HKUST Local Contest 2014
13 Sep 2014

Organized by: Prof. Ke Yi and Mr. Ng Tsz Yeung Ken

Problemsetter: Mr. Ng Tsz Yeung Ken

Judges: Mr. Ng Tsz Yeung Ken and Mr. Yin Zhu

Contest Time: 1:30pm - 5:30pm, 13 Sep 2014

Latest updated: 15th Sep, 2014.

Contest Rules and Regulations:

1. **This contest is an individual contest. Discussions between contestants are strictly prohibited.** Sanctions will be imposed on contestants if they are found to have violated the regulations governing integrity and honesty.
2. In this contest, **the contestants are given six programming problems**. The goal is to solve as many problems as possible. For those who solve the same number of problems, the one with lower score wins. (The scoring system will be explained below.)
3. **The programming languages to be used in this contest are C++ and Java.** The contestants use PC^2 to submit their source codes to the judge and the source codes are compiled by Visual Studio C++ 2010 or JDK 7.
4. **The contestant should read the input and write the output via standard I/O.** The contestants can assume that all test cases are of the format as stated in the problem statements. i.e. No exception handling is needed.
5. The correctness of each submission is judged by inputting test cases into the submitted program. The submission is regarded as correct if its outputs match completely with the model outputs. The submission is judged as correct or wrong. **No partial credit is given.**
6. The contestants can re-submit another source code after previous wrong submissions.
7. **All programs should not run for more than the time limit specified in the problem** (in most cases a "correct" implementation will run far less than the time limit we provide).
8. **The contestants are ranked firstly by the number of problems solved, and secondly the total time spent on solving the problems.** Time spent on solving one problem is the time between the start of contest and the submission of the correct implementation of that problem. For each problem you solved, a penalty of 20 minutes will be added to your score for each wrong submission of that problem.
9. Java API documentation and C++ STL documentation are provided. **The contestants are allowed to bring any hard copies of books, notes, references, dictionaries and sketch papers to the contest site.** Electronic devices are forbidden.

Problem A. Amidakuji

Input file:	`Standard Input`
Output file:	`Standard Output`
Time limit:	3 seconds
Memory limit:	256MB

There is a Japanese game called "Amidakuji" (or "Ghost leg" in English), it consists of N vertical lines and M horizontal lines that join two adjacent vertical lines.

The "Amidakuji" configuration can be described by a "Line sequence" S_i, $1 \leq i \leq M$:

1. Sort the M horizontal lines in decreasing order of height.
2. If the i-th highest horizontal line is joining the j-th and $j+1$-th vertical lines ($1 \leq j < N$), then $S_i = j$.

You may assume that the height of all horizontal lines are different and no horizontal line will join any end points of a vertical line.

The game is to determine the outcomes of all vertical lines, the rules are described as follows:

1. For each vertical line, start at the top of the line.
2. Follow the line downwards until you the end of a vertical line is reached or a horizontal line is encountered.
3. If the end of a vertical line is reached, that vertical line is the outcome of the given line;
4. Otherwise, cross the horizontal line to get to the other vertical line and repeat 2.

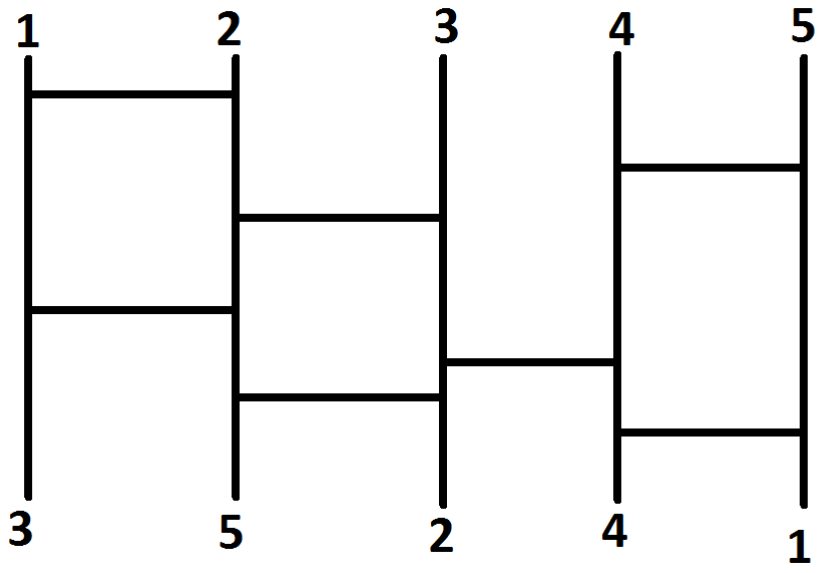


Figure 1: There are 5 vertical lines and 7 horizontal lines, the Line sequence is "1 4 2 1 3 2 4" and the outcome is "3 5 2 4 1".

