2 1 2 3 1 4 4 2 4 5	Case 1: 30 Case 2: 72

*Warning: Large I/O file, user faster input output functions.

Illustration:

In the figures below the dashed lines are normal highway and the solid lines are special highway.

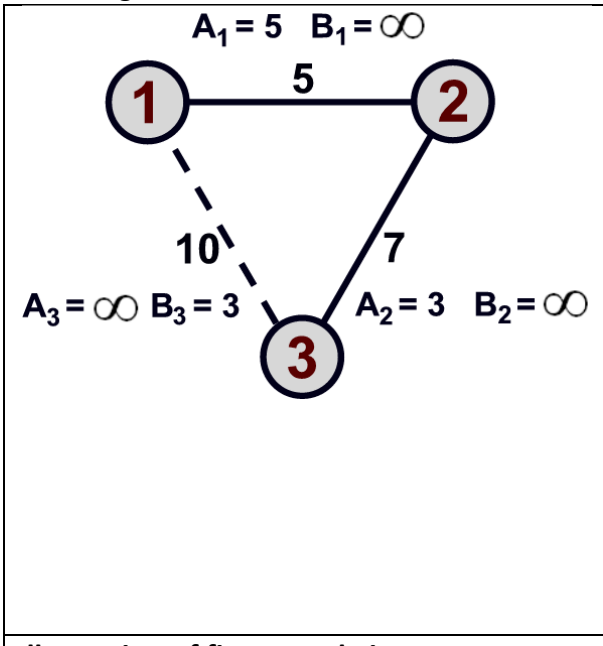


Illustration of first sample input:

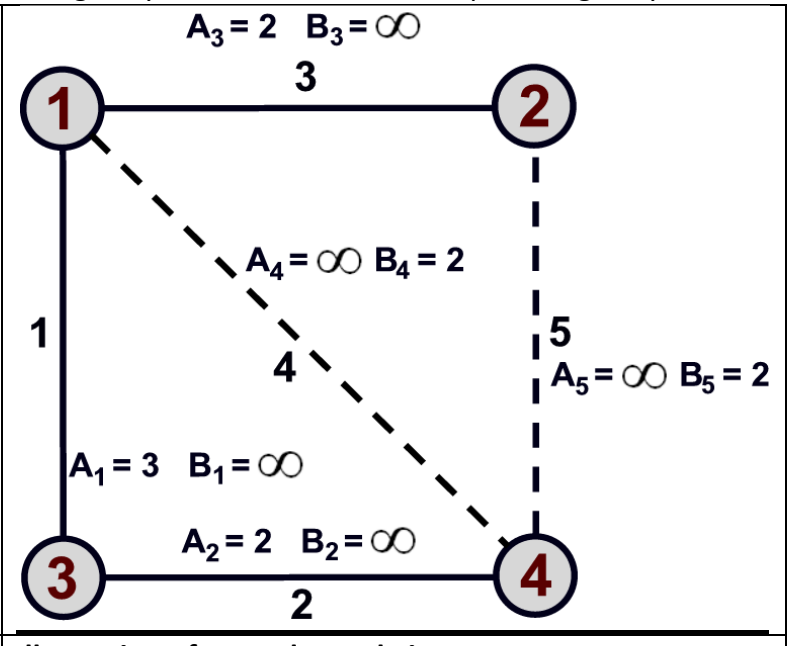


Illustration of second sample input:

D

Owllen

Input: Standard Input
Output: Standard Output

Wise owl has got a string **S** with **N (1 ≤ N ≤ 10^5)** characters. All the characters of **S** are lowercase English letters. Now she challenges Fallen to find out a string **T** of length **N** such that the length of the **LCS** (Longest Common Subsequence) of **S** and **T** is minimum. **T** also should be consisted of lowercase English letters only.

Now it is Fallen's problem to find out the string **T**. But you need to print the minimum length of such LCS given that Fallen has found T correctly.

Input*

Input file starts with a single integer **T (1 ≤ T ≤ 50)**, **T** test cases following. Each of the next **T** test cases has one string **S** on a line.

