Problem A. Easy Diameter Problem

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2.5 seconds
Memory limit:	1024 megabytes

Randias is given a tree with n vertices. He does the following operation until the tree contains 0 vertices:

- choose a vertex which is an endpoint of any **diameter**, and then remove it.

He asks you to find the number of removal orders modulo $10^9 + 7$.

Note that two orders are considered different if and only if there exists i ($1 \le i \le n$), where the i-th vertex being removed in one order is different from the i-th vertex being removed in the other order.

Recall that a vertex u is an endpoint of some diameter if there exists a vertex v such that $\texttt{dis}(u, v) \ge \texttt{dis}(i, j)$ for any pair of vertices i and j, where $\texttt{dis}(x, y)$ represents the number of edges in the shortest path from x to y.

Input

The first line contains one integer n ($1 \le n \le 300$), denoting the number of vertices of the tree.

The following $n - 1$ lines, each line contains two integers u and v ($1 \le u, v \le n$, $u \ne v$), denoting an edge connecting u and v.

It is guaranteed that the edges form a tree.

Output

Print a single integer, denoting the number of removal orders modulo $10^9 + 7$.

Examples

standard input	standard output

3
1 2
3 2
	4

5
4 1
4 5
1 2
1 3
	28

7
5 7
2 5
2 1
1 6
3 6
4 1
	116

Note

For the first example, $[1, 2, 3]$, $[1, 3, 2]$, $[3, 1, 2]$, $[3, 2, 1]$ are possible removal orders.

Problem B. The Game

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

Randias is playing a game. He is given two multisets (a set which can contain duplicate elements) A and B, consists of n and m integers $A_1, A_2, \ldots, A_n$, $B_1, B_2 \ldots, B_m$ repsectively. He can perform the following operation any number of times:

- Choose one element x where $x \in A$, change x to $x + 1$. Then remove the minimal element in A, Note that if there are more than one minimal element in A, we only remove one of them.

For example, if $A = \{4, 4, 5, 5, 6\}$ and we choose $x = 5$, A will become $\{4, 5, 6, 6\}$ after one operation.

He wants to know whether he can make $A = B$ after several (possibly zero) operations? If he can make it, please help him to determine which operations need to be performed.

Two multisets are consider to be the same, if for all x, the number of occurrence of x in A and B are the same.

Input

The first line contains t $(1 \le t \le 5 \cdot 10^4)$, representing the number of testcases.

For each testcase, the first line contains two integers n, m $(1 \le m \le n \le 3 \cdot 10^5)$, representing the size of two multisets.

The following line contains n integers $A_1, A_2, \ldots, A_n$ $(1 \le A_i \le 10^9)$.

The following line contains m integers $B_1, B_2, \ldots, B_m$ $(1 \le B_i \le 10^9)$.

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

Output

For each test case, if Randias cannot let $A = B$, output "-1".

Otherwise, on the first line, output the number of operations L $(0 \le L \le n)$ Randias needs to make.

The following line contains L integers $x_1, x_2, \ldots, x_L$, representing the integer x each operation choose. You must ensure that $x \in A$ before each operation.

If there are multiple solutions, output any.

Example

standard input	standard output

6
5 3
1 2 2 3 3
2 3 4
4 2
1 2 2 4
2 4
5 2
2 3 3 4 4
5 5
6 1
1 1 1 1 1 1
4
4 2
1 1 1 2
2 2
4 1
1 1 1 1
2
	2
	1 3
	-1
	3
	2 4 4
	5
	1 1 1 2 3
	2
	1 1
	-1

Problem C. Master of Both IV

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Prof.Chen is the master of arithmetic operations and binary operations. Today's homework for his students, Putata and Budada, is to find the number of non-empty subsequences $\{i_1, i_2, \ldots, i_m\}$ $(1 \le i_1 < i_2 < i_3 \cdots < i_m \le n, 1 \le m \le n)$ of sequence $\{1, 2, \ldots, n\}$ satisfying that $\forall x \in [1, m], a_{i_x} | \bigoplus_{j=1}^{m} a_{i_j}$, where $\{a_n\}$ is a given sequence.

