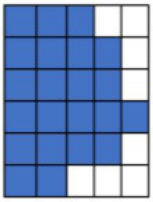
Problem A. Antivirus

Source file name:	antivirus.c, antivirus.cpp, antivirus.java, antivirus.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Eddy Cael Mamani Canaviri - Code Road Bolivia`

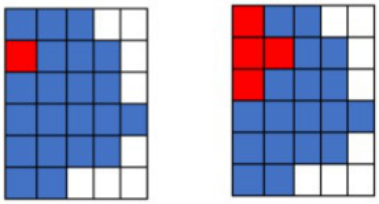
Machine Learning techniques aplies to a many fields quite know, incluiding the detection of computer viruses. In this ocassion we will analize the data produced by the C.C.C (Company of Capital Consumption).

The files of the C.C.C. are disposed in contiguous positions. In each position there is a set of cells with records. We can imagine these registers as a list of lists (List of files in this case).



It is well know that exists a virus that will infect the system, and this virus in particular has a very special way of infect archives. Using some heuristics has been detected that exists two versions of the virus (type A and type B). The virus starts to infect files from left to right, in other words, always starts with the register in the leftmost position and keeps moving to the right.

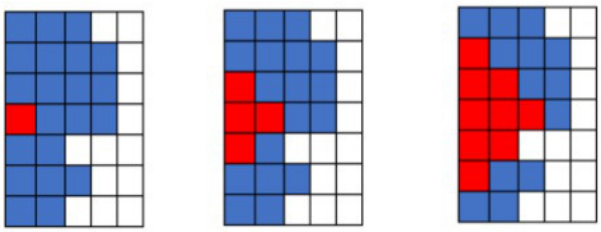
The virus type A selects an initial position i and starts to consume registers from that position and forms a kind of triangle (always form the largest possible triangle). We can show the virus' growth with the next graph: Be the starting position = 2 (The red cells are the infected registers).



Day 1 and day 2

The virus type A won't infect more registers because it cannot growth more (there are no more registers up, to extend the triangle).

Another example: Be the starting position = 4



Day 1, day 2 and day 3

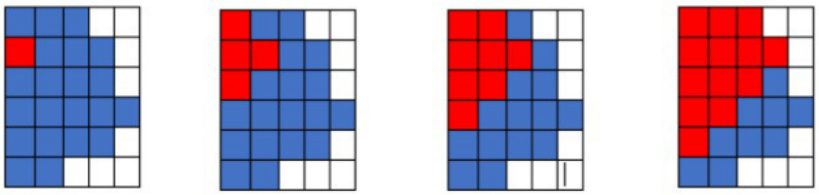
The virus type A will not infect more registers because there are no more free registers to the right.

In summary the virus type A tries to form a triangle centered in the initial position i, the triangle must be complete (no missing parts). Every file j located upper of position i must have an infected record less than the file in the position $j + 1$.

Analogously, each file j located down of the position i must have an infected record less than the file in the position $j - 1$.

The virus type B acts in a similar way, it selects an initial position i and starts to consume registers from that position and forms a kind of incomplete triangle (always forms the largest possible incomplete triangle).

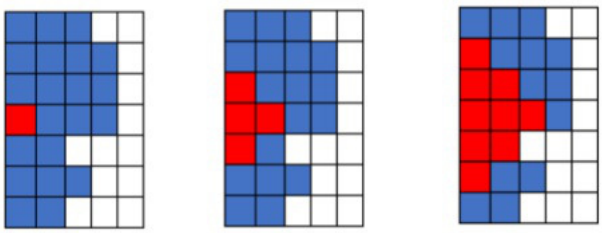
We can show the growth of the virus type B with the next graph: Be the initial position = 2 (red cells are infected records)



Day 1, day 2, day 3 and day 4

The virus type B will not infect more registers because there are no more free registers to the right.

Another example: Be the initial position = 4



Day 1, day 2 and day 3

The virus type B will not infect more registers because there are no more free registers to the right.