Output

For each case print your output in format, **"`Case X: Y`"**, on a single line where X denotes the case number starting from 1 and Y denotes the length of the shortest possible **LCS**.

Sample Input	Output for Sample Input
2 ab efzadeuopqxrvwxaghijklmnbcastbqy	Case 1: 0 Case 2: 1

*Note: Please use faster I/O routine as the input file is quite large(~4 MB).

E Sum of MSLCM

Input: Standard Input
Output: Standard Output

A positive integer **N** can be the **LCM** (Least Common Multiple) of different set of numbers. For example, **LCM(6, 24) = 24**, **LCM(12, 8) = 24**, **LCM (1,2,3,4,8) = 24** etc. For a given number N, maximum sum LCM indicates the set of numbers whose **LCM** is **N** and summation is maximum. Let, **MSLCM(N)** denote this maximum sum of numbers. Given the value of **N** you will have to find the value: $\sum_{i=2}^{N} MSLCM(i)$. Obviously, in a set the same value never comes twice.

Input

Input file contains at most **200** lines. Each line contains a positive integer which denotes the value of **N (1 < N < 20000001)**. Input is terminated by a line containing a single zero, which should not be processed.

Output

For each positive number **N** in the input, produce one line of output. This line contains an integer which denotes the value $\sum_{i=2}^{N} MSLCM(i)$

Sample Input	Output for Sample Input
10 1000 0	86 823080

F Unique Party

Input: Standard Input
Output: Standard Output

In a multi-storied building, there are some families who know each other very well. They like to have a lot of fun, so they often want to get together in one of those apartments to have a party. To understand the uniqueness of this group, let us consider a co-ordinate system where (x,y,z) denotes the co-ordinate of an apartment. Here, z axis denotes the floor of the apartment and (x, y) co-ordinates form the plane parallel to the land.

Now there is an interesting fact: there are no two families in their group such that their apartments have the same (x, y) co-ordinate. Moreover, for every possible value of (x, y), there exists exactly one apartment in which one of these families live. Therefore, we can represent the floor of each of their apartments in a table as the following figure:

6	10	3	1
5	4	2	5
1	7	4	15

Here, you can consider the rows of the table parallel to x-axis and columns of the tables parallel to y-axis of the building. So, if the cell of 1strow and 1st column denotes 6, that means one of their friends live in the apartment on 6th floor of the building with co-ordinate (1, 1).

It is not always possible for each family to be present in every party, but whenever a subset of these families gets together for a party, they have to satisfy the following properties:

- This subset is formed by a rectangle (axis-parallel) drawn on the table.
- The party will be held in one of the apartments of this subset (Obviously).
- The summation of floors they need to go up or down must be minimized. If there are multiple floors which satisfy this requirement, they will always choose the higher floor.
- After selecting the apartment, if it is on a floor lower than **h**, the party will be cancelled.

For example, say the families marked in grey (in the table shown above) want to hold a party and h=3. Both 4th and 5th floor has the same summation of floors they need to go up or down (2 + 1 + 11 = 14 for 4th floor and 3 + 1 + 10 = 14 for 5th floor).As they always prefer the higher floor, it will be held on the 5thfloor. As it is not lower than 3rd floor (h=3), it also satisfies the last requirement.

You need to write a program to determine the largest number of families who can get together for a party satisfying all those requirements.

Input

The first line will contain the number of test cases, **T (1 ≤ T ≤ 20)**. Each test case starts with a line containing two integers, **R (1 ≤ R ≤ 250)** and **C (1 ≤ C ≤ 250)** denoting the number of rows and columns in the table, respectively. Then **R** lines follow, each containing **C** integers denoting the floors of corresponding apartments. Then there will be a line containing a single integer, **Q (1 ≤ Q ≤ 10)** which denotes the number of queries to follow. Then the following line contains **Q** integers, where each one

denotes the value of **h**. All integers in the input file will be positive and in the range of 32-bit signed integer.