Here $\oplus$ means bitwise exclusive-or operation, $\bigoplus_{j=1}^{m} a_{i_j}$ equals to the bitwise exclusive-or of all elements a_{i_j} for $1 \le j \le m$. We say $x|s$ if and only if there exists an non-negative integer k such that $s = k \cdot x$.

Please help Putata and Budada finish their homework. In order to ruin the legends, please output the answer modulo 998 244 353.

Input

The first line contains one integer t $(1 \le t \le 2 \cdot 10^5)$, denoting the number of test cases.

For each test case, the first line contains one integer n $(1 \le n \le 2 \cdot 10^5)$, denoting the length of the sequence.

The second line contains n integers, the i-th integer is a_i $(1 \le a_i \le n)$, denoting the i-th element in the sequence. It is **possible** that $a_i = a_j$ for $i \ne j$.

It is guaranteed that the sum of n over all testcases does not exceed $2 \cdot 10^5$.

Output

For each test case, output one integer in one line, denoting the answer.

Example

standard input	standard output

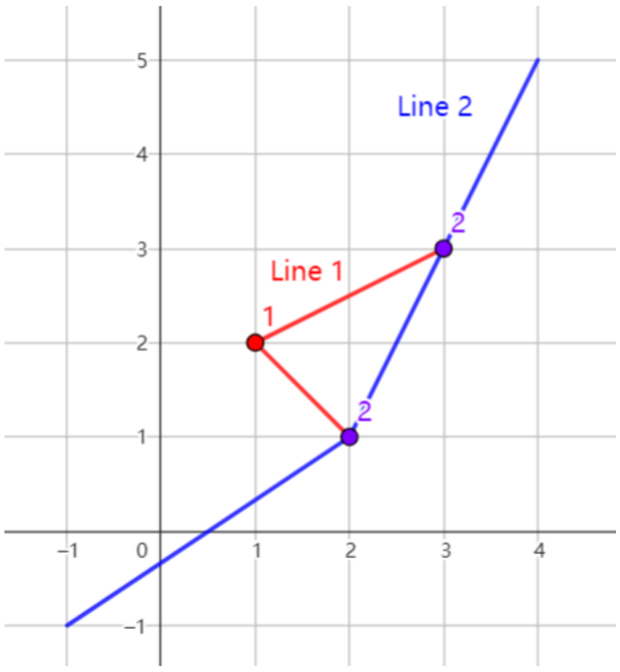
2
3
1 2 3
5
3 3 5 1 1
	4
	11

Problem D. Subway

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

Grammy is planning to build subway in Pigeland. She choosed n stations and decided to connect them with the fewest number of subway lines. Each subway line consists of one or more consecutive segments and passes through one or more stations. A subway line should not pass through a station twice. To make things more interesting, Grammy decided to make exactly a_i subway lines pass through the i-th station in the final subway plan.

Formally, each subway line can be described using a series of points $p_1, p_2, \ldots, p_L (L \geq 2)$, and the full span of the subway line is the union of segments $[p_1, p_2], [p_2, p_3], \ldots, [p_{L-1}, p_L]$. Each of the stations located on the subway line should be at point p_i for some $i \in [1, L]$. Note that the points p_i are not necessarily subway stations.



In this picture (illustrating the example test case), there are two subway lines. Line 1 passes through 3 stations and line 2 passes through 2 stations. The purple stations with a 2 on the corner are passed through by 2 subway lines, and the red station with a 1 on the corner are passed through by 1 line.

Due to budget limitations, a subway line should not self-intersect, and two subway lines should not intersect outside the stations.

Please help Grammy to find a subway plan using the minimum possible number of subway lines. If there are multiple solutions, output any.

To avoid precision issues, your subway plan should only contain integral points as the segments' ends.

Input

The first line contains an integer n $(1 \leq n \leq 50)$, denoting the number of stations.

Each of the following n line contains 3 integers x_i, y_i, a_i $(-10^3 \leq x_i, y_i \leq 10^3, 1 \leq a_i \leq 50)$, denoting that the i-th station is located at position (x_i, y_i), and should be passed through by exactly a_i subway lines.

It is guaranteed that the stations have distinct coordinates.

Output

The first line should contain the minimum possible number of subway lines k.