In summary, the virus type B tries to form an incomplete triangle centered in the initial position i. It is understood that the incomplete parts refer to the upper and bottom of the triangle. The corner that grows from left to right should always be complete. Every file j located upper of position i must have an infected record less than the file in the position $j + 1$.

Analogously, each file j located down of the position i must have an infected record less than the file in the position $j - 1$.

Knowing the way that the virus expands, we want to know how many records won't be infected in a certain range (we mean the range of infected files).

Input

The first line contains N ($1 \leq N \leq 5 * 10^5$), the number of files. The next line contains N integers r_i ($1 \leq i \leq N$ and $1 \leq r_i \leq 10^9$), the amount of registers of each file.

Next comes an integer Q ($1 \leq Q \leq 5 * 10^5$), the number of queries. After that, Q lines follows, specifying the queries in the following format:

- 1 i val = Replace the element in the position i ($1 \leq i \leq N$) with the value val ($1 \leq val \leq 10^9$)

- 2 i *type* = of the total of registers of the infected files, calculate how many registers won't be infected if the virus starts to infect the file i. A file is consider infected if it has as least one infected record.

For every query of type 2 you must assume that the system is not infected. The variable *type* indicates the virus type. If $type = 1$ the virus is of type A, in other case the virus is of type B.

In every case you must assume that the propagation will last at least r_i days (the virus will infect the maximum possible quantity of files).

Output

For every query of type 2 print a single line with two integers: the maximum quantity of infected registers (of the infected files), and the quantity of registers that are not infected. See the example input and output for more detail.

Example

Input	Output
12 3 4 5 5 6 2 5 4 3 2 1 1 9 2 6 0 2 6 1 2 4 0 2 4 1 1 6 6 2 6 0 2 6 1 2 4 0 2 4 1	2 9 2 9 4 14 4 14 6 8 6 8 5 14 4 18

Use fast I/O methods

Explanation

In the first query, the virus type is B, therefore it extends until infect all the records of file 6. Files 5, 6, and 7 are infected. One record of the file 5, two records of the file 6 and one record of the file 7 are infected. Thus $(6-1)+(2-2)+(5-1) = 9$ uninfected records remain.

In the second query, the virus type of type A growths in the same manner than the first query. In the third query, the virus of type B expands until consumes 4 records of the 4 file. The virus no longer progresses because there are no records to infect in the 6 file (its two registries have already been infected).

In the fourth query the virus of type A growths in the same manner than the third query.

In the fifth query we change the element in the position 6 with the value 6.

In the sixth query, files $[1 \ldots 11]$ get infected. The amount of infected records are: 1, 2, 3, 4, 5, 6, 5, 4, 3, 2, 1. The maximum is 6. 14 records remains uninfected.

In the seventh query, the virus A growths in the same manner than the last query.

In the eighth query, teh virus type B consumes all the records of the file 4. The files $[1 \ldots 8]$ get infected and the amount of infected records are: 2, 3, 4, 5, 4, 3, 2, 1. The maximum is 5. 14 records remains uninfected.

In the ninth query the virus type A infect 4 records of the 4 file. After that it can no longer expands. The maximum is 4. Files $[1 \ldots 7]$ are infected. 18 records remains uninfected.

Problem B. Beautiful Triad

Source file name:	beautiful.c, beautiful.cpp, beautiful.java, beautiful.py
Input:	`standard`
Output:	`standard`
Author(s):	`Gerson Yesid Lázaro - UFPS Colombia`

A **numerical triad** of limit N is a set of 3 numbers A, B and C where $0 \leq A, B, C \leq N$. A numerical triad of limit N is considered a **beautiful triad** in base K, if and only if all the pairs that can be formed between their values A, B and C differ by no more than K units.

For example $(4, 4, 6)$ is a beautiful triad in base 3 because the difference between A and B is 0, the difference between A and C is 2 and the difference between B and C is 2, all differences being less than 3. However, this is not a beautiful triad in base 1, because two of their differences are greater than 1.

Knowing N and K, can you tell how many different beautiful triads of limit N in base K can be formed? Note that $(4, 4, 6)$, $(4, 6, 4)$ and $(6, 4, 4)$ are three different triads.

