Problem A. World Fragments I

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

This problem is different from World Fragments II and World Fragments III. Please read the statements carefully.

You are given two non-negative **binary integers** x and y without leading zeros. You can apply the following operation to x any number of times (possibly zero):

1. Choose a digit b in x.
2. Let either $x \leftarrow x + b$ or $x \leftarrow x - b$.

For example, when $x = (110100)_2$ you can choose the digit $b = (1)_2$ in $(1\underline{1}0100)_2$, then let $x \leftarrow x - b = (110100)_2 - (1)_2 = (110011)_2$.

Find out the least number of operations needed to turn x into y. Print -1 if it is impossible to turn x into y.

Input

The first line contains a non-negative binary integer x. It is guaranteed that x has at most 60 digits.

The second line contains a non-negative binary integer y. It is guaranteed that y has at most 60 digits.

Output

Print a **decimal integer** in a single line, denoting the answer. If it is impossible to turn x into y, print -1.

Examples

standard input	standard output
1 0	1
10101010101010101010 11001100110011001100	139810

Note

In the first sample, the optimal way to apply the operation is shown as following:

- Choose the digit $b = (1)_2$, then let $x \leftarrow x - b = (0)_2$.

Problem B. Auspiciousness

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

Dog Card is a card game. In the game, there are a total of $2n$ cards in the deck, each card has a value, and the values of these $2n$ cards form a permutation of $1 \sim 2n$. There is a skill that works as follows:

1. Draw a card from the top of the deck.
2. If the deck is empty, then skip to step 3, otherwise you guess whether the card on the top of the deck has a higher value than your last drawn card and draw a card from the top of the deck. If your guess is correct, then repeat this step, otherwise skip to step 3.
3. End this process.

Nana enjoys playing this game, although she may not be skilled at it. Therefore, her guessing strategy when using this skill is simple: if the value of the last drawn card is less than or equal to n, then she guesses that the next card's value is higher; otherwise, she guesses that the next card's value is lower. She wants to know, for all different decks of cards (Obviously, there are $(2n)!$ cases), how many cards she can draw in total if she uses the skill only once in each case. Since this number can be very large, please provide the answer modulo a given value.

Input

The first line contains a single integer t ($1 \le t \le 300$) — the number of test cases.

Each test case consists of one line containing two integers n ($1 \le n \le 300$) and m ($2 \le m \le 10^9$), denoting half the total number of cards in the deck and the modulus respectively.

It is guaranteed that the sum of n over all test cases does not exceed 300.

Output

For each test case, print one integer — the total number of cards Nana can draw modulo m.

Example

standard input	standard output
3 1 1000000 2 1000000 3 1000000	4 84 3084

Note

For example, if $n = 2$ and the values of cards from the top to the bottom are $1, 2, 4, 3$, Nana can draw all cards by using the skill only once. The process is as follows:

- Draw the card of value 1.
- For $1 \le n$, she guesses that the next card has a higher value and draws the card of value 2.
- Her guess is correct. For $2 \le n$, she guesses that the next card has a higher value and draws the card of value 4.

- Her guess is correct. For $4 > n$, she guesses that the next card has a lower value and draws the card of value 3.

Note that if the values of cards from the top to the bottom are $1, 2, 3, 4$, although her last guess is wrong, she can draw all cards too according to the skill.

Problem C. Stillwater Prison

Input file:	`standard input`
Output file:	`standard output`
Time limit:	6 seconds
Memory limit:	1024 megabytes

In the problem H of BCPC 2020 Final, Mocha had used most of her magic to protect Arad mainland. After that, Mocha tried to go back to her own world by travel magic. However, Mocha spent so much magic to protect Arad mainland that she can't maintain mana stability during travel. As a result, Mocha found she woubld be transported into the Stillwater Prison!

The magic is forbidden in the Stillwater Prison, so Mocha wants to escape the Stillwater Prison as soon as possible. The Stillwater Prison can be regard as a convex polygon in an infinite two-dimensional plane. Mocha will be transported to some point P which is **strictly** inside the polygon. Mocha will choose one point Q which is on the edge of the polygon (Q can also be some vertice of the polygon), then she will move from point P to point Q along the segment PQ.