The game can be played several times by **starting over again with the same configuration after the end of a vertical line is reached**.

Ken loves playing "Amidakuji", he even tried to play this game for $T - 1$ times.

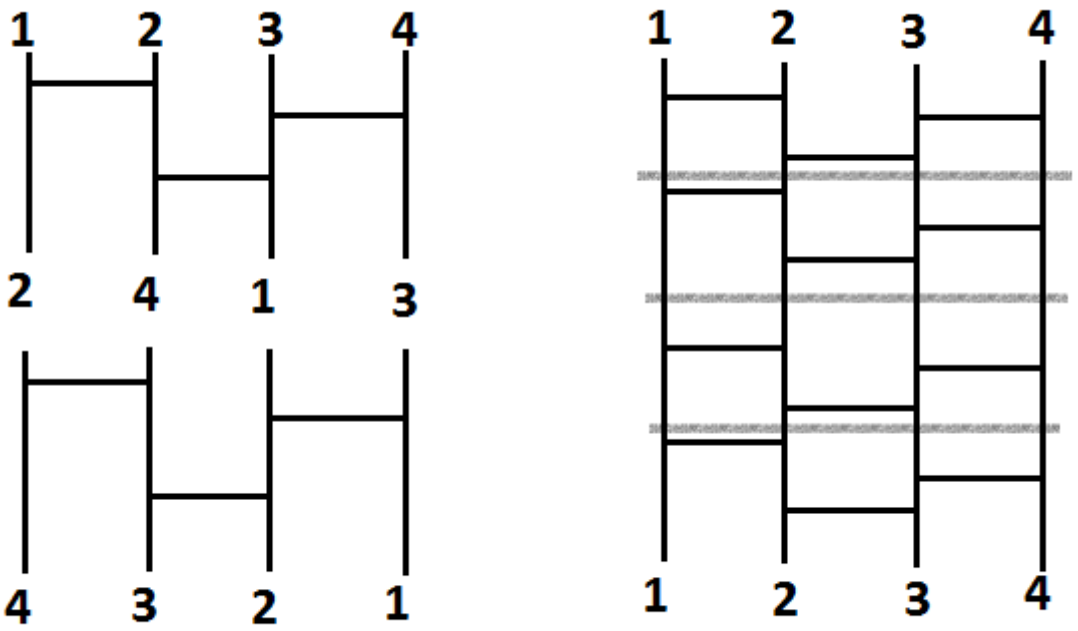


Figure 2: The left game is played twice and the right game is played 4 times.

Ken always asks you to play with him and you are **DONE**. You want to reveal the outcomes just after Ken gives you the Line sequence and leave him alone forever.

So, you are given a Line sequence and you have to compute the outcome of all vertical lines after playing for $T-1$ times.

Input

The first line of the input is the number of test cases $K \leq 15$.

Each test case has the following format:

$N \quad M \quad T$

$s_1 \quad s_2 \quad \ldots \quad s_M$

$2 \leq N \leq 1000$, $0 \leq M \leq 100000$, $1 < T \leq 10^{500}$.

Output

For each test case, output the outcome of all N vertical lines after playing for $T-1$ times.

Example

Standard Input	Standard Output
4 5 7 2 1 4 2 1 3 2 4 4 3 2 1 3 2 4 3 3 1 3 2 4 3 5 1 3 2	3 5 2 4 1 2 4 1 3 4 3 2 1 1 2 3 4

Problem B. Belle Mary et Macaron

Input file:	`Standard Input`
Output file:	`Standard Output`
Time limit:	10 seconds
Memory limit:	256MB

Mary is a famous French celebrity and Macaron is a world-known French confection. Ken loves them very much and now he is addicted to the mobile game 'Belle Mary et Macaron' (abbrev. M&M).

The game begins with a 5×6 grid, each grid on the board has either a Macaron or Mary on it, each Macaron is represented by a number from 1 to 9 and there is one and only one Mary on the board.

