A. Distanced Coloring

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

You received an $n \times m$ grid from a mysterious source. The source also gave you a magic positive integer constant k.

The source told you to color the grid with some colors, satisfying the following condition:

- If (x_1, y_1), (x_2, y_2) are two distinct cells with the same color, then $\max(|x_1 - x_2|, |y_1 - y_2|) \geq k$.

You don't like using too many colors. Please find the minimum number of colors needed to color the grid.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 1000$). The description of the test cases follows.

The only line of each test case consists of three positive integers n, m, k ($1 \leq n, m, k \leq 10^4$) — the dimensions of the grid and the magic constant.

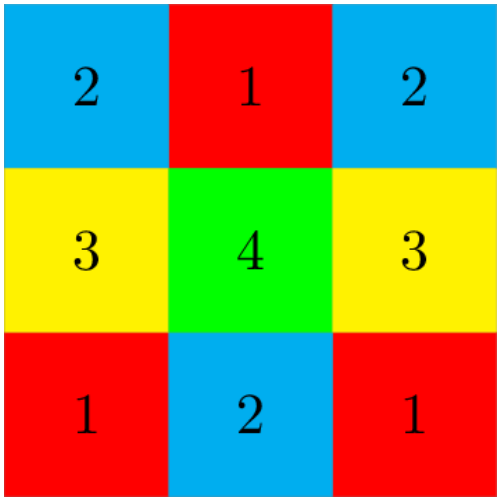
Output

For each test case, print a single integer — the minimum number of colors needed to color the grid.

Standard Input	Standard Output
6 3 3 2 5 1 10000 7 3 4 3 2 7 8 9 6 2 5 4	4 5 12 6 36 8

Note

In the first test case, one of the optimal constructions is:



In the second test case, the color of all cells must be pairwise different, so the answer is 5.

,

B. Removals Game

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

Alice got a permutation $a_1, a_2, \ldots, a_n$ of $[1, 2, \ldots, n]$, and Bob got another permutation $b_1, b_2, \ldots, b_n$ of $[1, 2, \ldots, n]$. They are going to play a game with these arrays.

In each turn, the following events happen in order:

- Alice chooses either the first or the last element of her array and removes it from the array;
- Bob chooses either the first or the last element of his array and removes it from the array.

The game continues for $n - 1$ turns, after which both arrays will have exactly one remaining element: x in the array a and y in the array b.

If $x = y$, Bob wins; otherwise, Alice wins. Find which player will win if both players play optimally.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^4$). The description of the test cases follows.

The first line of each test case contains a single integer n ($1 \le n \le 3 \cdot 10^5$).

The next line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le n$, all a_i are distinct) — the permutation of Alice.

The next line contains n integers $b_1, b_2, \ldots, b_n$ ($1 \le b_i \le n$, all b_i are distinct) — the permutation of Bob.

It is guaranteed that the sum of all n does not exceed $3 \cdot 10^5$.

Output

For each test case, print a single line with the name of the winner, assuming both players play optimally. If Alice wins, print `Alice`; otherwise, print `Bob`.

Standard Input	Standard Output
2 2 1 2 1 2 3 1 2 3 2 3 1	Bob Alice

Note

In the first test case, Bob can win the game by deleting the same element as Alice did.

In the second test case, Alice can delete 3 in the first turn, and then in the second turn, delete the element that is different from the one Bob deleted in the first turn to win the game.

,

C. Black Circles

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

There are n circles on a two-dimensional plane. The i-th circle is centered at (x_i, y_i). Initially, all circles have a radius of 0.

The circles' radii increase at a rate of 1 unit per second.

You are currently at (x_s, y_s); your goal is to reach (x_t, y_t) without touching the circumference of any circle (**including the moment you reach** (x_t, y_t)). You can move in any direction you want. However, your speed is limited to 1 unit per second.

Please determine whether this is possible.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^4$). The description of the test cases follows.

The first line of each test case contains a single integer n ($1 \le n \le 10^5$) — the number of circles.

The next n lines each contain two integers x_i, y_i ($1 \le x_i, y_i \le 10^9$) — the center of each circle.