Input

The first line of the input contains an integer T, the number of test cases. Each case contains two integers N and K as described previously ($0 \leq N \leq 2 * 10^9$, $0 \leq K \leq 1000$, $K \leq N$).

Output

Print one line per test case, the number of beautiful triads of limit N in base K that can be formed. It is guaranteed that this number fits in a 64 bits signed integer.

Example

Input	Output
5 0 0 1 0 1 1 2 1 2000000000 0	1 2 8 15 2000000001

Problem C. AirCraft: Monster

Source file name:	aircraft.c, aircraft.cpp, aircraft.java, aircraft.py
Input:	`standard`
Output:	`standard`
Author(s):	`Gerson Yesid Lázaro - UFPS Colombia`

AirCraft, the popular aviation video game has released its new version with more monstrous challenges than ever: "AirCraft: Monster", abbreviated by its fans as ACM. In each successful mission the players will receive an exact amount of experience points (XP), attack points (AP) and defense points (DP), which will accumulate in their profiles. The Missions in ACM doesn't have a specific order: The player can choose in which order to play them. A mission only gives points the first time it is surpassed, so it is not possible to win points twice with the same mission.

ACM awards medals for certain action. The most sought medal is the "Monster Player". To win this medal, the player must accumulate EXACTLY x experience points, a attack points and d defense points. However, some users have complained, they said that it's not possible to get exactly these scores with any possible combination of successful missions.

Given the x, a and d points needed to win the Monster Player Medal and knowing the XP, AP y DP that every mission awards, is it possible to win the medal, and therefore become a Monster Player?

Input

Input begins with an integer T, the number of test cases. For each case, the first line contains 4 integers x, a, d and m ($1 \leq x, a, d \leq 10^8$ and $1 \leq m \leq 30$), the number of experience points, attack points and defense points needed to win the medal and the number of missions of the game. Then m lines come, each one describing a mission of ACM. A mission consist of a string s (s containing no white spaces), the name of the mission, and 3 integers XP, AP and DP ($1 \leq XP, AP, DP \leq 10^7$), the points that the mission awards.

Output

For each test case prints a line containing "POSSIBLE" if its possible to become a Monster Player, or "IMPOSSIBLE" if there is no way to achieve the medal.

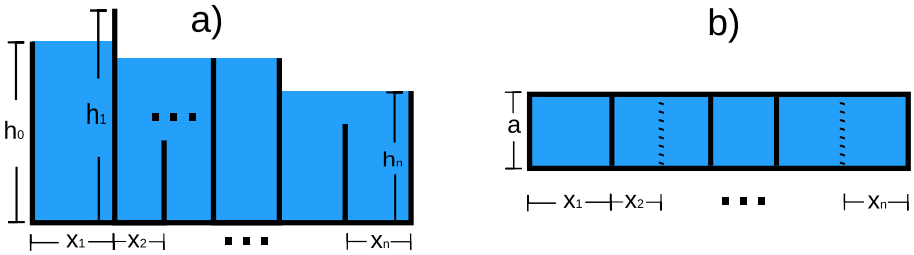
Example

Input	Output
2 100 100 100 5 MISSION1 30 10 40 MISSION2 40 70 30 MISSION3 40 10 20 MISSION4 20 20 50 MISSION5 10 50 90 100 100 100 3 a 10 30 10 b 10 10 40 c 10 60 50	POSSIBLE IMPOSSIBLE

Problem D. Drought In Nlogonia

Source file name:	drought.c, drought.cpp, drought.java, drought.py
Input:	`standard`
Output:	`standard`
Author(s):	`Gerson Yesid Lázaro - UFPS Colombia`

The meteorological service of Nlogonia forecast for the next months the worst drought in Nlogonia's history. This is why his mayor has decided to install the largest tank in the history of the planet, to maintain water supplies to enable them to face the crisis.



a) Frontal view of the tank. b) Aereal view of the tank (Dotted lines are submerged walls)

The tank is a big box, with n consecutive compartments as you can see in the previous picture. The walls of some compartments may be completely submerged. The mayor wants to know how many cubic centimeters of water will fit in the tank before filling it, since he does not want a single drop to be spilled. Can you help him?