There are 2 operations that a player can take: **'Move'** and **'Teleport'**. For **'Move'** operation, you can swap Mary with a Macaron that is above, below, left to her or right to her, i.e. the four directions; for **'Teleport'** operation, you can swap Mary with a Macaron at **ANY** position. **'Teleport'** can be used ONLY ONCE.

Players are allowed to take exactly M operations. After that, the system will count the score as follows:

1. For each three consecutive horizontal/vertical Macarons with the same number, they will be marked. Each Macaron can be marked at most once.
2. For each marked Macaron, 1 score will be added and it will be erased from the board. Mary and Macarons that are above it will fall and fill the emptied grid.
3. Repeat 1 and 2 until no Macaron is marked.

Ken wants to know the operation sequence (i.e. the M operations in sequential order) that can maximize the scores given a board configuration. Unfortunately, Ken writes buggy programs and fails to find out the optimal solution. Assistances from an experience programmer like you is crucial for him. If there are multiple of them, give Ken the one that is lexicographically smallest.

Input

The first line of the input is the number of test cases $K \leq 10$.

Each test case has the following format:

$$
\begin{array}{cccc}
M & & & \\
a_{11} & a_{12} & \dots & a_{16} \\
a_{21} & a_{22} & \dots & a_{26} \\
\vdots & \vdots & \ddots & \vdots \\
a_{51} & a_{52} & \dots & a_{56}
\end{array}
$$

$1 \leq M \leq 7$, a_{ij} will be either a number from 0 to 9 that represents a Macaron, or a letter 'B' that represents Mary.

Output

For each test case, output 1 or 2 lines. The first line is the maximum score that can be obtained. If $M \neq 0$, output the second line which represents the corresponding operations in sequential order.

For **Move** operation, output 'U', 'D', 'L' and 'R' to represent moving upward, downward, to left and to right respectively.

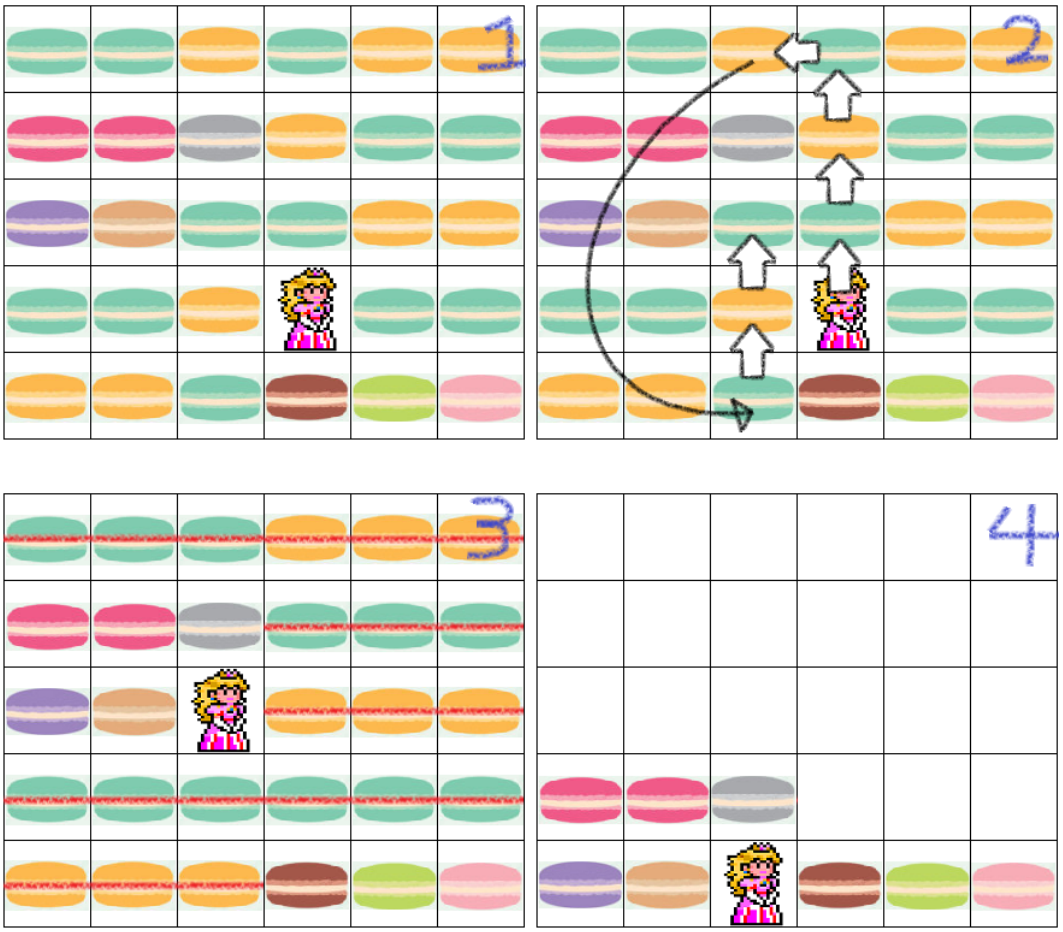
For **Teleport** operation, output 'Trc' where r and c are numbers that represent the target row and column respectively.

