Problem A. Mio visits ACGN Exhibition

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

One day, Mio visits an Animation Comic Game Novel (ACGN) Exhibition, and she would like to buy some presents for Ritsu.

We assure that the space of the exhibition is a $n \times m$ grid, called the grid A, and each cell in the grid represents a stall, only selling present 0 or 1. In other words, every cell of the $n \times m$ grid A is filled with 0 or 1.

Under the control policy for containing COVID-19, there are some restrictions on visiting route.

We define a SAFE PATH as a path from the top left cell $(1, 1)$, to the bottom right cell (n, m), and if you are in the cell (x, y), then you can only travel to the cells $(x + 1, y)$ or $(x, y + 1)$. Every visitor has to visit the exhibition through SAFE PATH, so does Mio.

The two paths are considered to be different if and only if at least one cell passed differs.

Mio wonders how many different SAFE PATHs, which have some 0s and 1s, and the number of 0 is at least p, the number of 1 is at least q.

Since the answer may be too large, you only need to output the result modulo 998244353.

Input

The first line contains four integers, n, m, p, q $(1 \le n, m \le 500, 0 \le p, q \le 10000)$.

Each of the next n lines contains m space separated integers $A_{i,j}$ $(0 \le A_{i,j} \le 1)$, denoting the number in the cell (i, j) of the grid A.

Output

Print a single integer, denoting the answer to the question, modulo 998244353.

Examples

standard input	standard output

`2 2 1 1`
`0 0`
`1 1`
	`2`

`3 3 2 0`
`0 0 1`
`0 0 1`
`1 0 0`
	`6`

Problem B. Continued Fraction

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

A continued fraction is an expression of the form:

$$a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{\ddots + \cfrac{1}{a_n}}}}$$

where $a_0, a_1, \ldots, a_n$ are nonnegative integers.

Given a fraction $\frac{x}{y}$(x, y are positive integers), please expand it into a continued fraction.

Input

The first line contains an integer T $(1 \le T \le 10^3)$, denoting the number of test cases.

The only line of each test case contains two integers x, y $(1 \le x, y \le 10^9)$, denoting a fraction $\frac{x}{y}$. It's guaranteed that $\gcd(x, y) = 1$.

Output

For each test case, output one line: first an integer n denoting the height of the continued fraction, then $n + 1$ integers denoting $a_0, \ldots, a_n$. Your solution should gurarantee that $0 \le n \le 100$, $0 \le a_i \le 10^9$.

If there are multiple valid solutions, you only need to output one of them.

Example

`standard input`	`standard output`

`2`
`105 38`
`1 114`
	`4 2 1 3 4 2`
	`1 0 114`

Note

For the convenience of you, we give explanation of sample:

$$\frac{105}{38} = 2 + \cfrac{1}{1 + \cfrac{1}{3 + \cfrac{1}{4 + \cfrac{1}{2}}}}$$

$$\frac{1}{114} = 0 + \frac{1}{114}$$

Problem C. Crystal Caves

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

One morning, Cirno woke up drowsy Daiyousei very early, and pointed out not far away with excitement.

「Hurry up, I heard, Under the ice there hides ancient relics」

「Wumm......」

As everyone knows, In Gensokyo, Rumors are the truth. Beside the ancient relics, there also buries a gem with incredible magic - 「Cristal of Cyan Heart」. However, what connecting the ice and relics is a large crystal cave.

Crystal caves contain n floors, and the floor near the ground is the narrowest. So we call it the first floor. After that, each floor is more spacious than the previous floor.

Abstractly, we define that the y-axis of i-th floor is $-i$, and the x-axis of it can be described with a closed interval $[l_i, r_i]$ ($l_i < l_{i-1} < r_{i-1} < r_i$). Moreover, The ground is $[l_0, r_0] = [0, 0]$.

In order to break the boundary between the present world and the relics, Cirno should draw support from the magic of crystal in the caves.

