Problem A. A Colorful Grid

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

There is a grid with n rows and m columns, and $n \times m$ is even. The grid is painted with $\frac{n \times m}{2}$ kinds of colors. Each kind of color should occupy exactly two adjacent blocks. Two blocks are adjacent means they share an edge in the grid. We define (x, y) as the block in the x-th row and y-th column.

Little Horse is walking on the grid. Each time he can move to an adjacent block, but he cannot move to the one that shares the same color with his current position. Now, we give you k conditions. In each condition, we give two positions (x_1, y_1) and (x_2, y_2), and we will tell you whether Little Horse can move from one position to the other one. All you need to do is construct a grid that meets all the conditions.

Input

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

The first line of each test case contains three integers n, m, k ($1 \le n \times m \le 10^5$, $n \times m$ is even, $1 \le k \le 10^5$) — the number of rows and columns of the grid, and the number of conditions. The sum of $n \times m$ and the sum of k will not exceed 2×10^5.

Then in the next k lines, each line contains five integers x_1, y_1, x_2, y_2, c ($1 \le x_1, x_2 \le n$, $1 \le y_1, y_2 \le m$, $0 \le c \le 1$).

- $c = 0$: Little Horse is unable to move from (x_1, y_1) to (x_2, y_2).
- $c = 1$: Little Horse is able to move from (x_1, y_1) to (x_2, y_2).

Output

For the x-th test case, if there's no possible solution, output `Case #x: No` in a single line.

Otherwise, output `Case #x: Yes` in a single line, and then output n lines. Each line is a string of length m which contains only `L`, `R`, `D`, `U`. For the y-th character in the x-th line:

- `L`: (x, y) shares the same color with $(x, y - 1)$.
- `R`: (x, y) shares the same color with $(x, y + 1)$.
- `D`: (x, y) shares the same color with $(x + 1, y)$.
- `U`: (x, y) shares the same color with $(x - 1, y)$.

If there are multiple solutions, output any.

Example

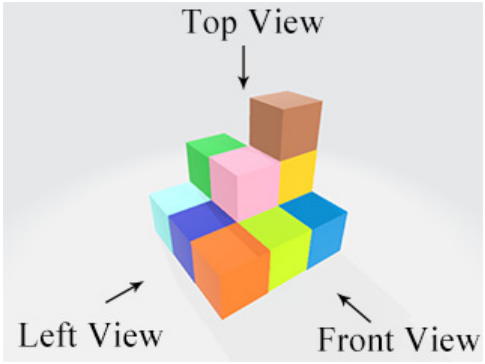
standard input	standard output
4 2 2 1 1 1 1 2 1 2 2 1 1 1 2 1 1 2 2 1 1 1 2 2 1 2 2 2 1 1 1 2 0 1 1 2 1 0	Case #1: Yes DD UU Case #2: Yes RL RL Case #3: No Case #4: No

Problem B. Building Blocks

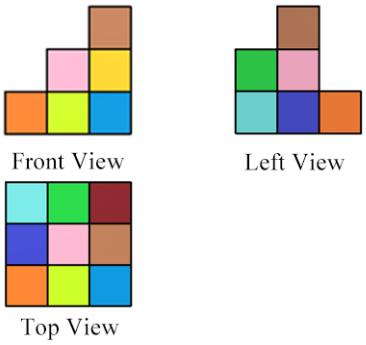
Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

As we know, for a three-dimensional object, we can draw its three views — front view, top view, and left view. We can regard these views as the projections of the object onto different planes.

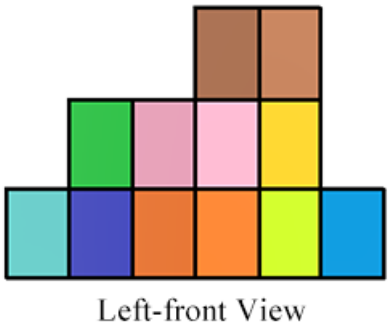
For example, consider some blocks like this:



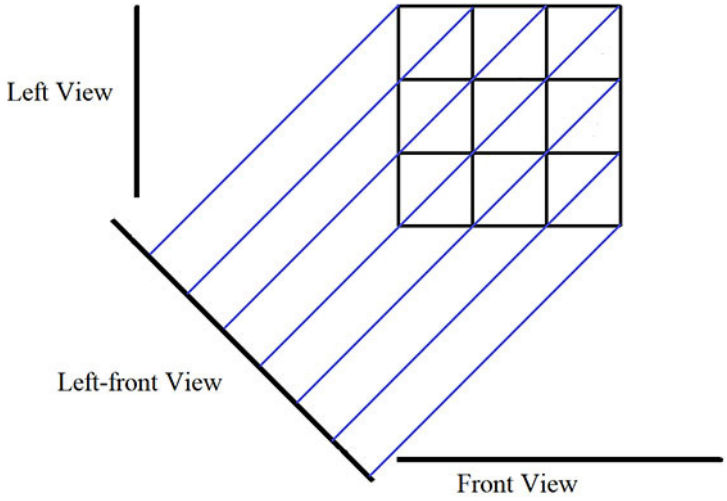
The three views can be drawn as follows:



But Little Rabbit thinks these views are too boring. He wants to observe these blocks in a quite special view — left-front view. Still consider the blocks given above. The left-front view can be drawn as follow:



The left-front view can also be regarded as a projection onto a plane, while the plane is at an angle of 45° to the left view's plane and the front view's plane. The following picture shows how it projects in a top view.



Therefore, if we regard the top view as a grid with n rows and m columns, the left-front view should have $n + m$ columns.

Now, Little Rabbit wants to build some blocks to meet the following conditions:

- The top view is a grid with n rows and m columns. Each square should place at least one block.
- Each column of the left-front view has a height of $a_1, a_2, \ldots, a_{n+m}$ from left to right.
- The x_i-th row and the y_i-th column of the top view has exactly h_i blocks. There are k such conditions.

Little Rabbit wonders how many different methods there are to meet all the conditions. Since the answer can be very large, please tell him the answer modulo $10^9 + 7$.

Input

The first line of the input contains an integer T ($1 \le T \le 10^5$) — the number of test cases.

The first line of each test case contains three integers n, m, k ($1 \le n, m, k \le 10^5$) — the number of rows and columns of the top view, and the number of conditions. The sum of n, the sum of m, and the sum of k will not exceed 10^5.

The second line contains $n + m$ integers $a_1, a_2, \ldots, a_{n+m}$ ($1 \le a_1, a_2, \ldots, a_{n+m} \le 10^9$) — the heights of the left-front view from left to right.

Then in the next k lines, the i-th line contains three integers x_i, y_i, h_i ($1 \le x_i \le n$, $1 \le y_i \le m$, $1 \le h_i \le 10^9$), which means the x_i-th row and the y_i-th column of the top view has exactly h_i blocks. It's guaranteed that all (x_i, y_i) are distinct.

Output

For the x-th test case, if the answer modulo $10^9 + 7$ is y, output `Case #x: y` in a single line.

Example

standard input	standard output
2 3 3 1 1 2 2 3 3 1 2 3 3 2 2 1 2 2 2 2 1 2 2	Case #1: 72 Case #2: 2

Problem C. Code a Trie

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

A Trie is a tree data structure used to store strings. Here is Little Rabbit's C++ code for inserting and querying a string in a Trie. The character set is lowercase letters.

```
struct Trie {
    int e[N][26], val[N], tot; // N is a constant
    Trie() {
        tot = 0;
        val[0] = Rand(1, 1000000000);
        memset(e, 0, sizeof e);
    }
    void insert(const char *s, int n) { // n is the length of string s
        for (int i = 0, u = 0; i < n; u = e[u][s[i] - 'a'], i++)
            if (!e[u][s[i] - 'a']) {
                e[u][s[i] - 'a'] = ++tot;
                val[tot] = Rand(1, 1000000000);
                // Rand(l, r) can generate
                // distinct random numbers in the range of [l, r]
            }
    }
    int query(const char *s, int n) { // n is the length of string s
        int u = 0;
        for (int i = 0; i < n; u = e[u][s[i] - 'a'], i++)
            if (!e[u][s[i] - 'a'])
                break;
        return val[u];
    }
};
```

Please note that the integers generated by `Rand` function are all distinct.

Little Rabbit inserts some (possibly zero) strings in the Trie by using the `insert` function. But after a while, he totally forgets those strings he has inserted. So he uses the `query` function n times to query some strings and gets n return values. According to these values, you need to tell Little Rabbit how many nodes the Trie has at least. Please note that the root of the Trie should also be counted as a node. In the code above, the number of nodes is tot + 1.