Note that the front and back walls (not visible in the drawing) are high enough so the water never reaches its upper limit, so if a drop spills, it will spill over the wall at height h_0 or h_n. Similarly, the walls were built in a super thin material, so its volume is negligible and you can ignore it.

Input

The first line of the input contains an integer T, the number of test cases. Each case begins with a line with two integers a and n, separated by spaces. The next line has n integers $x_1, x_2, ..., x_n$, separated by spaces, the distance of each intern wall of the tank to the previous wall (the width of each compartment). The last line of each test case has $n + 1$ integers $h_0, h_1, ..., h_n$ separated by spaces, the height of every intern wall. All the measures are given in meters ($1 \leq a, n, x_i, h_i \leq 10^4$).

Output

Print one line per test case, an integer with the maximum capacity of the tank in cubic meters. It is guaranteed that this number fits in a 64 bits signed integer.

Example

Input	Output
1 10 6 10 6 6 8 8 7 20 24 8 18 18 12 15	7850

Use fast I/O methods

Problem E. My Password is a Palindromic Prime Number

Source file name:	password.c, password.cpp, password.java, password.py
Input:	`standard`
Output:	`standard`
Author(s):	`Milton Jesús Vera - UFPS Colombia`

Gauss' is a clever guy who is a little obsessed with information security, that's why he is always looking for the most secure password. After a long research Gauss is choosing for his password a palindromic prime number *ppm*, which is ultra secure, the only problem is that he always forgets it. As a help to his bad memory, Gauss wants to write a file with the decimal number produced by the division of one (1) and his password, $1/ppm$. Gauss knows that the result is always a pure recurring (periodic) decimal number, and therefore he only wants to save the number with its period. For example:

Gauss' Password (PPM)	Decimal Number 1/PPM	Length 1/PPM (period)
101	0,0099	4
757	0,00132100396301188903566 7107	27
191	0,00523560209424083769633507853403141361256544502617801047120418848167539267015706806282722513089	95
11311	0,00008840951286358412165148970029175139244982760144991601096277959508443108478472283617717266377862257978958535938466979046945451330563168596941030854919989390858456369905401821235964989832906020687826010078684466448589868269825833259658739280346565290425249756873839625143665458403324197683670762974096012730969852356113517814516842012200512775174608787905578640261692158076209	377
14341	A number of 14340 decimals: 0,0000697301443413987866954884596611114985008018966599260860469981172861027822327592218115891499895404783487901819956767310508332752248 79...	14340
79997	A number of 39998 decimals	39998

Having the file, Gauss knows that he can execute an algorithm to remember his password. Can you help Gauss to write the file?

Input

The first line of the input contains an integer t, the number of test cases, followed by t lines, each line contains a palindromic prime number n ($10 < n < 10^5$).

Output

For each test case print the pure recurring decimal number that will help Gauss to remember his password. (Use . as a decimal point)

Example

Input	Output
2 101 757	0.0099 0.00132100396301188903566710 7

Use faster I/O methods

Problem F. Macarons in La Fête

Source file name:	macarons.c, macarons.cpp, macarons.java, macarons.py
Input:	`standard`
Output:	`standard`
Author(s):	`Angie Melissa Delgado - UFPS Colombia`

La Fête de la Musique is an international celebration that takes place on June 21, its aim is to promote music in two ways: The first that amateur musicians voluntarily go out to play in the street. The second is with the organization of free concerts, in which the public has the opportunity to witness their favorite artists regardless of style or origin.

Pierre is an excellent $confiseur$ who knows about the popularity of this festival so he wants to take advantage of it and sell some deserts. For this year's festival Pierre plans to sell his delicious $Macarons$. $Macarons$ are sweet meringue-based confection made with egg white, icing sugar, granulated sugar, almond powder or ground almond, and food colouring. $Macarons$ come in different flavors and types and Pierre has mastered two types of them: $Traditional\ Macarons$ and $Glazed\ Macarons$.