Specifically, she will choose a location on each floor of the crystal cave to activate crystal. Then the total magic value is the sum of Manhattan distances between all pairs of activated crystals.

To be on the safe side, Cirno decides to maximize the magic value.

And you are arranged to calculate the maximum of magic value to help her.

Input

The first line contains only one integer n.

And in the following n lines, each line contains two integers representing l_i, r_i, separated by blank.

$1 \le n \le 2000,\ -10^9 < l_n < \cdots < l_1 < 0 < r_1 < \cdots < r_n < 10^9$.

Output

The output contains one number that representing the answer.

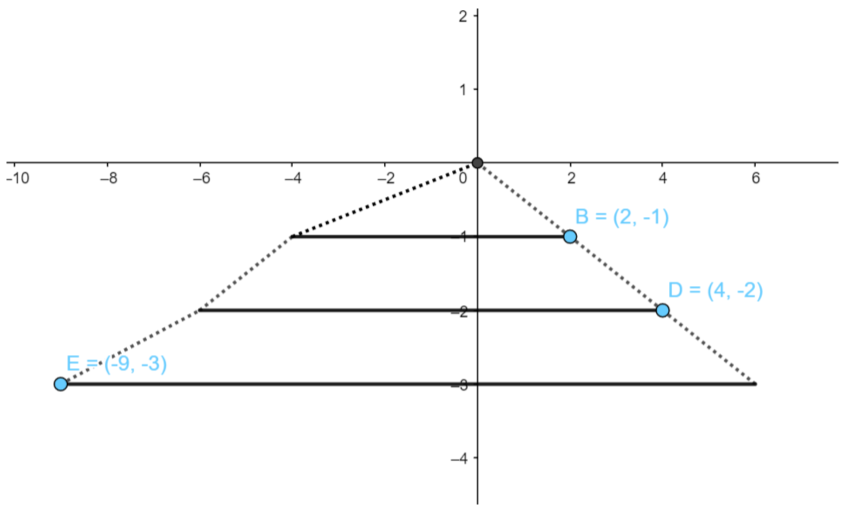
Examples

standard input	standard output

3
-4 2
-6 4
-9 6
	30

4
-3 3
-6 5
-7 8
-9 9
	70

Note



Sample 1

Dashed outline represent the section of crystal caves, and the black segments mean the region where Cirno can active crystals.

One of the Optimal placement scheme is marked by blue points.

Problem D. Character Distance

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

You are given an array a of length n and you need to rearrange the array to satisfy that there exists at least one number x that appears at least once in the array and for each pair i, j $(1 \le i < j \le n, a_i = a_j = x)$ that satisfy $d \le j - i$.

If there is no such array exist, please output -1. Otherwise, output the array rearranged with the minimum lexicographical order.

Input

The first line contains one integer T $(1 \le T \le 10^6)$, denoting the number of test cases.

Each test case contains two lines.

The first line contains two integers n, d $(1 \le n, d \le 10^6)$ denoting the length of array a and the minimum distance defined in the statement above.

The second line contains n integers, the i^{th} integer $a_i (1 \le a_i \le n)$ denotes the i^{th} number of array a.

It is guaranteed that the sum of n in all test cases does not exceed 10^6.

Output

For each test case output one line.

If there is no solution output -1 in one line, otherwise output n integers as the solution array with minimum lexicographical order.

Example

standard input	standard output

4
4 3
1 2 1 2
4 4
1 1 2 2
6 3
3 3 2 2 1 1
7 3
1 1 2 2 2 3 3
	1 2 2 1
	-1
	1 1 2 3 3 2
	1 1 2 3 2 2 3

Problem E. The Legend of God Flukehn in Eastern

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

Flukehn is a well-known god in Eastern and he is invincible with 5 ranks in Go. To challenge the legend of god Flukehn in Eastern, you decided to compose a modified Shogi (the Japanese variant of chess) match with Flukehn.

The chess board can be viewed as an infinite two-dimensional plane. Initially you have n Pawns uniquely occupying an integral point on the board, and Flukehn has one Gold general occupying an integral point either.