Output the operations consecutively. If there are multiple solutions, output the lexicographically smallest one.

Example

Standard Input	Standard Output
3 7 112122 995211 341122 112B11 221678 0 1011B1 100110 101011 010101 111001 7 101101 100110 100B11 011101 111001	21 UUULT53UU 28 29 UURRT23UL

Here is an illustrative explanation for sample test case 1.



Problem C. Crazy typists

Input file:	`Standard Input`
Output file:	`Standard Output`
Time limit:	4 seconds
Memory limit:	256MB

Ken is a speed typist, he types really fast and he loves competing with other typists on a website regularly.

One day, Ken was competing as usual and he won all of the matches until he met the crazy typist "Derek", Ken tried to challenge "Derek" but he could not win even a single match against the crazy typist "Derek". "Kid, you are TOOOO young," this is the last sentence from "Derek". Ken wants to beat "Derek", and he wants to improve his typing speed.

Normally, Ken types a letter in M unit time. For a specific set of words, he may type faster or slower. You are given M, a passage S and a set of N distinct words such that Ken can type each word w_i in t_i unit time. For simplicity, assumed that a passage contains only uppercase letter('A' to 'Z').

Ken is asking you to compute the minimum time that he needs to type the whole passage.

Input

The first line of the input is the number of test cases $K \leq 10$.

For each test case, the input format will be the following:

$$\begin{array}{cc} S & M \quad N \\ w_1 & t_1 \\ w_2 & t_2 \\ \vdots & \vdots \\ w_n & t_n \end{array}$$

$1 \leq t_i, M \leq 1000$, $1 \leq N \leq 40$, $1 \leq |w_i| \leq 16$, $1 \leq |S| \leq 10^6$. All words are distinct.

Output

For each test case, output the minimum time that Ken needs to type the whole passage.

Example

Standard Input	Standard Output
2 KENISSOHANDSOME 1 9 AND 2 SOME 2 OH 1 ME 1 SO 1 ENI 2 SS 1 KE 1 NI 1 ACRAZYMAN 3 3 AC 1 WA 1 TLE 999	8 22

Problem D. Dominating cows

Input file:	`Standard Input`
Output file:	`Standard Output`
Time limit:	3 seconds
Memory limit:	512MB

There is a farm, John is the owner and he has N cows.

Everyday, farmer John takes good care of his cows, he also measures the height h_i and weight w_i of his cows. He finds an interesting fact that the height and weight of his cows are i.i.d. random variables drawn from uniform distribution with minimum = 1 and maximum = 10^7.

One day, the cows assemble and chitchat to each others, they may even hold some contests. Tonight, they want to find out all dominating cows.

Let A and B be two cows. A dominates B if and only if A and B do not have same height and width, and B is not taller than A and B is not heavier than A. A is a dominating cow if and only if A is not dominated by any other cows in the farm.

The cows are serious but, well, they do not have fingers to count. Please help them to find out all dominating cows.

Input

The input will have the following format:

$$\begin{array}{ll} N & \\ h_1 & w_1 \\ h_2 & w_2 \\ \vdots & \vdots \\ h_N & w_N \end{array}$$

h_i and w_i are the height and weight of the i-th cow respectively. $1 \leq h_i, w_i \leq 10^7$. $1 \leq N \leq 2000000$.

Output

Print two lines. The first line contains a single number M which is the number of dominating cows. The second line contains M numbers which are the indices of dominating cows, the number should be printed in ascending order of indices.

Example

Standard Input	Standard Output
4 3 0 0 2 2 3 3 3 1 1 2 1 1 2 2 1 1 1 1 6 1 1 2 2 3 1 4 2 5 1 6 2	1 3 3 1 2 3 2 1 2 3 2 4 6

The examples are artificial in order to address any ambiguity. It is guaranteed that the data follows uniform distribution ranged $[0, 10^7]$.