The final line contains four integers x_s, y_s, x_t, y_t ($1 \le x_s, y_s, x_t, y_t \le 10^9$) — the coordinates of the starting point and the goal, respectively.

It is guaranteed that these $n+2$ points are distinct.

It is guaranteed that the sum of n over all testcases does not exceed 10^5.

Output

For each test case, output `YES` if it is possible to reach the goal without touching the circle boundaries, and output `NO` otherwise.

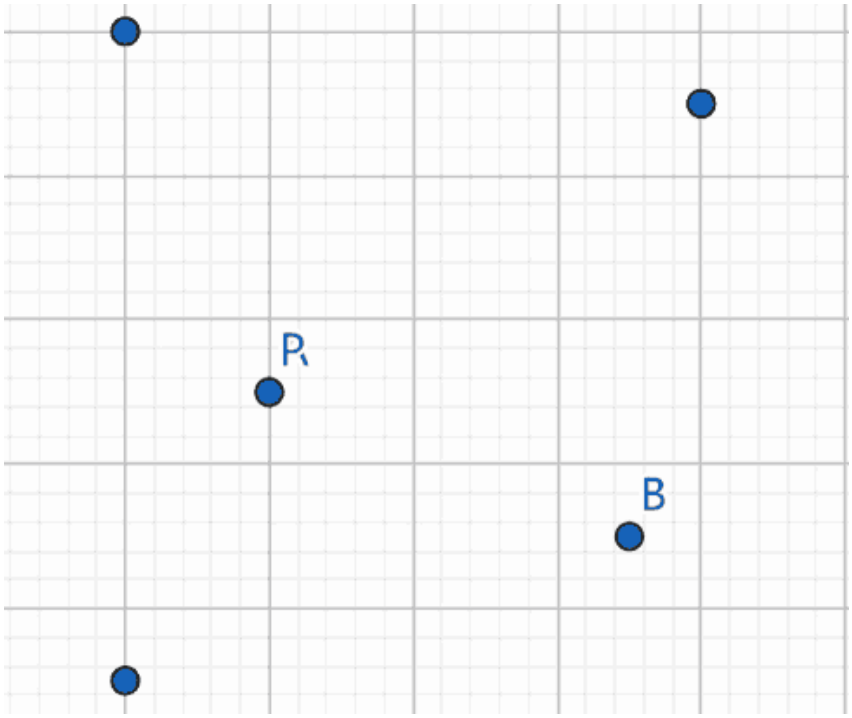
You can output `Yes` and `No` in any case (for example, strings `yEs`, `yes`, `Yes`, and `YES` will be recognized as a positive response).

Standard Input	Standard Output
7 3 2 5 2 14 10 13 4 9 9 7 3 10 11 6 9 12 12 14 13 4 8 1 5 7 12 6 11 13 2 1000000000 2 2 1000000000 1 1 2 2 1 999999998 1000000000 999999999 999999999 1 1 1	YES NO YES YES YES NO YES

1000000000 1 1 1000000000 1 1 10 989237121 2397081 206669655 527238537 522705783 380636165 532545346 320061691 207818728 199485303 884520552 315781807 992311437 802563521 205138355 324818663 223575704 395073023 281560523 236279118 216941610 572010615 323956540 794523071	

Note

In the first test case, a feasible way of movement is as follows.



,

D1. DFS Checker (Easy Version)

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 512 megabytes

This is the easy version of the problem. In this version, the given tree is a perfect binary tree and the constraints on n and q are lower. You can make hacks only if both versions of the problem are solved.

You are given a perfect binary tree[†] consisting of n vertices. The vertices are numbered from 1 to n, and the root is the vertex 1. You are also given a permutation $p_1, p_2, \ldots, p_n$ of $[1, 2, \ldots, n]$.

You need to answer q queries. For each query, you are given two integers x, y; you need to swap p_x and p_y and determine if $p_1, p_2, \ldots, p_n$ is a valid DFS order[‡] of the given tree.

Please note that the swaps are **persistent** through queries.