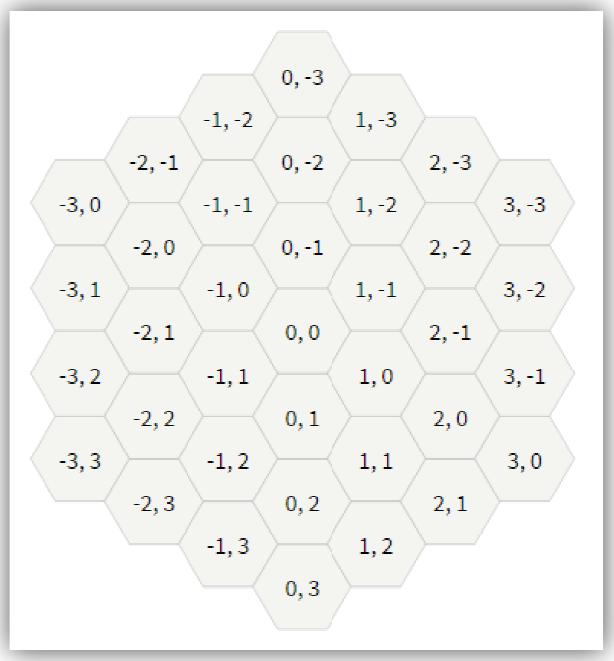
Output

For each case, print a line containing **"Case <x>:"** where **x** is the case number. Then the following **Q** lines should contain one integer: the largest number of families who can get together for the corresponding value of **h**.

Sample Input	Output for Sample Input
1 3 4 6 10 3 1 5 4 2 5 1 7 4 15 2 6 5	Case 1: 6 12

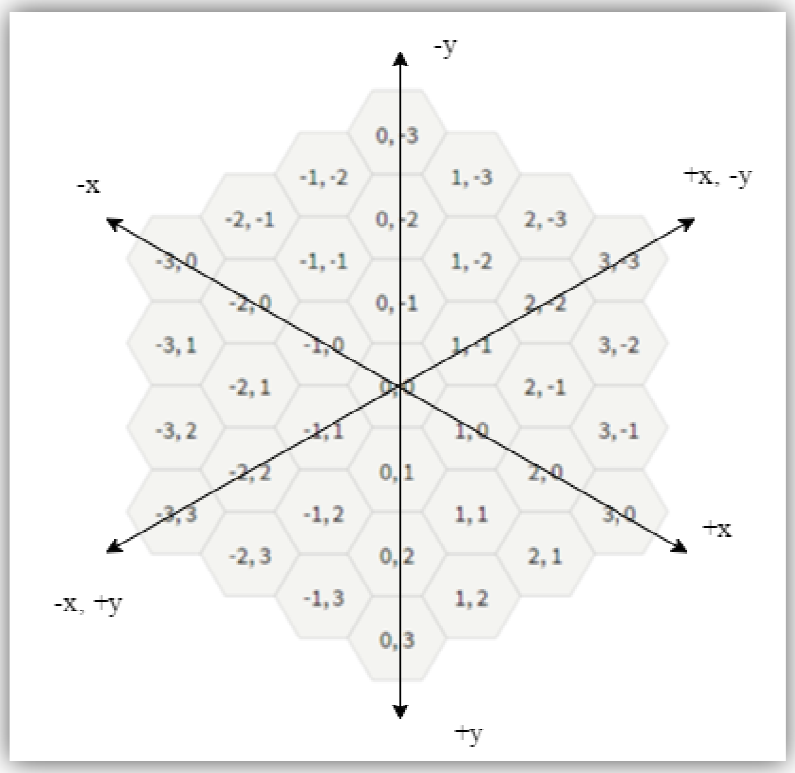
G Honey King

Input: Standard Input
Output: Standard Output



Sir Moo Moo Honey is the King of Honeycomb kingdom. Everyone knows him as Honey King. Honeycomb kingdom is a hexagonal grid of infinite size where each cell is a hexagon. Like any other grid system, the Honeycomb grid can be represented by a 2D coordinate system as the figure on the left.