Input

The first line of the input contains an integer T ($1 \le T \le 5000$) — the number of test cases.

The first line of each test case contains an integer n ($1 \le n \le 10^5$) — the number of times Little Rabbit uses the `query` function.

Then in the next n lines, each line contains a string s containing only lowercase letters and an integer v ($1 \le v \le 10^9$), which means Little Rabbit queries the string s, and the return value is v. The sum of the lengths of the strings in each test case will not exceed 10^5, and the sum of the lengths of the strings in all test cases will not exceed 5×10^5

Output

For the x-th test case, if the answer is y, output `Case #x: y` in a single line. If no Trie can meet the conditions, output `Case #x: -1` in a single line.

Example

standard input	standard output
6 2 aa 1 a 2 1 a 1 3 aa 1 ab 1 ac 1 2 aa 1 ac 2 3 aaa 1 ac 1 aa 3 3 aa 1 ac 1 a 3	Case #1: 3 Case #2: 1 Case #3: 1 Case #4: 3 Case #5: -1 Case #6: -1

Problem D. Defuse the Bombs

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

The terrorists have planted some bombs in a building! Our hero, Little Horse, decides to rescue the people in the building. Unfortunately, there is more than one bomb, and Little Horse is unable to defuse all of them. To strive for more time for other people to escape, Little Horse decides to sacrifice himself.

There are n bombs in the building, each of which has a countdown clock. In the beginning, the i-th bomb's clock is set to a_i. Then:

1. Little Horse chooses one bomb, making its clock increase by 1.
2. Every bomb's clock decreases by 1.
3. If at least one clock becomes lower than 0, all the bombs will explode. Otherwise, go back to step 1.

Obviously, the explosion is not avoidable. What a sad story. But Little Horse doesn't care about his survival now. He just wants to strive for more time. So can you tell him how many times he can do step 1 at most before the explosion?

Input

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

The first line of the input contains an integer n ($2 \le n \le 10^5$) — the number of bombs. The sum of n will not exceed 3×10^5.

The next line contains n numbers $a_1, a_2, \ldots, a_n$ ($0 \le a_1, a_2, \ldots, a_n \le 10^9$) — the clocks of the bombs in the beginning.

Output

For the x-th test case, if the answer is y, output `Case #x: y` in a single line.

Example

standard input	standard output
2 2 1 1 3 1 2 3	Case #1: 3 Case #2: 4

Problem E. Escape from the Island

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Little Horse wakes up and finds himself located on a desert island. The only way to escape from the island is by rowing a boat.

There are n islands numbered from 1 to n. Little Horse starts from one of these islands, and he needs to reach the n-th island to be rescued. There are m waterways, each of which connects two islands. In each waterway, the water flows from one island to the other one. Little Horse will apply the following steps in order:

1. Row the boat. At this step, he can go through at most k (possibly zero) waterways, either downstream or upstream. Going through each waterway will take 1 time unit.
2. Take a rest. At this step, he will randomly choose a waterway downstream and drift through it. He cannot stay at the current island unless there is no such waterway to drift through. Whether he moves or not, taking a rest will take 1 time unit.
3. Go back to step 1.

Once Little Horse reaches the n-th island, he will stop immediately. Please tell Little Horse that if he starts from the i-th island, what's the minimum time units he will spend in the worst case.

Input

The first line of the input contains an integer T ($1 \leq T \leq 10$) — the number of test cases.

The first line of each test case contains three integers n, m, k ($1 \leq n \leq 10^5$, $0 \leq m \leq 10^5$, $0 \leq k \leq 50$) — The number of islands and waterways, and the maximum number of waterways Little Horse can go through when he rows the boat. The sum of n and the sum of m will not exceed 10^5.

Then in the next m lines, each line contains two integers u, v ($1 \leq u, v \leq n$, $u \neq v$), which means there's a waterway between island u and v, and the water flows from u to v. It's guaranteed that the m waterways are all distinct, but it's possible that there's a waterway from u to v and also a waterway from v to u.

Output

The output of the x-th case begins with `Case #x:` in a single line.

Then in the next n lines, the i-th line contains an integer indicating the minimum time units in the worst case if Little Horse starts from the i-th island. If he cannot reach the n-th island in the worst case, output -1.