[†] A perfect binary tree is a tree with vertex 1 as its root, with size $n = 2^k - 1$ for a positive integer k, and where the parent of each vertex i ($1 < i \leq n$) is $\left\lfloor \frac{i}{2} \right\rfloor$. Thus, all leaves of this tree are at a distance $k - 1$ from the root.

[‡] A DFS order is found by calling the following `dfs` function on the given tree.

```
dfs_order = []

function dfs(v):
    append v to the back of dfs_order
    pick an arbitrary permutation s of children of v
    for child in s:
        dfs(child)
dfs(1)
```

Note that the DFS order is not unique.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 10^4$). The description of the test cases follows.

The first line of each test case contains two integers n, q ($3 \leq n \leq 65\,535$, $2 \leq q \leq 5 \cdot 10^4$) — the number of vertices in the tree and the number of queries. It is guaranteed that $n = 2^k - 1$ for a positive integer k.

The next line contains $n - 1$ integers $a_2, a_3, \ldots, a_n$ ($1 \leq a_i < i$) — the parent of each vertex in the given tree. It is guaranteed that $a_i = \left\lfloor \frac{i}{2} \right\rfloor$.

The next line contains n integers $p_1, p_2, \ldots, p_n$ ($1 \leq p_i \leq n$, all p_i are distinct) — the initial permutation p.

The next q lines each contain two integers x, y ($1 \leq x, y \leq n, x \neq y$) — the positions of the elements to swap in the permutation.

It is guaranteed that the sum of all n does not exceed $65\,535$, and the sum of all q does not exceed $5 \cdot 10^4$.

Output

For each test case, print q lines corresponding to the q queries. For each query, output `YES` if there is a DFS order that exactly equals the current permutation, and output `NO` otherwise.

You can output `Yes` and `No` in any case (for example, strings `yEs`, `yes`, `Yes` and `YES` will be recognized as a positive response).

Standard Input	Standard Output
2 3 3	YES YES

1 1 1 2 3 2 3 3 2 1 3 7 4 1 1 2 2 3 3 1 2 3 4 5 6 7 3 5 2 5 3 7 4 6	NO YES NO NO YES

Note

In the first test case, the permutation $p_1, p_2, \ldots, p_n$ after each modification is $[1,3,2]$, $[1,2,3]$, $[3,2,1]$, respectively. The first two permutations are valid DFS orders; the third is not a DFS order.

In the second test case, the permutation $p_1, p_2, \ldots, p_n$ after each modification is $[1,2,5,4,3,6,7]$, $[1,3,5,4,2,6,7]$, $[1,3,7,4,2,6,5]$, $[1,3,7,6,2,4,5]$, respectively.

,

D2. DFS Checker (Hard Version)

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 512 megabytes

This is the hard version of the problem. In this version, you are given a generic tree and the constraints on n and q are higher. You can make hacks only if both versions of the problem are solved.

You are given a rooted tree consisting of n vertices. The vertices are numbered from 1 to n, and the root is the vertex 1. You are also given a permutation $p_1, p_2, \ldots, p_n$ of $[1, 2, \ldots, n]$.

You need to answer q queries. For each query, you are given two integers x, y; you need to swap p_x and p_y and determine if $p_1, p_2, \ldots, p_n$ is a valid DFS order† of the given tree.

Please note that the swaps are **persistent** through queries.

† A DFS order is found by calling the following `dfs` function on the given tree.

```
dfs_order = []

function dfs(v):
    append v to the back of dfs_order
    pick an arbitrary permutation s of children of v
    for child in s:
        dfs(child)
dfs(1)
```

Note that the DFS order is not unique.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^4$). The description of the test cases follows.

The first line of each test case contains two integers n, q ($2 \le n \le 3 \cdot 10^5$, $2 \le q \le 10^5$) — the number of vertices in the tree and the number of queries.

The next line contains $n - 1$ integers $a_2, a_3, \ldots, a_n$ ($1 \le a_i < i$) — the parent of each vertex in the given tree.

The next line contains n integers $p_1, p_2, \ldots, p_n$ ($1 \le p_i \le n$, all p_i are distinct) — the initial permutation p.

The next q lines each contain two integers x, y ($1 \le x, y \le n, x \neq y$) — the positions of the elements to swap in the permutation.