The game is turn-based. At each turn,

1. you shall pick a Pawn and move it down one unit. More formally, you can move one Pawn with original coordinate (x, y) to $(x, y - 1)$. **Noted that you can't have two Pawns on the same point at any moment**.
2. Then Flukehn shall choose his Gold general to move up, down, left, right, up-left or up-right to the nearest integral point. More formally, with original coordinate (x, y), Flukehn shall choose his Gold general to move to $(x, y + 1)$, $(x, y - 1)$, $(x - 1, y)$, $(x + 1, y)$, $(x - 1, y + 1)$ or $(x + 1, y + 1)$.

No one can skip his own turn.

At any moment when Gold general and a Pawn are on the same integral point, the Pawn is considered defeated by Flukehn and should be picked out from the board (Even if it is on your turn).

The initial coordinate of Flukehn's Gold general is on $(0, 0)$.

There is no limitation on turns of the game.

You aim to minimize the number of defeated Pawns while Flukehn aim to maximize it and you both play **optimally** in the game.

What is final number of Pawns defeated in finite number of turns?

Input

First line contains one integer T $(1 \le T \le 10^6)$, denoting the number of test cases.

For every test case, the first line contains one integer n $(1 \le n \le 10^6)$ as the number of Pawns you initially own.

In the following n lines, the i^{th} line contains two integers x_i, y_i $(-10^9 \le x_i, y_i \le 10^9)$ as the coordinates of the i^{th} Pawn. **It is guaranteed that no two Pawns are on the same point and no Pawn is at** $(0, 0)$ **initially.**

The sum of n for all test cases does not exceed 10^6.

Output

For every test case, output one line containing one integer as the number of Pawns defeated in a finite number of turns.

Example

standard input	standard output

2
2
1 1
2 4
3
1 1
2 4
-1 -1
	2
	2

Note

In the second example, the third Pawn can continue to move down allowing it to escape from Gold general.

Problem F. Four Column Hanoi Tower

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	256 megabytes

Based on the classical problem of tower of Hanoi, there are four rods indexed by A,B,C,D (the only difference between this problem and the classical one) and N disks of various diameters, which can slide onto any rod. The puzzle begins with disks stacked on one rod in order of decreasing size, the smallest on the top, thus approximating a conical shape. The objective of the puzzle is to move the entire stack to the last rod (indexed by D), obeying the following rules:

- Only one disk may be moved at a time.
- Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack or on an empty rod.
- No disk may be placed on top of a disk that is smaller than it.

You need to calculate the minimum number of moves required to solve the problem.

Input

The first line is a positive integer $T(1 \leq T \leq 10000)$, indicating that there are T test data. Next, there are T lines. Each line has a positive integer $N(1 \leq N \leq 10000)$, indicating the number of plates in each of a test data.

Output

Each test data outputs a line as a positive integer, that is, the minimum number of steps required to move all N plates from the first column A to the last column D.

Example

standard input	standard output
5 1 2 3 4 5	1 3 5 9 13

Problem G. Magic Number Group

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

SevenFireflie has a sequence of positive integers with a length of n and quickly calculates all the factors of each number. In order to exercise his factor calculation ability, he has selected q consecutive subsequences from the sequence and found a positive integer p greater than 1 for each subsequence, so that p can divide as many numbers in this subsequence as possible. He has also found that p may have more than one.

So the question is, how many numbers in each subsequence can be divided at most?

Input

The first line contains an integer T $(1 \le T \le 5 \times 10^4)$, indicating that there is T test cases next.

The first line of each test cases has two positive integers n $(1 \le n \le 5 \times 10^4)$, q $(1 \le q \le 5 \times 10^4)$.

Next line n integers a_i $(1 \le i \le n, 1 \le a_i \le 1 \times 10^6)$, which representing the numbers in this sequence. The two adjacent numbers are separated by a space.