For the festival Pierre will bake n trays of $Macarons$ each tray of an unique flavour containing up to m $Macarons$. Each one of this trays will be of $traditionals\ Macarons$ or $glazed\ Macarons$, but as everybody knows $Macarons$ are very difficult to make and $glazedMacarons$ are even worst and Pierre doesn't want to work more unnecessarily.

Pierre already knows the preferences of his m clients and he wants to make them all happy. Pierre's clients will be happy if for each client he prepares at least one of his favourites $Macarons$. Each client has a list of favorites $Macaron's$ flavours, and at most 1 of them will be a $glazed\ Macaron$.

Pierre wants to make all his clients happy but also he want to make the least possible trays of $glazed$ $Macarons$ he can. Knowing the Pierre's clients preferences can you tell which $Macarons$ flavours will be $glazed$ or if it is not possible to make all the clients happy?

Input

The first line of the input contains an integer tc, the number of test cases.

Each test case begins with a line containing two integers n and m ($1 \leq n, m \leq 2000$) the number of Macaron's flavours and the number of clients. The next line contains n strings $s_1, s_2, s_3, ..., s_n$, the name of each Macaron flavour that Pierre will prepare(s whitout blank spaces). Following that, there are m lines, one for each client preferences. Following that, there are m lines, one for each client's list of preferences. Each client's list of preferences begins with a number a ($1 \leq a \leq n$) the amount of preferences he has. Next there are a preferences separated by a blank space, each preference is formed by one integer t and one string f separated by a blank space, denoting the type and the flavour of the Macaron. The type of each preference can only be 1 if it is $glazed$ or 2 if it is $traditional$. It is guaranteed that the lenght of each string does not exceed 100 characters and that the flavour of the preference will be prepared by Pierre.

Output

For each test case if it is possible that Pierre makes all his clients happy print one line with the amount of glased trays that Pierre has to do, followed for the list of the glased flavours in alphabtical order, one per line. If it's not possible, print $Impossible$ instead.

Example

Input	Output
3 5 4 grains-de-poivre au-citron au-chocolat a-la-myrtille a-la-vanille 1 1 au-citron 3 2 a-la-vanille 2 au-chocolat 1 a-la-myrtille 3 1 a-la-vanille 2 a-la-myrtille 2 au-chocolat 2 2 au-citron 1 grains-de-poivre 3 3 au-chocolat au-caramel au-miel 2 2 au-miel 2 au-caramel 1 1 au-caramel 1 2 au-caramel 5 3 au-chocolat au-caramel au-miel a-la-vanille a-la-rose 2 2 au-caramel 2 au-miel 3 2 au-miel 2 a-la-vanille 2 a-la-rose 2 2 au-chocolat 2 au-caramel	2 au-citron grains-de-poivre Impossible 0

Use faster I/O methods

Explanation

In the second test case, it is impossible to make all his clients happy beacuse client 2 wants a glazed Macaron au-caramel and client 3 wants a traditional Macaron au-caramel and Pierre only do 1 tray of every flavour.

Problem G. Geonosis

Source file name:	geonosis.c, geonosis.cpp, geonosis.java, geonosis.py
Input:	`standard`
Output:	`standard`
Author(s):	`Angie Melissa Delgado - Marcos Javier Alvarez - UFPS Colombia`

After years of planning, the Galactic Empire has built the planetary fortress of Geonosis, a perfect prison for the leaders of the rebelion and a special place reserved for the evil Luke Skywalker. However, as it has been demonstrated with the failures of Death Star I and II, the fortress is not perfect and the evil Rebel Alliance plans to use a weakness detected in the communication towers surrounding Geonosis.

To achieve their evil plans, the Rebel Alliance has stolen a map with the structure of the fortress. Geonosis has a communication complex formed by n towers. Each pair of the complex's tower is connected by power lines that send messages at a w cost of energy.