Problem E. Emon's Dorayaki

Input file:	`Standard Input`
Output file:	`Standard Output`
Time limit:	2 seconds
Memory limit:	256MB

Emon is a Japanese guy who loves eating Dorayakis (a Japanese bread with red bean inside), they can be made using N different raw ingredients.

There are many raw ingredients, each of them is associated with a unique positive integer. All of the raw ingredients can be bought on the market, they can also be mixed using two different raw ingredients. Let A and B be positive integers representing two different raw ingredients, a new raw ingredient $C = A \oplus B$ can be created, where $\oplus$ represents the bitwise XOR operation.

So you can buy part(or all) of them from the market, then mix the rest using the raw ingredients you bought and make Dorayakis. For example, if raw ingredients 1, 2 and 3 are needed, Emon can buy 1 and 3, then create 2 by using 1 and 3. So buying 2 distinct raw ingredients is sufficient for him to make Dorayakis.

Obita, the closest friend of Emon, wants to know the minimum number of distinct raw ingredients that Emon needs to buy from the market.

Obita passes no maths test. Please help him.

Input

The first line of the input is the number of test cases $K \leq 50$.

Each test case has the following format:

N

$r_1 \quad r_2 \quad \ldots \quad r_N$

$1 \leq N \leq 50$. $1 \leq r_i \leq 10^9$ are the raw ingredients that Emon needs.

Output

For each test case, output the minimum number of distinct raw ingredients that Emon needs to buy.

Example

Standard Input	Standard Output
2 3 1 2 3 3 1 7 3	2 3

Problem F. Farmer John

Input file:	`Standard Input`
Output file:	`Standard Output`
Time limit:	4 seconds
Memory limit:	256MB

It is almost open day of John's farmland. In order to introduce his lovely cows to the public, he is setting up a field. The field is set using fences and N railings on a flatland. He puts each railing at a location (x_i, y_i) and connect them one by one using the fences.

John loves his cows so much and he wants to let his cows walk freely in the field. Therefore, he puts the railings in a way such that for any cows in any location of the field, it can walk towards any other location of the field straightly. i.e. the field is a 2D convex polygon.

His cows doubt if John has set up the field in the most effective way. The effectiveness of a railing is the square of Euclidean distance from it to the farthest railing, the effectiveness of a field is the sum of the effectiveness of all railings.

Farmer John is the legend and he knows everything, but the cows do not want to upset John if the result is bad. So they want you to compute the effectiveness of the field.

Input

The first line of the input is the number of test cases $K \leq 10$.

Each test case has the following format:

N
$x_1 \quad y_1$
$x_2 \quad y_2$
$\vdots \quad \vdots$
$x_n \quad y_n$

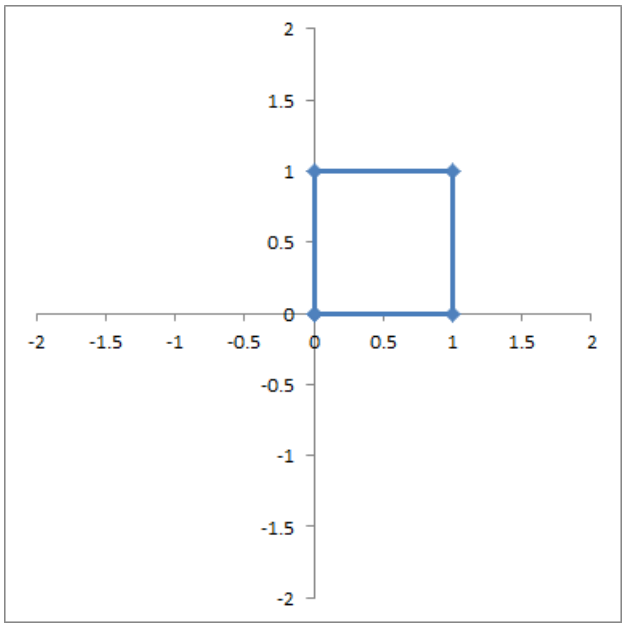
$1 \leq N \leq 10000$, $-10^6 \leq x_i, y_i \leq 10^6$. The position of railings (x_i, y_i) are integers and will be given in counter-clockwise order.