From each cell if we go to up (down) direction then the **y** value of a coordinate decreases (increases) and x value remains same. Similarly if we go to up-left (down-right) direction the **x** value of a coordinate decreases (increases) and **y** value remains same. But for the direction up-right (down-left) the **x** value increases (decreases) as well as **y** value decreases (increases). So for a cell **(x, y)**, there are six surrounding cells, **up(x, y-1)**, **down (x, y+1)**, **up-left (x-1, y)**, **down-right (x+1, y)**, **up-right (x+1, y-1)** and **down-left (x-1, y+1)**.

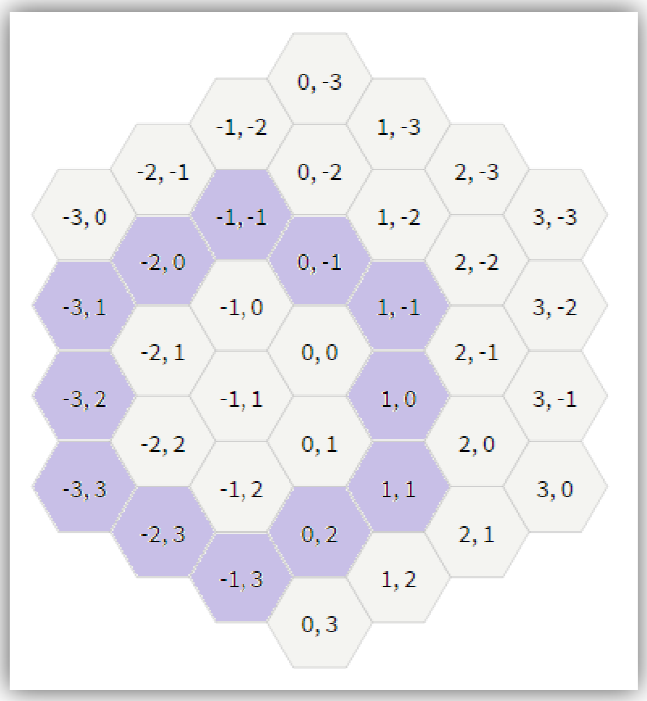


Honey King does not love his people much. He always wanted to separate them from his beautiful palace. He came up with a new idea to do this.

He will build a wall outside of his palace to separate the people of Honeycomb from his palace. The wall will be a hexagonal shaped ring of hexagonal cells. As the king is a disciplined person he likes regularity and the wall should be a regular hexagon (all six sides of the hexagon should be equal).

There are some beautiful cells in the honeycomb grid. King's palace is in one of the beautiful cells. Honey King wants to build the wall in such way that all the beautiful cells remain inside the wall. Also the king wants to minimize the area inside the wall.

For example, if cell **(0, 0)**, **(-1, 0)** and **(-2, 2)** are beautiful cells, then one way to build the wall is as follows:



The shaded cells form the wall in the image above.
Help Honey King to build the smallest regular hexagonal wall in Honeycomb kingdom such that all the beautiful cells remain inside the wall.

Input

First line of the input contains an integer number **T (T ≤ 200)**, denoting the number of test cases.
First line of each test case contains an integer number **N (1 ≤ N ≤ 100000)**, number of beautiful cells. Next **N** lines will contain **N** pairs of space separated integers X_i and Y_i **($-10^4 \le X_i, Y_i \le 10^4$, $1 \le i \le N$)**, where **(X_i, Y_i)** is the coordinate of the **i-th** beautiful cell. No two coordinates in a single test will be same. Note that, sum of **N** in all test cases will be less than **200000**.

Output

For each test case, print the test case number and number of hexagonal cells inside the smallest regular hexagonal wall containing all beautiful cells.

Sample Input	Output for Sample Input
2 1 0 0 3 0 0 -1 0 -2 2	Case 1: 1 Case 2: 7

For second test case, see the example in the problem description.

H Design New Capital

Input: Standard Input
Output: Standard Output

If you go through history you will come across many fun facts about the capital city of a country. In many countries the most famous or most populated or most industrialized cities are not declared capital. For example, Zurich or Geneva is not capital of Switzerland it is Berne. In Australia the capital is neither Melbourne nor Sydney, it is Canberra and guess what Canberra was built to be capital! In Brazil Brasilia is capital not Rio.