Mocha is too anxious to calm down and calculate. Please help her to calculate the minimum distance she needs to move to escape the Stillwater Prison. Since Mocha can't confirm where she will be transported, you need to answer multiple queries.

Input

The first line contains one integer n ($3 \le n \le 10^5$), indicating the number of vertices of the convex polygon.

In the next n lines, each line contains two integers x, y ($-10^9 \le x, y \le 10^9$), indicating the coordinates of the vertices of the convex polygon. The coordinates of the vertices are given in counterclockwise order.

The next line contains one integer q ($1 \le q \le 10^5$), indicating the number of queries.

In the next q lines, each line contains two integers P_x, P_y ($-10^9 \le P_x, P_y \le 10^9$), indicating the coordinate of the point P where Mocha will be transported to. It's guaranteed that P is **strictly** inside the polygon.

Output

For each query, print a real number in a single line, indicating the minimum distance she needs to move to escape the Stillwater Prison. Your answer will be considered correct if its relative or absolute error does not exceed 10^{-6}.

Examples

standard input	standard output
3 -10 -10 1000 2 0 11515 3 1 1 250 500 700 600	10.8685398419 259.5573860250 247.1282543552
4 0 0 4 0 4 4 0 4 5 2 2 2 1 2 3 1 2 3 2	2.0000000000 1.0000000000 1.0000000000 1.0000000000 1.0000000000
4 0 0 7 0 5 5 -1 4 3 2 1 4 3 1 3	1.0000000000 1.6712580436 1.3151918984

Problem D. Ama no Jaku

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

Let me tell you the things that I have been thinking for a long time.

You are given an $n \times n$ matrix A with each number in A either 0 or 1.

You can do the following operation on A any number of times (possibly zero):

- Choose a row or a column from A, and flip it (i.e. all the 0s in that row or column become 1s, and all the 1s become 0s).

Define r_i as $(\overline{A_{i1}A_{i2}\dots A_{in}})_2$, which means writing all the number in the i-th row from left to right, forming a binary number r_i.

Define c_j as $(\overline{A_{1j}A_{2j}\dots A_{nj}})_2$, which means writing all the number in the j-th column from top to bottom, forming a binary number c_j.

Note that r_i and c_j may change after operations.

Determine whether you can achieve $\min(r_i) \geq \max(c_j)$ by operations. If so, find out the minimal number of operations needed.

Input

The first line contains a positive integer n ($1 \leq n \leq 2000$), denoting the size of matrix A.

Each of the following n lines contains a string consisting of n `0` and `1`s, denoting each row of matrix A.

Output

If it is possible to achieve $\min(r_i) \geq \max(c_j)$, print the minimal number of operations in a single line. Otherwise, print -1.

Examples

standard input	standard output
3 100 010 001	-1
5 00100 00100 11011 00100 00100	2

Problem E. Koraidon, Miraidon and DFS Shortest Path

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

One day, Koraidon gave Miraidon a **directed** graph with n vertices and m edges. The weights of all edges are 1. Koraidon asked Miraidon to calculate the shortest path from vertex 1 to each vertex. It is guaranteed that vertex 1 can reach all other vertices through the edges.

However, Miraidon just forgot the BFS algorithm. He tried to use the DFS algorithm as a replacement. The algorithm looks like:

Algorithm 1 DFS Shortest Path

```
1: function DFS(u)
2:     visit[u] ← true
3:     edgeSet ← edges start from u
4:     RandomShuffle(edgeSet)                ▷ Arbitarily shuffle the edge set
5:     for each edge ∈ edgeSet do
6:         v ← the end vertex of edge
7:         if visit[v] then
8:             continue
9:         else
10:            dis[v] ← dis[u] + 1
11:            DFS(v)
12:        end if
13:    end for
14: end function
15: dis[1] ← 0
16: DFS(1)
```

In line 4, Miraidon tried to use a random shuffle function to get a correct answer. The shuffles in different recursive calls of DFS are mutually independent.