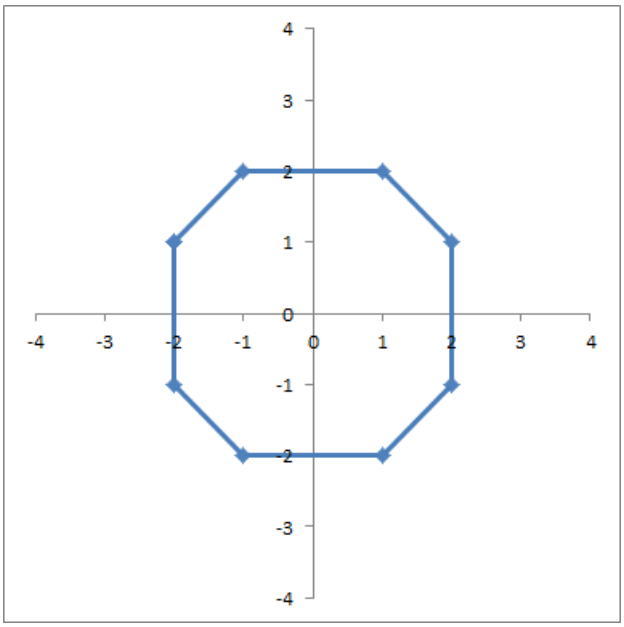
Output

For each test case, output the effectiveness of the field E.

Example

Standard Input	Standard Output
2 4 0 0 0 1 1 1 1 0 8 -2 1 -2 -1 -1 -2 1 -2 2 -1 2 1 1 2 -1 2	8 160





Euclidean distance of two points (x_1, y_1) and (x_2, y_2) over a plane $= \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$.

Hint 1: You may want to analyse the $n \times (2n - 1)$ distance matrix, it has a very nice property.

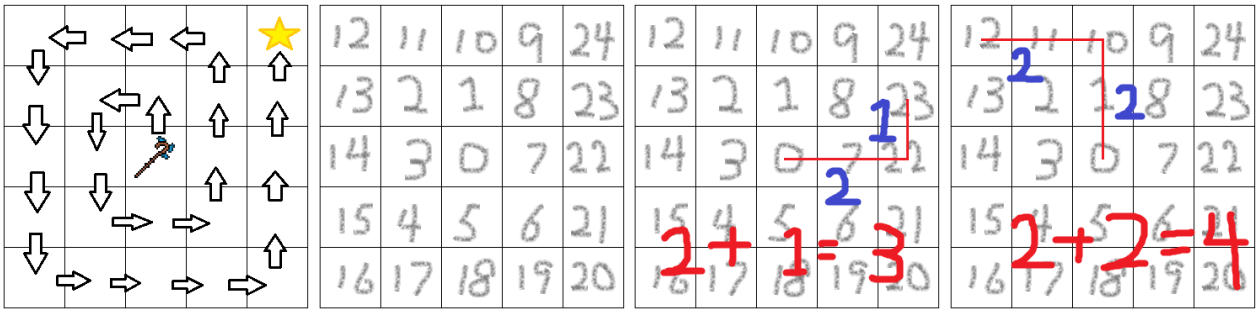
Hint 2: You may also consider the geometry. Some monotonic property may help speeding up your algorithm.

Problem G. Guru Guru

Input file:	`Standard Input`
Output file:	`Standard Output`
Time limit:	1 seconds
Memory limit:	64 megabytes

Kuriri is a magician, she knows a powerful magic "Guru Guru" that can destroy the world.

In order to cast that magic, she has to point her staff on the center of a $N \times N$ (N must be a positive odd number) 2D grid and then turn her staff round and round on the grid until she has gone through all the grids. She can move to another grid in 1 unit time. Basically, she needs $N^2 - 1$ unit time to cast the magic. The following figures show a 5×5 grid and the direction that she has to turn her staff to complete the casting.



She feels so boring when she is casting the magic, so she wants to know the Manhattan distance between the starting grid and the grid that she is pointing at time T. The Manhattan distance of two points (x_1, y_1) and (x_2, y_2) is equal to $|(x_1 - x_2)| + |(y_1 - y_2)|$.

Kukuri is weak at mathematics, please tell her the required distance at time T. If $T \geq n^2$, the world will be destroyed and you only need to tell her "The world has been destroyed!".

Input

The first line of the input is the number of test cases $K \leq 2000$.

Each test case has the following format:

N T

$1 \leq N \leq 10^7$, N must be an odd number. $0 \leq T \leq 10^{16}$.

Output

For each test case, output the Manhattan distance or "The world has been destroyed!".

Example

Standard Input	Standard Output
4 5 0 5 23 5 12 5 25	0 3 4 The world has been destroyed!