The rebels want to build the Rogue Two ship to destroy the fortress and rescue its prisoners. To achieve that, they have to destroy all the towers of the fortress. They have discovered that the power necessary to destroy a tower is equal to the sum of the minimum energy needed to send messages between each pair of the fortress towers, either directly or indirectly through other towers. In other words, if we define $d(u, v)$ as the minimum energy to send a message from the tower u to the tower v, the power needed to destroy one tower will be:

$$\sum_{v,u,u!=v} d(u, v)$$

Note that once a tower is destroyed, it stops counting for this formula.

On the other hand, the commander of the Rogue Two is very capricious so he wants to destroy the towers in a given order. Knowing the information of the communication towers in Geonosis and the order in which each tower will be destroyed, can you tell what is the power needed for the Rogue Two to complete it's mission?

Input

The first line of the input contains an integer t, the number of test cases.

Each test case begins with a line containing an integer n ($1 \leq n \leq 500$) the number of towers in the fortress. Next n lines contain n integers each, the j_{th} number in the i_{th} line w_{ij} ($1 \leq w_{ij} \leq 10^4$ $w_{ii} = 0$) represents the amount of energy that cost to send a message from tower i to tower j.

Each test case ends with a line containing n distinct integers: $x_1, x_2, ..., x_n$ ($0 \leq x_i < n$), the order in which the towers will be destroyed.

Output

For each test case print the amount of energy that Rogue Two will need to destroy the Geonosis fortress.

Example

Input	Output
3 3 0 35 58 12 0 5 1 2 0 0 1 2 3 0 9982 1413 8327 0 5612 3577 7893 0 1 0 2 4 0 50 10 30 4 0 23 58 2 1 0 5 67 24 25 0 0 3 1 2	96 41118 287

Use faster I/O methods

Explanation

In the first test case, the energy needed to destroy the tower 0 is 89, to destroy the tower 1 is 7, to destroy the tower 2 is 0. So, the total energy needed by Rogue Two is 96.

Problem H. How many inversions?

Source file name:	howmany.c, howmany.cpp, howmany.java, howmany.py
Input:	`standard`
Output:	`standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`

Humbertov Moralov in his student days, he is attended system engineering at "University of missing hill". He was evaluated in its first course of Analysis of Algorithms (at the first half of 1997) with the following topics and questions:

Inversions :

Let $A[1 \ldots n]$ an array of distinct integers of size n. If $i < j$ and $A[i] > A[j]$, then the pair (i, j) is called an **inversion** of A.

Given the above definition about an inversion, *Humbertov Moralov* must answer the following questions:

1. List all inversions in $\langle 3, 2, 8, 1, 6 \rangle$.
2. What array of size n, with all the numbers from the set $1, 2, 3, \ldots, n$ has the largest amount of inversions? How many inversions?
3. Write an algorithm to determine the number of inversions in any permutation of n elements with $\theta(n \log n)$ in the worst case run time.

Humbertov Moralov answered questions 1. and 2. without any problem, but he was not able to solve the question 3 at time. Days later he thought the following solution:

```
1: inv ← 0
2: function MERGE(A[], p, q, r)
3:     n1 ← q − p + 1
4:     n2 ← r − q
5:     create arrays L[1 ... n1 + 1] and R[1 ... n2 + 1]
6:     for i = 1 to n1 do
7:         L[i] ← A[p + i − 1]
8:     end for
9:     for j = 1 to n2 do
10:        R[j] ← A[q + j]
11:    end for
12:    L[n1 + 1] ← ∞
13:    R[n2 + 1] ← ∞
14:    i ← 1
15:    j ← 1
16:    for k = p to r do
17:        if L[i] ≤ R[j] then
18:            A[k] ← L[i]
19:            i ← i + 1
20:        else
21:            A[k] ← R[j]
```

```
22:             j ← j + 1
23:             inv ← inv + n1 − i + 1
24:         end if
25:     end for
26: end function
27: function MergeSort(A[], p, r)
28:     if p < r then
29:         q ← ⌊(p + r)/2⌋
30:         MergeSort(A[], p, q)
31:         MergeSort(A[], q + 1, r)
32:         Merge(A[], p, q, r)
33:     end if
34: end function
```

Will this code solve the problem? Just adding the lines 1 and 23 will be enough to solve the problem?