Koraidon judged the answer given by Miraidon very strictly. He would consider the algorithm correct if and only if the algorithm gives correct answers on all possible random shuffle results.

Please help Miraidon check whether his DFS algorithm is correct on the given graph.

Input

Each test contains multiple test cases.

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

For each test case:

The first line contains two integers n, m ($1 \le n, m \le 5 \times 10^5$).

Each of the next m lines contains two integers u and v ($1 \le u, v \le n$), denoting a directed edge from u to v. The graph is **not** guaranteed to be a simple directed graph. But it is guaranteed that vertex 1 can reach all other vertices through the edges.

It is guaranteed that the sum of n and the sum of m over all test cases do not exceed 5×10^5.

Output

For each test case, print "`Yes`" if the algorithm is correct on this graph, and "`No`" otherwise.

You can output the answer in any case (upper or lower). For example, the strings "`yEs`", "`yes`", "`Yes`",

and "YES" will be recognized as positive responses.

Example

standard input	standard output
2 3 3 1 2 2 3 3 1 3 3 1 2 1 3 2 3	Yes No

Problem F. World Fragments II

Input file:	`standard input`
Output file:	`standard output`
Time limit:	9 seconds
Memory limit:	1024 megabytes

This problem is different from World Fragments I and World Fragments III. Please read the statements carefully.

There are q queries for you to answer. Only when you have answered the previous queries can you get the next query.

For each query, you are given two non-negative **decimal integers** x and y without leading zeros. You can apply the following operation to x any number of times (possibly zero):

1. Choose a digit b in x.
2. Let either $x \leftarrow x + b$ or $x \leftarrow x - b$.

For example, when $x = (616)_{10}$ you can choose the digit $b = (1)_{10}$ in $(6\underline{1}6)_{10}$, then let $x \leftarrow x - b = (616)_{10} - (1)_{10} = (615)_{10}$.

Find out the least number of operations needed to turn x into y. Print -1 if it is impossible to turn x into y.

Input

The first line contains a positive integer T ($1 \le T \le 3 \times 10^5$) — the number of queries.

Each query consists of a line that contains two integers x', y'. Let the answer of the previous query be *lastans*. Specifically, *lastans* is considered to be 0 for the first query. The decimal integers x, y in the given query are shown as below, where $\oplus$ denotes the bitwise exclusive OR operation.

$$x = x' \oplus (lastans + 1),\ y = y' \oplus (lastans + 1)$$

It is guaranteed that $0 \le x, y \le 3 \times 10^5$ holds for each query.

Output

For each query, print a decimal integer in a single line, denoting the answer. If it is impossible to turn x into y, print -1.

Example

standard input	standard output
6 0 8 17 16 16 17 1 5 12 0 4 7	6 6 2 -1 3 -1

Note

The actual queries and the operations in their optimal ways are shown as followings:

1. $x = 1,\ y = 9$, optimal 6 operations: $\underline{1} \to \underline{2} \to \underline{4} \to \underline{8} \to 1\underline{6} \to \underline{1}0 \to 9$.

2. $x = 22,\ y = 23$, optimal 6 operations: $\underline{2}2 \to 2\underline{4} \to \underline{2}8 \to \underline{3}0 \to \underline{2}7 \to \underline{2}5 \to 23$.

3. $x = 23,\ y = 22$, optimal 2 operations: $2\underline{3} \to \underline{2}0 \to 22$.

4. $x = 2,\ y = 6$, impossible.

5. $x = 12,\ y = 0$, optimal 3 operations: $1\underline{2} \to \underline{1}0 \to \underline{9} \to 0$.

6. $x = 0,\ y = 3$, impossible.

Problem G. Beautiful Matrix

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	1024 megabytes

You are given a character matrix s of size $n \times m$.

Let's call a $n \times m$ matrix s **beautiful** if and only if $n = m$ and $\forall i, j \in [1, n], s_{i,j} = s_{n-i+1,n-j+1}$.

Let's call a matrix's **beauty** the number of **beautiful** submatrices in it.

Your task is to calculate the **beauty** of the given character matrix.