Each of the next q lines contains two integers l, r $(1 \le l \le r \le n)$, representing a subsequence being queried, $a_l, a_{l+1}, \cdots, a_r$, and l, r are separated by a space.

The input guarantees that the sum of n does not exceed 5×10^4 and the sum of q does not exceed 5×10^4.

Output

For each test case, output q lines, each line contains a positive integer, indicating the answer.

Example

standard input	standard output

1
10 5
20 15 6 1 21 12 2 3 17 9
1 4
2 5
3 6
4 7
5 10
	2
	3
	3
	2
	4

Problem H. Hearthstone So Easy

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

Hearthstone is a turn-based card game. The game flow of each round is: Player 1 draws card ⇒player 1 plays cards ⇒player 2 draws card ⇒player 2 plays cards.

We simplify the game logic as follows:

- During each player's draw stage, the player attempts to draw a card from his or her deck.
- During each player's playing stage, the player can choose:
  1. to increase his/her health by k points. Note that there is **no** upper limit on health.
  2. to reduce the opponent's health by k points.

When there are no cards in the player's card deck, the player will enter a state of **fatigue**. At this time, the player will increase his/her **fatigue value** by one every times he/she tries to draw a card, and then deduct the amount of health by **the fatigue value**. The **fatigue value** of each player is initially 0 points.

pllj and freesin like playing hearthstone very much. In a certain game, both players have no cards in their decks, and both the fatigue points are 0 points, and the health points are both n points. When a player's health is less than or equal to 0, the player **immediately** loses the game.

At this time, it's pllj's turn to draw card. Both players are very smart, so they play the game optimally. Who will be the winner? Please output his name.

Input

The first line contains a single integer t ($1 \le t \le 10^5$), which represents the number of data cases.

Each group of data is a row of two positive integers n, k separated by spaces ($1 \le n, k \le 10^9$), of which meaning is described before.

Output

For each case of data, output a line of string `pllj` or `freesin` to indicate the winner.

Example

standard input	standard output

2
10 9
5 3
	pllj
	freesin

Note

For the first data case:

- pllj's draw stage: pllj tries to draw cards from a empty deck. His fatigue value increases by 1 to become 1, and then pllj's health deducts by one point, leaving 9 health points.
- pllj's playing stage: pllj causes 9 points of damage to freesin. After this time, freesin has 1 point of life left.

- freesin's draw stage: freesin tries to draw cards from a empty deck. His fatigue value increases by 1 to become 1, and then freesin's health deducts by one point, leaving 0 points. At this time, freesin loses the game.

Problem I. Homework

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	256 megabytes

Cirno is a strict teacher, while Wings is a naughty student.

Cirno assigns a huge amount of homework to n of her students, including Wings every day. Wings wants to finish the homework as soon as possible. He calls on the classmates to copy homework from each other. Due to the different abilities of these students, the time it takes for each to finish the homework independently may be different. After discussion, in order to make everyone finish the homework as soon as possible, these students developed a solution: one can ask other who has finished the homework for answers and then just copy that. Cirno is too strict with her students, she has confiscated all the Internet-capable devices of her students, so these students must copy the homework in a more primitive way — going to another student's home.

There are $n-1$ bidirectional roads between the homes of n students and each road connects two students' homes. Any two students can reach each other's home directly or indirectly. If student i intends to copy student j's homework, he or she must wait for student j to finish the homework, then leaves for student j's home and copies the homework, and finally returns home (**The homework is considered to be finished only when the student returns home**). The time taken for student i to leave for student j's home, copy the homework, and finally return home equals to the distance between student i's home and student j's home.

Soon, Cirno discovered Wings's tricks. She was very angry, so she planned to calculate the earliest time for each student to finish the homework, and increase the amount of homework according to the situation as a punishment. However, the students' learning status and the road lengths will change. Cirno asks you to help her to calculate the earliest time for each student to finish the homework in different situation.

Input

The first line contains two positive integers n, q ($1 \le n, q \le 10^5$). n denotes the number of students. q denotes the total number of changes and queries.