Please help *Humbertov Moralov* to validate this solution! For this, you must implement this solution in any of the programming languages accepted by the ACM-ICPC and verify if the expected results are generated.

Input

The input may contain several test cases. Each input case begins with a positive integer n ($1 \leq n \leq 10^6$) denoting the length of A, followed by n distinct lines. Each line contains a positive integer from array A. For $i \in [1, n]$, will meet that $0 \leq A[i] \leq 10^8$. The input ends with a test case in which n is zero, and this case must not be processed.

Output

For each test case, your program must print a positive integer representing the total number of inversions in the array A. Each valid test case must generate just one output line.

Example

Input	Output
5 3 2 8 1 6 5 5 4 3 2 1 1 10 0	5 10 0

Use faster I/O methods

Problem I. Quidditch Match

Source file name:	match.c, match.cpp, match.java, match.py
Input:	`standard`
Output:	`standard`
Author(s):	`Gerson Yesid Lázaro - Angie Melissa Delgado - UFPS Colombia`

Quidditch is the most popular sport among the students of Hogwarts School of Witchcraft and Wizardry. Quidditch is played between two teams of N players each, in an oval or rectangular stadium (This season, the oval stadium is under repair, and therefore all the matchs are played always in the rectangular one).

If you know Harry Potter, you may know a lot of things about this game: its rules, its punctuation system, the positions of the players and that while Harry is on the ground, his team Gryffindor will win to Slytherin (come on, we all know that Harry always defeats Malfoy).

Madame Hooch is the flight instructor and Quidditch referee in Hogwarts. Although she admires Harry's skills in Quidditch she is tired of seeing him win always. For this reason, she has decided to create an alternative system of punctuation, in which at least on some occasions, Slytherin could win Gryffindor.

The game as it is well know ends when the seeker catchs the snitch. In that moment, Madame Hooch registers the position of each player on the plane of the stadium (Although the players compete flying over their brooms, in the scheme of Madame Hooch the height to which they fly is despised and only takes into account their location in the plane in two dimensions).

In the Madame Hooch's system, the team who *dominates* the most part of the field when the match ends wins the game. She consider that all the players fly at the same speed and therefore, each player dominates the region that surround him and which has all the points of the field that are most close to him than to any other player, because flying at the same speed, he could reach to any point of his region before any other player.

Under this idea, the 2D map of the field is divided into $2N$ regions, one for each player. Then Madame Hooch sums the area of the regions belonging to each team, declaring winner to that team that dominates the greater area of the Quidditch stadium.

However, doing this calculation is really tedious, and since Madame Hooch has not found an spell that calculates the winner, she has decided to appeal to muggle programming. Can you help her?

Input

The first line of the input contains an integer T, the number of test cases. Each case begins with a line with 4 integers X_0, Y_0, X_f and Y_f ($0 \leq X_0, Y_0, X_f, Y_f \leq 100$; $X_0 + 2 \leq X_f$, and $Y_0 + 2 \leq Y_f$), the coordinates of the lower left point and the upper right point in the Cartesian plane. The next line contains an integer N ($1 \leq N \leq 10$), the number of players of each team. Following there are N lines with the integers x and y ($X_0 < x < X_f$ and $Y_0 < y < Y_f$), the coordinates of each Gryffindor's player. Then N more lines come, with two integers x and y each ($X_0 < x < X_f$ and $Y_0 < y < Y_f$), the coordinates of each Slytherin's player.

All the coordinates given are integers values. However, keep in mind that the regions defined for each player may have decimal vertices.

Output

Print a single line for each case, the name of the winner team ($Gryffindor$ or $Slytherin$) according to Madam Hooch's calculation. It is guaranteed that there will be no draws.