A submatrix of a matrix is formed by selecting some **contiguous** rows and columns in the matrix and forming a new matrix by using those entries in the same relative positions.

Please pay attention to the the time complexity of the program.

Input

The first line contains two integers n and m ($1 \leq n, m \leq 2000$) – the size of the given matrix.

Then there are n lines, each line contains a string of lowercase English characters with length m – the matrix.

Output

Output a single integer – the beauty of the given matrix.

Example

standard input	standard output
3 4 abcz dedz cbaz	13

Problem H. Until the Blue Moon Rises

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

You are given an array $A = (A_1, A_2, \ldots, A_n)$ of n positive integers.

You can do the following operation any number of times (possibly, zero) on A:

- Choose two **distinct** indices i, j $(1 \le i, j \le n)$, then let $A_i = A_i + 1, A_j = A_j - 1$.

Determine whether you can make all the integers in the array A prime numbers by applying the operations.

Recall that a prime number is a positive integer greater than 1 that is not a product of two smaller positive integers.

Input

The first line contains an integer n $(1 \le n \le 1000)$ — the length of array A.

The second line contains n positive integers $A_1, A_2, \ldots, A_n$ $(1 \le A_i \le 10^9)$ — the integers in A.

Output

If it is possible to make all the integers in the array A prime numbers, print '`Yes`' in a single line (without quotes). Otherwise, print '`No`' (without quotes).

Example

standard input	standard output
5 10 8 2 3 5	Yes

Problem I. To the Colors of the Dreams of Electric Sheep

Input file:	`standard input`
Output file:	`standard output`
Time limit:	7 seconds
Memory limit:	512 megabytes

You are visiting a colorful tree in the dream of Electric Sheep, which has at most 60 colors. These colors are numbered from 0 to 59.

The colorful tree has n vertices, connected with $n-1$ edges. Each vertex is painted in some of the colors. Let non-negative integer c_i denotes the colors of vertex i. If the lowest i-th bit of c_i is 1, the colors that vertex i is painted contains color i. Note that a vertex can be painted in no colors, which means this vertex is transparent.

You also have your own color c. When you are travelling on the colorful tree, you can visit vertex i iff the colors of vertex i contains your color c. If you are at vertex u and your color is c, you can do one of the following operations in 1 second:

- Move to vertex v that is adjacent to vertex u.

  Note that both the colors of vertex u and those of vertex v should contain color c.

- Choose a color c' and change your color to c'.

  Note that the colors of vertex u should contain both color c and c'.

Now you are planning for q trips. The i-th trip is planned from vertex u_i to another vertex v_i. You can choose any color you want as your color c at vertex u_i before the trip. Find out the number of seconds that need to be taken to arrive at vertex v_i in the optimal way for each trip.

If you can't reach vertex v_i from vertex u_i, print -1.

Input

The first line contains two positive integers n, q ($2 \le n \le 5 \times 10^5$, $1 \le q \le 5 \times 10^5$) — the number of vertices, and the number of trips.

The second line contains n non-negative integers $c_1, c_2, \ldots, c_n$ ($0 \le c_i < 2^{60}$) — the colors of each vertex.

Each line in the following $n-1$ lines contains two positive integers u, v ($1 \le u, v \le n$) — the edge connecting vertex u and vertex v.

Each line in the following q lines contains two positive integers u_i, v_i ($1 \le u_i, v_i \le n$, $u_i \ne v_i$) — the i-th trip from vertex u_i to vertex v_i.

Output

For each trip, print the number of seconds that need to be taken in the optimal way in a single line. If vertex v_i is not reachable from vertex u_i in the trip, print -1.

Examples

standard input	standard output
10 8 12 9 14 5 1 18 18 9 14 7 1 3 2 3 2 4 3 6 4 10 5 8 7 9 8 9 8 10 1 7 3 8 4 5 5 6 6 1 8 10 9 2 10 5	10 5 3 8 3 1 5 2
3 2 0 1 2 1 2 2 3 1 3 3 2	-1 -1

Note

In the first sample, the optimal way for the 3-rd trip from vertex 4 to vertex 5 is shown as followings:

- Choose color 0 as your own color before the trip at vertex 4.
- Move to vertex 10 from vertex 4. It takes 1 second.
- Move to vertex 8 from vertex 10. It takes 1 second.
- Move to vertex 5 from vertex 8. It takes 1 second, and 3 seconds in total.