Head of your country decided to redesign the capital. Dr. Krakow is appointed as the head architect of this project. Having been in Europe and America, he has vast experience about city life. He wants to lay roads like New York. That is, if you see from above, it will look like a big 2d grid. He wants to build the parliament at **(0, 0)** coordinate of this grid. After that, he asks some people where would they love to stay. Of course it is not possible to allow to build homes anywhere the people like. There are many constraints to consider. One of the constraints is the parliament has to be at the optimal position. A position is optimal if summation of Manhattan distance to all the houses from the position is minimum.

For example, suppose there are three proposals to build houses at following positions: **(1, 1)**, **(-1, -1)** and **(2, 2)**. If Dr. Krakow approves to build houses at **(1, 1)** and **(-1, -1)** then **(0, 0)** would be optimal position. But if he approves **(1, 1)** and **(2, 2)** then **(0, 0)** is no more optimal position. Please note, there may be many optimal positions for some set of approval, but **(0, 0)** has to be one of them. Given some proposals, help Dr. Krakow to figure out number of ways he can approve some of the proposals so that parliament is at one of the optimal positions. Please note, there may be request for the same place from different persons. In such case Dr. Krakow can approve none or all or some of these proposals. When computing summation of Manhattan distance, all of the approved proposals are counted, that is, if there are two approved proposals at the same place, then distance from this place to the parliament will be counted twice.

Input

The first line of the input will contain a positive integer **T (T ≤ 5)**. Hence **T** cases follow.

First line of the test case contains number of proposals **n (1 ≤ n ≤ 10^5)**. Hence **n** lines follow, each describing a proposal **x, y (-10^9 ≤ x, y ≤ 10^9** and **xy != 0)**.

Output

For each case print case number followed by space separated n numbers where i'th of these numbers denotes number of ways you can accept i proposals. Since the answer can be very big you need to present the solution in modulo **7340033**.

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

I Numbered Cards

Input: Standard Input
Output: Standard Output

You have **N** cards and each has an unique number between **1** and **N** written on it. In how many ways can you select a non-empty subset of the cards such that the number written on any two of your selected cards don't have any common digits? For example, when **N = 12**, **{1, 2, 3}** , **{2, 11}** , **{3, 4, 5, 6, 7, 8, 9, 12}** are some valid selections. But **{1, 2, 10}**,**{2, 5, 12}** are not allowed.

Input

The first line of the input contains an integer **T (T ≤ 15)** which is the number of test cases. Each of the following **T** lines denote a test case, containing an integer **N (1 ≤ N < 10^9)**.

Output

For each test case, output the case number followed by the number of subsets modulo 1000000007.