Input	Output
1 0 0 2 8 3 1 1 1 2 1 6 1 4 1 5 1 3	Gryffindor

Problem J. The Robot's Grid

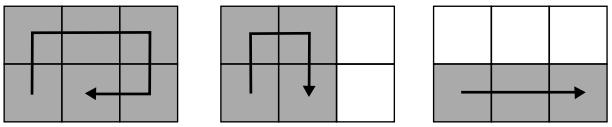
Source file name:	robot.c, robot.cpp, robot.java, robot.py
Input:	`standard`
Output:	`standard`
Author(s):	`Gerson Yesid Lázaro - UFPS Colombia`

A robot travels over a luminous grid. Initially the grid is off and every time the robot walks over a cell, this cell turns on a light that keeps lighted while the robot keeps moving.

The robot can only make to actions:

- Take a step forward
- Turn 90 degrees clockwise and advance one step forward.

The robot always starts from the lower left corner facing the top of the grid. It can do any sequence of movements a and b (it can make many consecutive a movements, or many consecutive b movements, or alternate them in any way) as long as it does not leaves the grid and don't pass over a previously lighted cell (ie, it can only pass once over every cell). Whenever it's possible, the robot will keep moving, stopping only when he reaches a cell where he can't make more movements, ending the route. How many different routes can the robot make over the grid?



Example for a grid of 2 rows and 3 columns, where 3 different routes can be done

Input

The first line of the input contains an integer T, the number of test cases. Each case contains two integers R and C, the number of rows and columns of the grid ($2 \leq R \leq 25$, $1 \leq C \leq 25$).

Output

For each case print a line with the number of different routes that the robot can make.

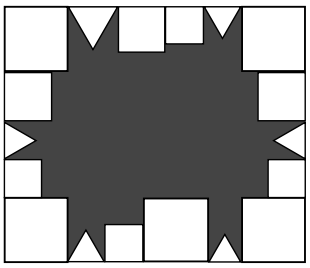
Example

Input	Output
4 2 1 2 2 2 3 3 3	1 2 3 6

Problem K. Polygonal Park

Source file name:	park.c, park.cpp, park.java, park.py
Input:	`standard`
Output:	`standard`
Author(s):	`Milton Vera - Melissa Delgado - Gerson Lázaro - UFPS Colombia`

When Polygonal Town was built, in its center it was left a rectangular space to built a park. However, there was a serious problems in the construction and many houses were built over the park space. More exactly, every limit of the park was invaded with two types of houses: Houses with square base and houses with equilateral triangle base, as you can see in the following picture (The white squares and triangles are houses).



The mayor of Polygonal Town has decided to left the houses in their actual location and build the park in the free space left (the grey area in the picture). Can you calculate the area of the park?

It is guaranteed that in the four corners of the park are square houses, that the consecutive houses are always together and that no house overlaps other.

Input

The first line of the input contains an integer T, the number of test cases. Each case begins with an integer N ($4 \leq N \leq 3100$), the number of houses located in the original park space. Next there are N lines, each one representing a house. The first line of this N lines will represent the house located at the upper left corner and all the other houses will appear clockwise. Each one of the lines that represents a house contains a character C and a number K ($1 \leq K \leq 1500$). The character C can has the value of 'S' if it is a square house, 'T' if if its a triangle house or 'C' if it is a corner house (remember that if the house is corner, it will be a square house). The number K is the length in meters of one side of the house.

Output

For each test case prints a single line with the area of the park in meters. This number has to have four digits after the decimal point.

Example

Input	Output
1 8 C 4 T 3 C 3 S 2 C 4 S 2 C 4 T 1	20.6699

Problem L. Problem with a ridiculously long name but with a ridiculously short description

Source file name:	problem.c, problem.cpp, problem.java, problem.py
Input:	`standard`
Output:	`standard`
Author(s):	`Gerson Yesid Lázaro - UFPS Colombia`

All of us hate looooooong descriptions, so here goes one short: Calculate $(66^n) mod 100$ for the given n.

Input

First line of input contains T, the number of test cases ($T \leq 5000$). Then, there are T lines, one for each case, containing n ($1 \leq n \leq 10^{1000}$).

Output

Print one line per case, the solution of $(66^n) mod 100$.

Example

Input	Output
4 0 1 2 99999999999999999999	1 66 56 36