It is guaranteed that the sum of all n does not exceed $3 \cdot 10^5$, and the sum of all q does not exceed 10^5.

Output

For each test case, print q lines corresponding to the q queries. For each query, output `YES` if there is a DFS order that exactly equals the current permutation, and output `NO` otherwise.

You can output `Yes` and `No` in any case (for example, strings `yEs`, `yes`, `Yes`, and `YES` will be recognized as a positive response).

Standard Input	Standard Output
3 3 3 1 1 1 2 3 2 3 3 2	YES YES NO YES NO NO

1 3 7 4 1 1 2 2 3 3 1 2 3 4 5 6 7 3 5 2 5 3 7 4 6 5 4 1 1 3 4 2 3 4 5 1 5 1 4 5 3 4 2 3	YES YES NO NO YES

Note

In the first test case, the permutation $p_1, p_2, \ldots, p_n$ after each modification is $[1,3,2]$, $[1,2,3]$, $[3,2,1]$, respectively. The first two permutations are valid DFS orders; the third is not a DFS order.

In the second test case, the permutation $p_1, p_2, \ldots, p_n$ after each modification is $[1,2,5,4,3,6,7]$, $[1,3,5,4,2,6,7]$, $[1,3,7,4,2,6,5]$, $[1,3,7,6,2,4,5]$, respectively.

,

E. Cosmic Rays

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 512 megabytes

Given an array of integers $s_1, s_2, \ldots, s_l$, every second, cosmic rays will cause all s_i such that $i = 1$ or $s_i \neq s_{i-1}$ to be deleted simultaneously, and the remaining parts will be concatenated together in order to form the new array $s_1, s_2, \ldots, s_{l'}$.

Define the **strength** of an array as the number of seconds it takes to become empty.

You are given an array of integers compressed in the form of n pairs that describe the array left to right. Each pair (a_i, b_i) represents a_i copies of b_i, i.e. $\underbrace{b_i, b_i, \cdots, b_i}_{a_i \text{ times}}$.

For each $i = 1, 2, \ldots, n$, please find the **strength** of the sequence described by the first i pairs.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 10^4$). The description of the test cases follows.

The first line of each test case contains a single integer n ($1 \leq n \leq 3 \cdot 10^5$) — the length of sequence a.

The next n lines contain two integers each a_i, b_i ($1 \leq a_i \leq 10^9, 0 \leq b_i \leq n$) — the pairs which describe the sequence.

It is guaranteed that the sum of all n does not exceed $3 \cdot 10^5$.

It is guaranteed that for all $1 \leq i < n$, $b_i \neq b_{i+1}$ holds.

Output

For each test case, print one line containing n integers — the answer for each prefix of pairs.

Standard Input	Standard Output
4 4 2 0 1 1 3 0 5 1 6 4 6 1 3 4 6 4 0 7 6 6 3 7 9 0 7 1 5 0 7 1 9 0 1 1 2 0 10 10 7 4 9 2 2	2 2 4 5 4 4 7 7 10 10 9 9 9 9 9 9 10 10 10 10 10 10 10 12 15 15 15

7 9 2 8 8 5 11 7 15 5 12 7 4 0	

Note

In the first test case, for the prefix of length 4, the changes will be $[0,0,1,0,0,0,1,1,1,1,1] \rightarrow [0,0,0,1,1,1,1] \rightarrow [0,0,1,1,1] \rightarrow [0,1,1] \rightarrow [1] \rightarrow []$, so the array becomes empty after 5 seconds.

In the second test case, for the prefix of length 4, the changes will be $[6,6,6,6,3,6,6,6,6,0,0,0,0] \rightarrow [6,6,6,6,6,6,6,0,0,0] \rightarrow [6,6,6,6,6,6,0,0] \rightarrow [6,6,6,6,0] \rightarrow [6,6,6] \rightarrow [6,6] \rightarrow [6] \rightarrow []$, so the array becomes empty after 7 seconds.

,

F1. Court Blue (Easy Version)

Input file: standard input
Output file: standard output
Time limit: 3 seconds
Memory limit: 512 megabytes

This is the easy version of the problem. In this version, $n = m$ and the time limit is lower. You can make hacks only if both versions of the problem are solved.