Example

standard input	standard output
3 3 3 1 1 2 2 3 1 3 3 2 1 2 1 3 2 4 3 2 2 1 3 2 4 3	Case #1: 1 1 0 Case #2: -1 1 0 Case #3: 5 2 1 0

Problem F. Fracture Ray

Input file:	`standard input`
Output file:	`standard output`
Time limit:	10 seconds
Memory limit:	512 megabytes

A monster is attacking our city! Our hero, Little Rabbit, comes to rescue the city. Little Rabbit has a very powerful skill — Fracture Ray. The rays can penetrate the monster and cause huge damage to it. But the monster also has a defensive skill — Hall of Mirrors. Several mirrors will be placed on the battleground, which can reflect the rays. However, since the rays are too powerful, each mirror can only reflect the ray once, and it will break down afterward.

We can regard the battleground as a two-dimensional plane, which can be described by Cartesian coordinates. There are n double-sided mirrors on the plane, which can be considered as segments that are parallel to the x-axis or y-axis. The coordinates of their endpoints are all integers.

Little Rabbit will emit m rays. Each ray starts from a point and has an initial direction which is at an angle of 45° to the x-axis and y-axis. When the ray hits a mirror, it will be reflected specularly (the angle of reflection equals the angle of incidence). And it will also break the mirror down, which means the mirror will disappear. Little Rabbit will emit the rays one by one. Only if a ray goes out of range, which means it won't hit any more mirrors, Little Rabbit will emit the next ray. It's guaranteed that the ray won't go through any points whose coordiantes are integers. See the input format for more information.

Little Rabbit has the confidence to defeat the monster. However, he wonders which mirrors the i-th ray will hit.

Input

The first line of the input contains an integer T ($1 \le T \le 2$) — the number of test cases.

The first line of each test case contains two integers n, m ($1 \le n, m \le 10^5$) — the number of mirrors and rays.

Then in the next n lines, the i-th line contains four integers x_1, y_1, x_2, y_2 ($0 \le x_1, y_1, x_2, y_2 \le 10^9$), which means the endpoints of the i-th mirror are (x_1, y_1) and (x_2, y_2). It's guaranteed that $x_1 = x_2$ or $y_1 = y_2$, and $(x_1, y_1) \neq (x_2, y_2)$. It's also guarenteed that the intersection of every pair of mirrors is an empty set or only a point.

Then in the next m lines, the i-th line contains four integers x, y, z, d ($0 \le x, y \le 10^9$, $0 \le z \le 1$, $0 \le d \le 3$), indicating the start point and the initial direction of the i-th ray.

The first three integers x, y, z indicate the start point of the ray.

- $z = 0$: the ray starts at $(x + 0.5, y)$.
- $z = 1$: the ray starts at $(x, y + 0.5)$.

The last integer d indicates the initial direction of the ray. We use a vector to describe the direction.

- $d = 0$: the initial direction $\vec{s} = (1, -1)$.
- $d = 1$: the initial direction $\vec{s} = (-1, -1)$.
- $d = 2$: the initial direction $\vec{s} = (-1, 1)$.
- $d = 3$: the initial direction $\vec{s} = (1, 1)$.

It's guaranteed that the start point of a ray won't be on the segment of any mirror.

Output

The output of the x-th test case begins with `Case #x:` in a single line.

Then in the next m lines, the first integer of the i-th line is y, which means the i-th ray hits y mirrors in total. Then there follows y integers, which means the i-th ray hits the a_1-th, a_2-th, ..., a_y-th mirror in order. These y integers must be given in the order of the ray hitting the mirror.

Example

standard input	standard output
1 4 2 1 0 6 0 5 4 4 4 1 3 3 3 5 6 4 6 7 2 1 1 0 2 0 3	Case #1: 2 1 3 1 4

Problem G. Game of Cards

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Little Rabbit and Little Horse love playing odd card games. Now, they are playing a card game called *0123 − game*.

There are several cards on the table. c_0 of them are labeled with 0, c_1 of them are labeled with 1, c_2 of them are labeled with 2, and c_3 of them are labeled with 3. Little Rabbit and Little Horse take turns to play the game, and Little Rabbit goes first. In each turn, the player should choose two cards on the condition that the sum of the numbers on the two cards is no more than 3, then replace these two cards with a card labeled with their sum. The player who cannot make a move loses the game.

Little Rabbit and Little Horse wonder who will win the game.

Input

The first line of the input contains an integer T ($1 \leq T \leq 10^5$) — the number of test cases.