Each of the following k lines contains the description of a subway line in the format below:

The first integer in the line L $(2 \le L)$, denoting the total number of endpoints in this subway line.

Followed by L pairs of integers $x_i, y_i (-10^9 \le x_i, y_i \le 10^9)$ denoting the endpoints of this subway line.

The sum of L in your output should not exceed 10^4.

Examples

standard input	standard output

3
1 2 1
2 1 2
3 3 2
	2
	3 2 1 1 2 3 3
	4 -1 -1 2 1 3 3 4 5

1 1 1 1	1 5 0 0 1 1 2 2 4 4 8 8

Problem E. Prefix Mahjong

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

A positive integer multiset s is a “Pong” if $s = \{x, x, x\}$ for some positive integer x.

A positive integer multiset s is a “Chow” if $s = \{x, x + 1, x + 2\}$ for some positive integer x.

A positive integer multiset s is an “Eyes” if $s = \{x, x\}$ for some positive integer x.

A positive integer sequence is a “Mahjong” if it can be divided into some (possibly zero) “Pong”s, some (possibly zero) “Chow”s, and **exactly one** “Eyes”.

For example, sequence $s = \{1, 1, 4, 5, 1, 4, 4, 3\}$ is a “Mahjong” because it can be divided into $\{1, 1, 1\}$, $\{4, 5, 3\}$, $\{4, 4\}$.

For each prefix of a given positive integer sequence, determine if it is a “Mahjong”.

Input

Each test contains multiple test cases. The first line contains a single interger t ($1 \le t \le 10^5$), denoting the number of test cases.

For each test case, the only line contains an integer n ($1 \le n \le 10^5$) and the following n positive integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$), denoting the length of the integer sequence and the elements of the positive integer sequence, respectively.

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

Output

For each test case, print a string consisting of ‘0’ and ‘1’ in one line. The i-th character is ‘1’ if the prefix of length i is a “Mahjong”; otherwise it is ‘0’.

Examples

standard input	standard output

4
8 1 1 4 5 1 4 4 3
14 1 1 3 5 4 2 5 5 4 6 6 2 2 4
17 3 5 3 2 2 3 3 1 4 3 1 3 3 5 2 4 4
8 2 4 11 11 14 8 6 3
	01000001
	01001001000001
	00000000001000001
	00000000

10
2 1 1
3 1 1 1
3 1 2 3
5 1 1 1 1 1
5 1 1 1 2 2
5 1 1 1 2 3
8 1 1 1 1 1 1 2 3
5 2 2 1 1 1
5 3 2 1 1 1
8 3 2 1 1 1 1 1 1
	01
	010
	000
	01001
	01001
	01001
	01001001
	01001
	00001
	00001001

Problem F. Redundant Towers

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	1024 megabytes

There are n wireless communication towers in Byteland, labeled by $1, 2, \ldots, n$. The i-th tower is located at (x_i, y_i). No two towers share the same x-coordinate, and no two towers share the same y-coordinate. The transmission radius of each tower is R. In other words, the i-th tower can send a message to the j-th tower in one jump if and only if $\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \le R$. Initially, all the towers are in use.

The a-th ($1 \le a \le n$) tower is considered to be redundant if and only if it satisfies all the following conditions:

- The a-th tower is in use.
- For every pair of towers b and c ($1 \le b < c \le n$, $b \ne a$, $c \ne a$), if they are both in use, and b can send message directly or indirectly to c, then b can also send message directly or indirectly to c without passing through the a-th tower. Note that messages can only be transmitted by towers in use.

You are required to perform q operations. In each operation, you will be given an integer k ($1 \le k \le n$), denoting the label of the tower whose status is changed. If the k-th tower is in use, it will now become not in use, and if it's not in use, it will now become in use. After each operation, you are required to count the number of redundant towers.

Input

The first line contains two integers n and R ($1 \le n \le 10^5$, $2 \le R \le 5$), denoting the number of towers and the transmission radius.

In the next n lines, the i-th line contains two integers x_i and y_i ($1 \le x_i, y_i \le n$), denoting the location of the i-th tower. It is guaranteed that no two towers share the same x-coordinate, and no two towers share the same y-coordinate.

The next line contains a single integer q ($1 \le q \le 10^5$), denoting the number of operations.