In the court of the Blue King, Lelle and Flamm are having a performance match. The match consists of several rounds. In each round, either Lelle or Flamm wins.

Let W_L and W_F denote the number of wins of Lelle and Flamm, respectively. The Blue King considers a match to be **successful** if and only if:

- after every round, $\gcd(W_L, W_F) \leq 1$;
- at the end of the match, $W_L \leq n, W_F \leq m$.

Note that $\gcd(0, x) = \gcd(x, 0) = x$ for every non-negative integer x.

Lelle and Flamm can decide to stop the match whenever they want, and the final score of the performance is $l \cdot W_L + f \cdot W_F$.

Please help Lelle and Flamm coordinate their wins and losses such that the performance is **successful**, and the total score of the performance is maximized.

Input

The first line contains an integer t ($1 \leq t \leq 10^3$) — the number of test cases.

The only line of each test case contains four integers n, m, l, f ($2 \leq n \leq m \leq 2 \cdot 10^7$, $1 \leq l, f \leq 10^9$, $\mathbf{n = m}$): n, m gives the upper bound on the number of Lelle and Flamm's wins, l and f determine the final score of the performance.

Unusual additional constraint: it is guaranteed that, for each test, there are no pairs of test cases with the same pair of n, m.

Output

For each test case, output a single integer — the maximum total score of a **successful** performance.

Standard Input	Standard Output
8 3 3 2 5 4 4 1 4 6 6 2 2 7 7 2 3 9 9 9 1 2 2 1 4 5 5 1 4 8 8 6 7	19 17 18 33 86 9 24 86
1 20000000 20000000 1341 331	33439999007
2 1984 1984 19 84 9982 9982 44 35	204143 788403

Note

In the first test case, a possible performance is as follows:

- Flamm wins, $\gcd(0, 1) = 1$.
- Lelle wins, $\gcd(1, 1) = 1$.

- Flamm wins, $\gcd(1, 2) = 1$.
- Flamm wins, $\gcd(1, 3) = 1$.
- Lelle wins, $\gcd(2, 3) = 1$.
- Lelle and Flamm agree to stop the match.

The final score is $2 \cdot 2 + 3 \cdot 5 = 19$.

In the third test case, a possible performance is as follows:

- Flamm wins, $\gcd(0, 1) = 1$.
- Lelle wins, $\gcd(1, 1) = 1$.
- Lelle wins, $\gcd(2, 1) = 1$.
- Lelle wins, $\gcd(3, 1) = 1$.
- Lelle wins, $\gcd(4, 1) = 1$.
- Lelle wins, $\gcd(5, 1) = 1$.
- Flamm wins, $\gcd(5, 2) = 1$.
- Flamm wins, $\gcd(5, 3) = 1$.
- Flamm wins, $\gcd(5, 4) = 1$.
- Lelle and Flamm agree to stop the match.

The final score is $5 \cdot 2 + 4 \cdot 2 = 18$. Note that Lelle and Flamm can stop the match even if neither of them has n wins.

,

F2. Court Blue (Hard Version)

Input file: standard input
Output file: standard output
Time limit: 4 seconds
Memory limit: 512 megabytes

This is the hard version of the problem. In this version, it is not guaranteed that $n = m$, and the time limit is higher. You can make hacks only if both versions of the problem are solved.

In the court of the Blue King, Lelle and Flamm are having a performance match. The match consists of several rounds. In each round, either Lelle or Flamm wins.

Let W_L and W_F denote the number of wins of Lelle and Flamm, respectively. The Blue King considers a match to be **successful** if and only if:

- after every round, $\gcd(W_L, W_F) \leq 1$;
- at the end of the match, $W_L \leq n, W_F \leq m$.

Note that $\gcd(0, x) = \gcd(x, 0) = x$ for every non-negative integer x.

Lelle and Flamm can decide to stop the match whenever they want, and the final score of the performance is $l \cdot W_L + f \cdot W_F$.

Please help Lelle and Flamm coordinate their wins and losses such that the performance is **successful**, and the total score of the performance is maximized.