Each test case contains four integers c_0, c_1, c_2, c_3 ($0 \leq c_0, c_1, c_2, c_3 \leq 10^9$) — the number of cards labeled with $0, 1, 2, 3$.

Output

For the x-th test case, if Little Rabbit wins, output `Case #x: Rabbit` in a single line. Otherwise, output `Case #x: Horse` in a single line.

Example

standard input	standard output
2 1 1 1 1 2 2 2 2	Case #1: Horse Case #2: Rabbit

Problem H. Hide and Seek

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Little Rabbit and Little Horse are playing hide and seek on a plane that can be described by Cartesian coordinates. Little Horse is the hider and Little Rabbit is the seeker. Little Rabbit is eager to win, so he decides to cheat. He prepared three locating devices, placing one at $(0, 0)$, one on Little Horse, and one on himself. These devices cannot show accurate coordinates but can show the Manhattan distance between every two devices.

The Manhattan distance is the sum of the lengths of the projections of the line segment between the points onto the coordinate axes. For example, the Manhattan distance between (x_1, y_1) and (x_2, y_2) is $|x_1 - x_2| + |y_1 - y_2|$. That is to say, if Little Rabbit is at (x_1, y_1), and Little Horse is at (x_2, y_2), then Little Rabbit's devices will show $d_{01} = |x_1| + |y_1|$, $d_{02} = |x_2| + |y_2|$ and $d_{12} = |x_1 - x_2| + |y_1 - y_2|$.

After getting these data, it's much easier for Little Rabbit to find Little Horse. But Little Rabbit wants to know that if their coordinates are integers, how many different situations of their locations there are. Please note that when $(x_1, y_1) \neq (x_2, y_2)$, Little Rabbit at (x_1, y_1) and Little Horse at (x_2, y_2) is a different situation from Little Rabbit at (x_2, y_2) and Little Horse at (x_1, y_1). It's also possible that Little Rabbit and Little Horse are at the same point.

Input

The first line of the input contains an integer T ($1 \leq T \leq 10^5$) — the number of test cases.

Each test case contains three integers d_{01}, d_{02}, d_{12} ($0 \leq d_{01}, d_{02}, d_{12} \leq 10^9$), the meanings of which are given above.

Output

For the x-th test case, if the answer is y, output `Case #x: y` in a single line.

Example

standard input	standard output
4 2 4 2 1 3 2 3 4 5 1 1 1	Case #1: 32 Case #2: 20 Case #3: 48 Case #4: 0

Problem I. Invaluable Assets

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

Lucid waters and lush mountains are invaluable assets. Guided by this conviction, Little Horse starts to plant trees. Little Horse has n trees. In the beginning, the i-th tree has a height of h_i. To make the trees grow higher, Little Horse needs to buy fertilizers. The fertilizer which can make a tree's height increase by x has a cost of $x^2 + c$ (c is a given const, and x must be an integer).

Now, Little Horse wants to know what's the minimum cost to make all trees' heights no less than k. There are q queries about it. Please note that buying only one fertilizer for a tree is not necessarily the best solution. For example, to make a tree's height increse by 3, one method is to spend $3^2 + c$, another method is to spend $(1^2 + c) + (2^2 + c)$.

Input

The first line of the input contains an integer T ($1 \le T \le 10$) — the number of test cases.

The first line of each test case contains three integers n, c, q ($1 \le n, q \le 10^5$, $1 \le c \le 10^4$) — the number of trees, the given const, and the number of queries.

Then the next line contains n integers $h_1, h_2, \ldots, h_n$ ($0 \le h_1, h_2, \ldots, h_n \le 10^9$) — the height of each tree in the beginning.

Then in the next q lines, each line contains an integer k ($0 \le k \le 10^9$), indicating the query.

It's guaranteed that $h_1, h_2, \ldots, h_n$ and k are generated uniformly and randomly within $[0, 10^9]$.

Output

The output of the x-th test case begins with `Case #x:` in a single line.

Then in the next q lines, each line contains an integer, indicating the answer to each query.

Example

standard input	standard output
2 3 2 3 3 0 2 1 2 3 4 3 5 0 2 3 4 1 2 3 4 5	Case #1: 3 6 12 Case #2: 4 7 15 25 40

Problem J. Joy of Handcraft

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Little Horse always does some handcrafts, which is full of joy. This time, he builds a circuit that can turn on and off the bulbs periodically.