Each of the next q lines contains a single integer k ($1 \le k \le n$), denoting an operation. Let *last* be the previous number of redundant towers that you answered. Note that *last* should be reset to 0 at the beginning of the input. For each operation, k is encrypted: its actual value is $k \oplus last$. In the expressions above, the symbol "$\oplus$" denotes the bitwise exclusive-or operation. Also, note that the constraints described in the statement above apply to the corresponding parameters only after decryption, the encrypted values are not subject to those constraints.

Output

For each operation, print a single line containing a single integer, denoting the number of current redundant towers.

Example

standard input	standard output

5 3
3 2
4 1
5 4
1 3
2 5
6
1
6
0
3
0
1
	4
	3
	2
	2
	2
	3

Problem G. Hard Brackets Problem

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Putata is coding with his favorite IDE. The best part of this IDE is the parentheses completion function. The function works as follows: assume that $S|T$ is currently displayed on the screen, where | is the current position of the cursor and S and T are two (possibly empty) strings.

- Putata enters left parenthesis '`(`', $S(|)T$ will be displayed on the screen, where | is the new position of the cursor.
- Putata enters right parenthesis '`)`'. If $T =)T'$, which means that T begins with a right parenthesis, then the string will not change, and the cursor will move to the right of the next character, $S)|T'$ will be displayed on the screen, otherwise $S)|T$ will be displayed on the screen.

Putata worked very hard last night and when he wakes up in the morning, he only remembers the code saved on his computer and that he only entered several parentheses. Help him find the parenthesis sequence he typed in order or tell him it is impossible to type this string by only entering parentheses.

Input

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

For each test case, the only line contains one string S ($1 \le |S| \le 10^6$). It is guaranteed that S only consists of parentheses, '`(`' and '`)`'.

It is guaranteed that the sum of $|S|$ over all testcases does not exceed 10^6.

Output

For each test case, if it is possible to type the string by entering parentheses, output one string in one line, denoting the answer. Otherwise, output "impossible" in one line. If there are multiple answers, you can output any of them.

Please notice that you **don't have to minimize** the length of the answer. Your answer should only contain parentheses and the length of your answer should be no greater than the input string for each test case if the answer exists.

Example

standard input	standard output

`3`
`((()))`
`(`
`)))()`
	`(((`
	`impossible`
	`)))(`

Problem H. Sweet Sugar

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

Prof.Chen is practicing baking cakes now. In the garden of his big house, there is an ingredient tree with n vertices, labeled by $1, 2, \ldots, n$. On the i-th vertex of the tree, there are c_i sweet sugars.

A cake will consume exactly k sweet sugars. Every time before baking a new cake, Prof.Chen will come to the garden, select a component (or the whole tree) of vertices from a tree, then cut the component down, and take all the sugars from it. When a component is cut down, the original tree may split into several disconnected new trees. Also, note that it is not a good idea to waste sugars, so Prof.Chen will always make sure there are exactly k sugars in the selected component.

Prof.Chen wants to make as many cakes as possible. Please help Prof.Chen to determine how many cakes he can make.

Input

The first line contains a single integer t ($1 \le t \le 10^6$), the number of test cases. For each test case:

The first line contains two integers n and k ($1 \le n \le 10^6$, $1 \le k \le 2 \cdot 10^6$), denoting the number of vertices and the number of sugars in each cake.

The next line contains n integers $c_1, c_2, \ldots, c_n$ ($0 \le c_i \le 2$), denoting the number of sweet sugars on each vertex.

Each of the following $n - 1$ lines contains two integers u_i and v_i ($1 \le u_i, v_i \le n$, $u_i \ne v_i$), describing an undirected tree edge between the u_i-th vertex and the v_i-th vertex. It is guaranteed that the edges form a tree.

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

Output

For each test case, output a single line containing an integer, denoting the maximum number of cakes that Prof.Chen can make.

Example

standard input	standard output

4
7 5
1 2 1 2 2 1 2
1 2
2 3
3 4
3 5
5 6
5 7
2 2
1 0
1 2
1 1
1
1 2
1
	2
	0
	1
	0

Problem I. Barkley II

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

Prof.Hui is the coach of Pigeland University Programming Team. There are n students in his team. All algorithms are numbered by Prof.Hui in ascending order of difficulty, from 1 to m. Which means that algorithm 1 is the easiest algorithm, while algorithm m is the hardest. The i-th student masters the a_i-th easiest algorithm.