The next line contains n space separated integers a_i ($0 \le a_i \le 10^9$), denoting the time for student i to finish the homework independently.

The next $n-1$ lines contain the description of roads between students' homes. Each line contains three integers u, v, w ($1 \le u, v \le n, 0 \le w \le 10^9$), denoting that there is a road with length w, connecting student u's home and student v's home.

The next q lines contain descriptions of changes and queries. Each line describes one type of change or a single query.

The first type of change is described by three integers op, i, x ($op = 1, 1 \le i \le n, 0 \le x \le 10^9$), which means the time for student i to finish the homework independently changes to x;

The second type of change is described by three integers op, i, w ($op = 2, 1 \le i < n, 0 \le w \le 10^9$), which means the length of the i-th road in the input changes to w.

The query is described by one integer op ($op = 3$), which means you need to calculate the earliest time for each student to finish the homework.

It's guaranteed that any two students can reach each other's home directly or indirectly. And it's guaranteed that the number of queries does not exceed 200.

Output

For each query, print an integer in one line. Let t_i denotes the earliest time for student i to finish the homework. The integer you need to print is $t_1 \oplus t_2 \oplus \cdots \oplus t_n$ ($\oplus$ denotes the bitwise XOR operation).

Example

standard input	standard output

4 5
4 4 2 7
2 1 8
3 1 9
4 3 1
3
1 1 1
3
2 1 1
3
	1
	4
	2

Note

For the first query, student 1, student 2 and student 3 finish their homework independently, and the times are $t_1 = 4$, $t_2 = 4$, $t_3 = 2$. Student 4 goes to student 3's home after student 3 finished the homework to copy homework. The time for student 4 to finish the homework is $t_4 = 2 + 1 = 3$.

For the second query, the earliest times for the four students to finish the homework are $t_1 = 1$, $t_2 = 4$, $t_3 = 2$, $t_4 = 3$.

For the second query, the earliest times for the four students to finish the homework are $t_1 = 1$, $t_2 = 2$, $t_3 = 2$, $t_4 = 3$.

Problem J. LRU

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

It costs a long time for CPUs to access data from memory, so most of CPUs have caches where requests for data can be served faster. To be cost-effective and to enable efficient use of data, caches must be relatively small. Therefore, we must use some policies to choose some data wisely and storage them in the cache. We divide the memory into many blocks which have the same size and index them from 1 to 10^9, and every block has an unique index. A cache will have a capacity for K blocks, which means it can storage at most K blocks simultaneously. A cache hit occurs when the requested block is available in the cache, or we say a cache miss occurs. Now we introduce a **LRU** (Least Recently Used) placement policy on a fully associative cache.

1. If the requested block is available in the cache, a cache hit occurs.
2. If not, CPU can only access the block from the memory and write the block into the cache. If cache is not full, append the block into the cache.
3. If the cache is full, cache is full, the block which haven't been visited for the longest time in the cache will be replaced by the new block.

An example for cache with capacity of 3 blocks is shown below.

Blocks Index	3	4	2	6	4	3	7	4
Cache #1	3	3	3	6	6	6	7	7
Cache #2		4	4	4	4	4	4	4
Cache #3			2	2	2	3	3	3
Hit	×	×	×	×	√	×	×	√
Blocks Index	3	6	3	4	8	4	6	
Cache #1	7	6	6	6	8	8	8	
Cache #2	4	4	4	4	4	4	4	
Cache #3	3	3	3	3	3	3	6	
Hit	√	×	√	√	×	√	×	

The 8^{th}, 4^{th} and 3^{rd} are in the cache when the 6^{th} block is requested, so a cache miss occurs. At that time, the cache is full, and we must decide the block to be replaced. The most recent request of 8^{th} block is the 13^{th} request, the most recent request of 4^{th} block is the 14^{th} request and the most recent request of 3^{rd} block is the 11^{th} request. The 3^{rd} block will be replaced by the 6^{th} block because the 3^{rd} block hasn't been visited for longest time.