There are n bulbs in the circuit, the i-th of which has a period t_i and a luminance x_i. Formally, the i-th bulb will be turned on from the $(2kt_i + 1)$-th second to the $(2kt_i + t_i)$-th second, and it will be turned off from the $(2kt_i + t_i + 1)$-th second to the $(2kt_i + 2t_i)$-th second, $k = 0, 1, 2, \ldots$ When the i-th bulb is on, its luminance will be x_i, otherwise its luminance will be 0.

Now, Little Horse wants to know, for each second from the first second to the m-th second, what's the maximum luminance among all the bulbs.

Input

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

The first line of each test case contains two integers n, m ($1 \le n, m \le 10^5$) — the number of bulbs, and the number of integers you need to output. The sum of n and the sum of m will not exceed 2×10^5.

Then in the next n lines, the i-th line contains two integers t_i, x_i ($1 \le t_i, x_i \le 10^5$) — the period and the luminance of the i-th bulb.

Output

The x-th test case begins with `Case #x:`, and there follow m integers. The i-th integer indicates the maximum luminance among all the bulbs in the i-th second. If no bulb is on in the i-th second, output 0.

Example

standard input	standard output
3 2 3 1 1 2 2 2 5 1 2 2 3 3 3 1 1 1 2 1 3	Case #1: 2 2 1 Case #2: 3 3 2 0 3 Case #3: 3 0 3

Problem K. Knowledge is Power

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Knowledge is power. Little Rabbit and Little Horse both long for more knowledge, so they always challenge each other to some quizzes. Today, Little Rabbit creates a new quiz for Little Horse.

Little Rabbit gives Little Horse a positive integer x. Little Horse needs to find a set of integers $S = \{a_1, a_2, \ldots, a_n\}$ that meets the following conditions.

- $n \geq 2$
- $a_i > 1$, for $1 \leq i \leq n$
- $\sum_{i=1}^{n} a_i = x$
- a_i and a_j are co-prime, for any $i \neq j$

For example, if $x = 12$, then $S = \{3, 4, 5\}$ and $S = \{5, 7\}$ and $S = \{2, 3, 7\}$ are all valid sets. Two integers are said to be co-prime if the only positive integer that evenly divides both of them is 1.

We define $a_{\max}$ as the maximum element of S, and $a_{\min}$ as the minimum element of S. Little Rabbit wants the value of $(a_{\max} - a_{\min})$ to be as small as possible. Can you help Little Horse to find the minimum value of $(a_{\max} - a_{\min})$?

Input

The first line of the input contains an integer T ($1 \leq T \leq 10^5$) — the number of test cases.

Each test case contains an integer x ($5 \leq x \leq 10^9$) — the integer Little Rabbit gives to Little Horse.

Output

For the x-th test case, if the answer is y, output `Case #x: y` in a single line. If there's no possible solution, output `Case #x: -1` in a single line.

Example

standard input	standard output
4 5 6 7 10	Case #1: 1 Case #2: -1 Case #3: 1 Case #4: 3

Problem L. Lottery

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Little Rabbit walks into a shopping mall and comes across a lottery activity. There are n boxes. Inside the i-th box, there are x_i balls labeled with 2^{a_i}. Little Rabbit can pick out any number (including zero) of balls from each box. The sum of numbers on the chosen balls is the score he gets. Whether Little Rabbit wins a prize in the lottery depends on the score.

However, Little Rabbit doesn't care if he can win a prize. He is curious about how many different scores he is likely to get. Can you help him with it? Since the answer can be very large, please tell him the answer modulo $10^9 + 7$.

Input

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

In each test case, the first line contains an integer n ($1 \le n \le 10^5$) — the number of boxes. The sum of n will not exceed 10^5.

Then in the next n lines, the i-th line contains two integers a_i ($0 \le a_i \le 10^9$) and x_i ($1 \le x_i \le 10^9$), which means there are x_i balls numbered 2^{a_i} inside the i-th box. It's guaranteed that $a_1, a_2, \ldots, a_n$ are all distinct.

Output

For the x-th test case, if the answer modulo $10^9 + 7$ is y, output `Case #x: y` in a single line.

Example

standard input	standard output
2 3 1 1 2 1 3 1 3 1 1 2 2 3 3	Case #1: 8 Case #2: 18