Now Prof.Hui wants to choose a team satisfying the following conditions:

- The index of the students in the team forms an interval. Which means that there exists two integers l, r such that $1 \le l \le r \le n$ and student x is in the team if and only if $l \le x \le r$.
- The rating of the team is maximized. The more algorithms the team mastered, the stronger they are, but if they cannot solve a hard problem in one contest, they will feel more disappointed. So the rating of the team is the number of **different** algorithms that the students in the team mastered minus the index of the **easiest** algorithm that no one in the team mastered. If the students in the team masters all the algorithms, the index of the **easiest** algorithm that no student in the team mastered is considered to be $m + 1$. For example, if $m = 5$ and there are 6 students in the team, mastering algorithm $2, 5, 4, 4, 1, 1$ respectively, the rating of the team is $4 - 3 = 1$.

Please help Prof.Hui to find the maximum rating of a team.

Input

The first line contains an integer t ($1 \le t \le 5 \cdot 10^5$), denoting the number of test cases.

For each test case, the first line contains two integer n, m ($1 \le n, m \le 5 \cdot 10^5$), denoting the number of students and the number of algorithms.

The second line contains n integers, the i-th integer a_i ($1 \le a_i \le m$) denoting the number of algorithm the i-th student masters.

It is guaranteed that the sum of n over all testcases does not exceed $5 \cdot 10^5$. Please notice that there is **no limit** on sum of m.

Output

For each test case, output one integer in one line, denoting the answer.

Example

standard input	standard output

2
5 4
1 2 2 3 4
5 10000
5 2 3 4 1
	2
	3

Problem J. The Phantom Menace

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	1024 megabytes

Putata has brought his newest string problem to this contest. You are given two string sequences A, B, each of the sequences contains exactly n strings, and all strings have a length of m. You are asked to reorder the strings so that concatenation of the strings in the two sequences are cyclic isomorphic after concatenation.

Formally, you should choose two permutations p, q of $1, 2, \ldots, n$, so that $A_{p_1} + A_{p_2} + \cdots + A_{p_n}$ and $B_{q_1} + B_{q_2} + \cdots + B_{q_n}$ are cyclic isomorphic. String $R = S + T$ satisfy that for $i \le |S|, R_i = S_i$, otherwise $R_i = T_{i-|S|}$. Two strings S, T are said to be cyclic isomorphic if and only if there exists an integer d, such that $S_i = T_{((i+d) \mod |S|)+1}$ for all $1 \le i \le |S|$, and $|S| = |T|$.

Please help Budada to find p and q, or report that there is no such p, q.

Input

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

For each test case, the first line contains two integers n, m ($1 \le n, m \le 10^6, 1 \le n \cdot m \le 10^6$).

Then n line follows, the i-th of which contains one string A_i ($|A_i| = m$).

Then n line follows, the i-th of which contains one string B_i ($|B_i| = m$).

It is guaranteed that all input strings only contain lowercase English letters.

It is also guaranteed that the sum of $n \cdot m$ over all test cases does not exceed 10^6.

Output

For each test case, if permutation p and q exists, output them in two lines, and the elements in one permutation are seperated by spaces. Otherwise output -1 in one line.

Example

standard input	standard output

2
3 3
abc
ghi
def
bcd
efg
hia
1 3
abc
def
	1 3 2
	1 2 3
	-1

Problem K. Randias Permutation Task

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

For two permutations A and B of length n, Randias can generate a permutation C of length n as $C = A \circ B$ in this way: for each $1 \le i \le n$, $C[i] = A[B[i]]$.

Now he is given m permutations $A_1, A_2, \ldots, A_m$, each of them is of length n. He wants to choose a non-empty set of indices $i_1, i_2, \ldots, i_k$ ($1 \le k \le m$,$1 \le i_1 < i_2 \cdots < i_k \le m$), and calculate $C = (((A_{i_1} \circ A_{i_2}) \circ A_{i_3}) \circ A_{i_4}) \cdots \circ A_{i_k}$. Randias wants to know, how many possible permutations C he can generate? Output the answer modulo $10^9 + 7$.