Input

The first line contains an integer t ($1 \leq t \leq 10^3$) — the number of test cases.

The only line of each test case contains four integers n, m, l, f ($2 \leq n \leq m \leq 2 \cdot 10^7$, $1 \leq l, f \leq 10^9$): n, m give the upper bound on the number of Lelle and Flamm's wins, l and f determine the final score of the performance.

Unusual additional constraint: it is guaranteed that, for each test, there are no pairs of test cases with the same pair of n, m.

Output

For each test case, output a single integer — the maximum total score of a **successful** performance.

Standard Input	Standard Output
8 3 4 2 5 4 4 1 4 6 6 2 2 7 9 2 3 8 9 9 1 2 7 1 4 5 9 1 4 5 6 6 7	22 17 18 37 77 30 41 59
2 3082823 20000000 1341 331 20000000 20000000 3 5	10754065643 159999991
1 139 1293 193 412	559543

Note

In the first test case, a possible performance is as follows:

- Flamm wins, $\gcd(0, 1) = 1$.
- Lelle wins, $\gcd(1, 1) = 1$.

- Flamm wins, $\gcd(1, 2) = 1$.
- Flamm wins, $\gcd(1, 3) = 1$.
- Flamm wins, $\gcd(1, 4) = 1$.
- Lelle and Flamm agree to stop the match.

The final score is $1 \cdot 2 + 4 \cdot 5 = 22$.

,

G. Lattice Optimizing

Input file: standard input
Output file: standard output
Time limit: 7 seconds
Memory limit: 1024 megabytes

Consider a grid graph with n rows and n columns. Let the cell in row x and column y be (x, y). There exists a directed edge from (x, y) to $(x + 1, y)$, with non-negative integer value $d_{x,y}$, for all $1 \le x < n, 1 \le y \le n$, and there also exists a directed edge from (x, y) to $(x, y + 1)$, with non-negative integer value $r_{x,y}$, for all $1 \le x \le n, 1 \le y < n$.

Initially, you are at $(1, 1)$, with an empty set S. You need to walk along the edges and eventually reach (n, n). Whenever you pass an edge, its value will be inserted into S. Please maximize the MEX* of S when you reach (n, n).

* The MEX (minimum excluded) of an array is the smallest non-negative integer that does not belong to the array. For instance:

- The MEX of $[2, 2, 1]$ is 0, because 0 does not belong to the array.
- The MEX of $[3, 1, 0, 1]$ is 2, because 0 and 1 belong to the array, but 2 does not.
- The MEX of $[0, 3, 1, 2]$ is 4, because $0, 1, 2$, and 3 belong to the array, but 4 does not.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 100$). The description of the test cases follows.

The first line of each test case contains a single integer n ($2 \le n \le 20$) — the number of rows and columns.

Each of the next $n - 1$ lines contains n integers separated by single spaces — the matrix d ($0 \le d_{x,y} \le 2n - 2$).

Each of the next n lines contains $n - 1$ integers separated by single spaces — the matrix r ($0 \le r_{x,y} \le 2n - 2$).

It is guaranteed that the sum of all n^3 does not exceed 8000.

Output

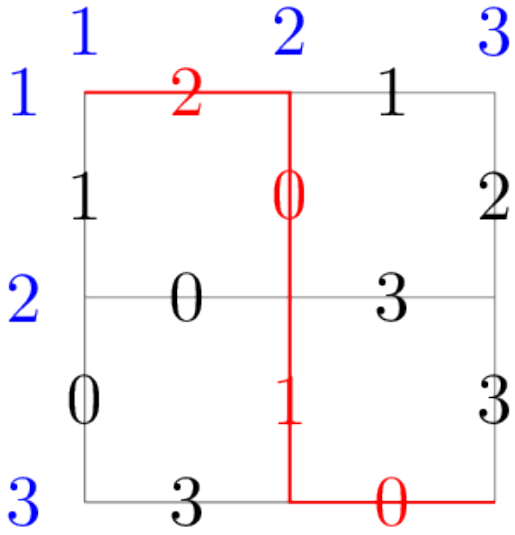
For each test case, print a single integer — the maximum MEX of S when you reach (n, n).