Now the sequence of requested blocks and the capacity of the cache are given, please determine the minimum capacity for the cache in order to ensure at least K requests to hit the cache.

Input

The first line contains two integers N and $K(1 \leq K \leq N \leq 10^5)$, denoting the length of sequence and the number of requests required to hit the cache. The second line contains N integers and the i^{th} integers $a_i(1 \leq a_i \leq 10^9)$ denoting the index of the block requested by the i^{th} request.

Output

The output contains only one line. If it is possible to ensure at least K requests to hit the cache, then output a single integer denoting the smallest capacity in blocks, otherwise output "cbddl"(without quotes).

Examples

standard input	standard output
15 6 3 4 2 6 4 3 7 4 3 6 3 4 8 4 6	3
15 5 3 4 2 6 4 3 7 4 3 6 3 4 8 4 6	3
15 10 3 4 2 6 4 3 7 4 3 6 3 4 8 4 6	cbddl

Problem K. Many Littles Make a Mickle

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

Gather sand to form a tower is a Chinese idiom, whose Pinyin is *jù shā chéng tǎ*. It means to pile sand into a pagoda, referring that a little makes a lot. From the *fahua Sutra - convenience products*.

Suppose the tower has N floors. There are $i \times i$ rooms on the i^{th} floor. Each room can accommodate M people. How many people can the N floor tower accommodate?

Input

The first line is a positive integer T $(1 \le T \le 100)$, indicating that there are T test data.

Next, there are T lines. Each line has two positive integers N and M $(1 \le N, M \le 100)$, indicating the number of floors of the tower and the number of people that can be accommodated in each room in a test data.

Output

Each test data outputs a line containing one positive integer, that is, the total number of people that can be accommodated in the N floor tower.

Example

standard input	standard output

2
2 2
3 3
	10
	42

Problem L. It Rains Again

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

Suppose that in a Cartesian coordinate system on an infinite plane, the x-axis represents the ground and there are n rainscreens (We don't consider their thickness.) in the sky.

Every rainscreen can be described as a segment which starts from the (x_1, y_1) and ends at (x_2, y_2). Now if it starts to rain from the infinite high sky, I want you to tell me how many units of the x-axis will not get rained on.

To simplify the problem,the rain can only move vertically whenever it falls.

Note that two rainscreens can overlap and intersect with each other, and there is no rainscreen which is placed vertically.

Input

The first line contains one positive integer $n (1 \le n \le 100000)$.

Each of the next n lines contains four positive integers $x_1, y_1, x_2, y_2 (1 \le x_1 < x_2 \le 100000, 1 \le y_1, y_2 \le 100000)$, representing a rainscreen which starts from the (x_1, y_1) and ends at (x_2, y_2).

Output

The only one integer - the number of units of the x-axis which will not get rained on.

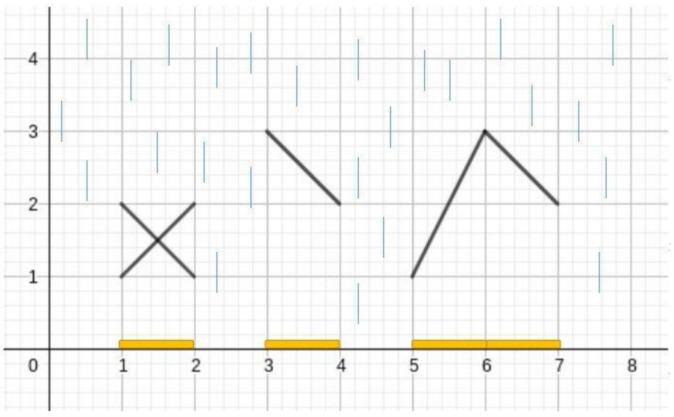
Example

standard input	standard output

5
1 2 2 1
1 1 2 2
3 3 4 3
5 1 6 3
6 3 7 2
	4

Note

Consider the example, we can draw a picture like below:



It's easy to calculate the answer is 4.