Sample Input	Output for Sample Input
2 3 12	Case 1: 7 Case 2: 1151

J The Hypnotic Spirals

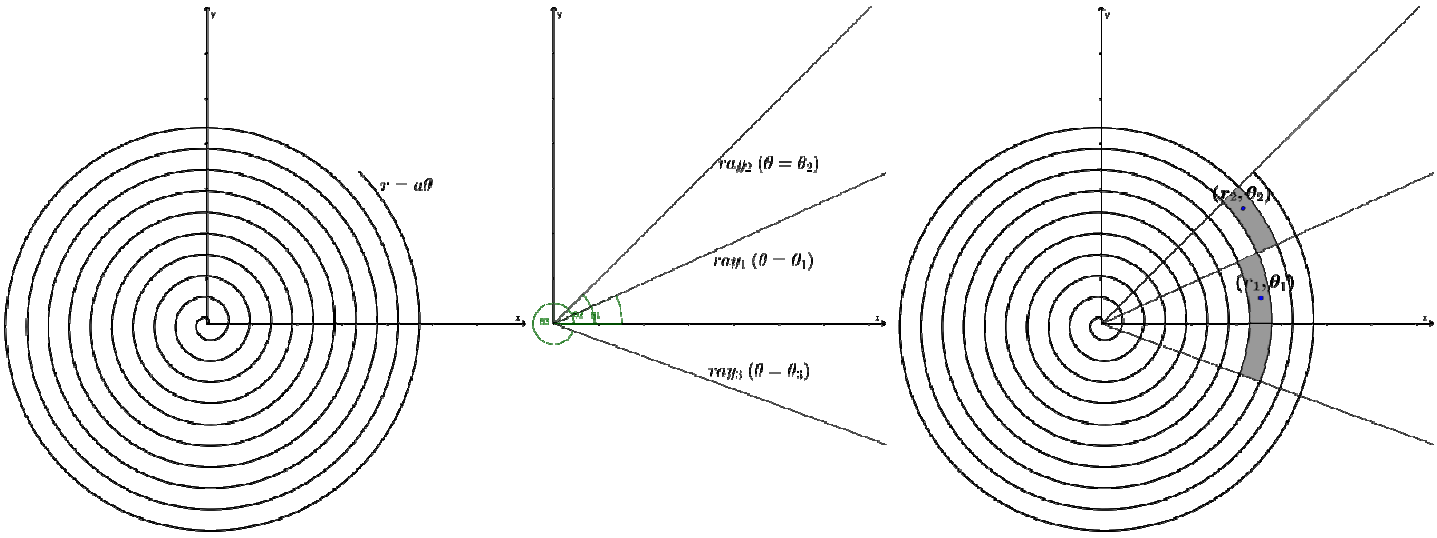
Input: Standard Input
Output: Standard Output

Professor Parinita Pratim Das, a brilliant Mathematician of this era, dreams of winning the Fields Medal someday for her patented Spirals that are custom made to hypnotize Engineers. To show that her ideas actually work she wants to paint her office room walls with various shapes of spirals.

Her method of drawing spirals goes something like this:

- She picks a spiral (in polar coordinates) of the form, $\mathbf{r = a\theta}$
- Then she adds n rays shooting out from the origin – the $\mathbf{i}^{th}$ ray being of the form $\boldsymbol{\theta = b_i}$ (positive or counter clockwise angle with the **x** axis)
- Finally she marks out **m** points $\mathbf{(r_j, \theta_j)}$ – if a point is in a bounded region then that entire region will be painted black.

The spiral on the left is one such example. The following figure illustrates the construction steps of this spiral.



Professor Das computes **r** in meters and knows that she needs **1** liter of black paint for per **10** meter^2 area on the wall. She needs your help to calculate how much paint she needs to buy.

Input

The input for this problem starts with **S (S ≤ 100)**, the number of spirals Professor Das wants to paint. Each of the hypnotic spirals is described in three lines. The first line of a spiral's description gives a floating point number giving the value of **a (0 < a < 10)**. The next line starts with the value of integer **n (0 ≤ n ≤ 10)**. Then follows **n** floating-point number θ_i, $0 \le \theta_i < 2\pi$, each giving the **θ** values for the **n** rays. You can assume that all the θ_i values are unique (Differs by at least **0.01**). The last line of spiral's description start with the value of **m (0 ≤ m ≤ 1000)**. The next **m** pairs of floating-point numbers, (r_j, θ_j), $0 < r_j < 100$, $0 \le \theta_j < 2\pi$, in the line gives the coordinates of the points which Professor Das mark out for painting. You can assume that none of the points will be on the spiral's curve or on any of the rays. Note that all the **θ** values in the input will be given in radians.

Output

For each test case print the spiral's number (in the format shown in the sample output). Then print the amount of black paint (in liters, rounded to **4** digits after the decimal point) needed to paint that spiral on Professor Das' wall. If the region to be painted is unbounded, print a **-1** instead. The amount of black paint will be such that an error of 5×10^{-8} will not change the printed output.

Sample Input	Output for Sample Input
2 1.00 2 0.43 0.78 1 11.06 0.63 1.00 0 2 17.05 0.55 17.59 0.74	Spiral 1: 2.2057 liters Spiral 2: -1