The optimal way for the 5-th trip from vertex 6 to vertex 1 is shown as followings:

- Choose color 1 as your own color before the trip at vertex 6.
- Move to vertex 3 from vertex 6. It takes 1 second.
- Choose color 3 and change your own color to it. It takes 1 second.
- Move to vertex 1 from vertex 3. It takes 1 second, and 3 seconds in total.

In the second sample, it is not allowed to visit vertex 1, so you can't start the trip at vertex 1. It is impossible to travel from vertex 2 to vertex 3, because the colors of vertex 2 and those of vertex 3 share no colors.

Problem J. Fine Logic

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

We consider a set G consisting of n entities and the relations on G. For simplicity, these n entities are labeled from 1 to n.

The **ranking** R on G is defined as an n-order permutation $R = r_1, r_2, \dots r_n$ $(1 \le r_i \le n)$.

The **winning relation** on G is defined as an ordered pair $\langle u, v \rangle$ $(1 \le u, v \le n,\ u \ne v)$, representing that the entity u **wins** in the competition with the entity v.

We say that the ranking R **satisfies** the winning relation $\langle u, v \rangle$ if and only if there exist $1 \le i < j \le n$ such that $r_i = u$ and $r_j = v$.

Given the number of entities n, and m winning relations, you are required to construct the minimal number of rankings such that each winning relation is satisfied by at least one ranking in the constructed rankings.

Input

The first line contains two positive integers n, m $(2 \le n \le 10^6,\ 1 \le m \le 10^6)$, representing the number of entities, and the number of winning relations.

Each of the following m lines contains two positive integers u, v $(1 \le u, v \le n,\ u \ne v)$, representing a winning relation $\langle u, v \rangle$.

Output

The first line contains a positive integer k, representing the minimal number of rankings needed.

Each of the following k lines contains n integers $r_{i1}, r_{i2}, \dots r_{in}$ $(1 \le r_{ij} \le n)$, representing the i-th ranking $R_i = r_{i1}, r_{i2}, \dots r_{in}$.

Example

standard input	standard output
6 4 1 4 2 3 5 6 5 3	1 2 5 1 6 3 4

Problem K. World Fragments III

Input file:	standard input
Output file:	standard output
Time limit:	5 seconds
Memory limit:	512 megabytes

This problem is different from World Fragments I and World Fragments II. Please read the statements carefully.

There are q queries for you to answer.

For each query, you are given two non-negative **decimal integers** x and y without leading zeros. You can apply the following operation to x any number of times (possibly zero):

1. Choose a digit b in x.
2. Let either $x \leftarrow x + b$ or $x \leftarrow x - b$.

For example, when $x = (616)_{10}$ you can choose the digit $b = (1)_{10}$ in $(6\underline{1}6)_{10}$, then let $x \leftarrow x - b = (616)_{10} - (1)_{10} = (615)_{10}$.

Find out the least number of operations needed to turn x into y. Since the answer can be very large, output it modulo 998 244 353. Print -1 if it is impossible to turn x into y.

Input

The first line contains a positive integer T ($1 \le T \le 10^3$) — the number of queries.

Each query consists of a line that contains two integers x, y ($0 \le x \le y \le 10^{10^6}$). Please note that in this version, $x \le y$.

It is guaranteed that the sum of the lengths of the decimal representations of all ys does not exceed 2×10^6.

Output

For each query, print a decimal integer in a single line, denoting the answer modulo 998 244 353. If it is impossible to turn x into y, print -1.

Example

standard input	standard output
4 1 7 1 9 13 23 123456789 987654321	-1 6 8 102755278

Note

The operations in the third example are:

$1\underline{3} \to 1\underline{6} \to 2\underline{2} \to 2\underline{4} \to 2\underline{8} \to \underline{3}0 \to \underline{2}7 \to \underline{2}5 \to 23$.