A permutation of length n is an array consisting of n distinct integers from 1 to n in arbitrary order. For example, $[2, 3, 1, 5, 4]$ is a permutation, but $[1, 2, 2]$ is not a permutation (2 appears twice in the array), and $[1, 3, 4]$ is also not a permutation ($n = 3$ but there is 4 in the array)

Input

The first line contains two positive integers n, m ($1 \le n \cdot m \le 180$), denoting the length of the permutation and the number of permutations.

The following m lines, each line contains n distinct integers, denoting one permutation.

Output

One single integer, denoting the number of possible permutations C Randias can generate, modulo 10^9+7.

Examples

standard input	standard output

`5 4`
`1 2 3 4 5`
`5 1 3 4 2`
`3 4 1 5 2`
`5 2 4 1 3`
	`8`

`2 1` `2 1`	`1`

Problem L. Alea Iacta Est

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

SSerxhs has two dice with n and m sides respectively, and the numbers on their faces are $1, 2, 3, \ldots, n$ and $1, 2, 3, \ldots, m$. He is interested in the probability distribution of the sum of numbers rolled on both dice. Help him find another pair of dice, different from the original one, such that their sum follows this distribution.

Since it is difficult to create dice with a large number of faces, you should minimize the sum of the number of faces on the dice.

It is assumed that the probability of each face being rolled on any die is equal.

Two pairs of dice are considered different if and only if one die from one pair is different from any die in the other pair.

Two dice are considered different if and only if there exists a number x such that the number of occurrences of x on the faces of the two dice is different.

Two random variables X and Y follow the same probability distribution if and only if $\forall k, P(X = k) = P(Y = k)$.

Input

Each test contains multiple test cases. The first line contains a single interger t ($1 \le t \le 4000$), denoting the number of test cases.

For each test case, the only line contains two integers n and m ($1 \le n, m \le 10^6$), denoting the number of faces of the two dice.

It is guaranteed that the sum of $\max\{n, m\}$ over all testcases does not exceed 10^6.

Output

For each test case, print two lines containing several integers describing the two dice.

For the i-th line, the first integer f_i denotes the number of face of the i-th die. The remaining f_i integers on the line $a_1, a_2, \ldots, a_{f_i}$ ($1 \le a_j < n + m$) denotes the numbers on each face of the corresponding die. It is possible for the numbers on the faces to be repeated.

If there are multiple solutions, you can print any of them.

Especially, if there is no solution, print “0” on both lines instead.

It is guaranteed that the sum of f_i over all test cases does not exceed $3 \cdot 10^6$.

Additional blank lines have been added in the sample to separate different test cases. You are free to choose whether to print these blank lines in your output.

Example

standard input	standard output

3
2 8
1 9
2 9
	4 1 2 3 4
	4 1 2 5 6
	
	3 1 2 3
	3 1 4 7
	
	6 1 2 4 5 7 8
	3 1 2 3

Problem M. Flipping Cards

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2.5 seconds
Memory limit:	1024 megabytes

n cards are placed in a row, where n is an odd number. Each card has numbers written on both sides. On the i-th card, a_i is facing up and b_i is facing down.

Grammy wants to maximize the median of all the numbers that are facing up. In order to achieve this, she can do the following operation **at most once**.

- Choose an interval $[l, r]$ and flip all the cards in the interval. After flipping the cards, b_i will be facing up and a_i will be facing down for $i \in [l, r]$.

Please help Grammy to calculate the median of all the numbers that are facing up under her optimal strategy.

Recall that the median of a sequence of numbers is the $\frac{n+1}{2}$-th largest number in the sequence.

Input

The first line contains two integers n $(1 \le n < 3 \cdot 10^5, n \bmod 2 = 1)$, denoting the number of cards.

Each of the next n lines contains 2 integers a_i, b_i $(1 \le a_i, b_i \le 10^9)$, denoting the initial number that is facing up and the initial number that is facing down for each card.

Output

Output one integer, denoting the median of all the numbers that are facing up under Grammy's optimal strategy.

Examples

standard input	standard output

5
3 6
5 2
4 7
6 4
2 8
	6

1 2 1	2