Standard Input	Standard Output
2 3 1 0 2 0 1 3 2 1 0 3 3 0 3 1 2 0 0 1 2 2 0 1 2 0 1	3 2
1 10 16 7 3 15 9 17 1 15 9 0 4 3 1 12 13 10 10 14 6 12 3 1 3 9 5 16 0 12 7 12 11 4 8 7 13 7 15 13 9 2 2 3 9 9 4 12 17 7 10 15 10 6 15 17 13 6 15 9 4 9 13 3 3 14 1 2 10 10 12 16 8 2 9 13 18 7 1 6 2 6	14

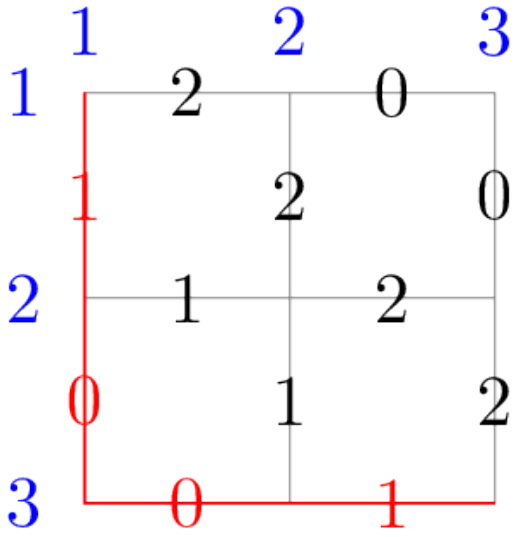
15 12 2 6 0 0 13 3 7 17 7 3 17 17 10 15 12 14 15 4 3 3 17 3 13 11 16 6 16 17 7 7 12 5 2 4 10 18 9 9 3 5 9 1 16 7 1 0 4 2 10 10 12 2 1 4 14 15 16 15 5 8 4 18 7 18 10 11 2 0 14 8 18 2 17 6 0 9 6 13 5 11 5 15 7 11 6 3 17 14 5 1 3 16 16 13 1 0 13 11	

Note

In the first test case, the grid graph and one of the optimal paths are as follows:



In the second test case, the grid graph and one of the optimal paths are as follows:



,

H. Counting 101

```
Input file:     standard input
Output file:    standard output
Time limit:     10.1 seconds
Memory limit:   1010 megabytes
```

It's been a long summer's day, with the constant chirping of cicadas and the heat which never seemed to end. Finally, it has drawn to a close. The showdown has passed, the gates are open, and only a gentle breeze is left behind.

Your predecessors had taken their final bow; it's your turn to take the stage.

Sorting through some notes that were left behind, you found a curious statement named **Problem 101**:

- Given a positive integer sequence $a_1, a_2, \ldots, a_n$, you can operate on it any number of times. In an operation, you choose three consecutive elements a_i, a_{i+1}, a_{i+2}, and merge them into one element $\max(a_i + 1, a_{i+1}, a_{i+2} + 1)$. Please calculate the maximum number of operations you can do without creating an element greater than m.

After some thought, you decided to propose the following problem, named **Counting 101**:

- Given n and m. For each $k = 0, 1, \ldots, \lfloor \frac{n-1}{2} \rfloor$, please find the number of integer sequences $a_1, a_2, \ldots, a_n$ with elements in $[1, m]$, such that when used as input for **Problem 101**, the answer is k. As the answer can be very large, please print it modulo $10^9 + 7$.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^3$). The description of the test cases follows.

The only line of each test case contains two integers n, m ($1 \le n \le 130$, $1 \le m \le 30$).

Output

For each test case, output $\lfloor \frac{n+1}{2} \rfloor$ numbers. The i-th number is the number of valid sequences such that when used as input for **Problem 101**, the answer is $i - 1$, modulo $10^9 + 7$.

Standard Input	Standard Output
2 3 2 10 10	6 2 1590121 23399118 382293180 213020758 379696760

Note

In the first test case, there are $2^3 = 8$ candidate sequences. Among them, you can operate on $[1, 2, 1]$ and $[1, 1, 1]$ once; you cannot operate on the other 6 sequences.