A. Morning Routine

Limits: 1 sec., 256 MiB

Every morning (around 6:00 AM) Zenyk is solving the following problem.

Zenyk is given a string consisting of upper case characters `A` and `B`. In a single turn he can remove the first and the last occurrences of any character, but only if they don't coincide. What is the lexicographically smallest non-empty string that he can obtain after any number of turns?

String s is considered lexicographically smaller than t if s is a prefix of t, or s has a smaller character at the first position they differ (from left to right).

Input

The only line contains the initial string s that was given to Zenyk.

Output

Print the answer to the problem.

Constraints

$1 \leq |s| \leq 10^5$,
s consists only of characters `A` and `B`.

Samples

Input (*stdin*)	Output (*stdout*)
`BBABBAB`	`ABA`

B. Towers

Limits: 2 sec., 256 MiB

Zenyk and Marichka like tower sequences. Tower sequence is a sequence of n towers in a row.

Let's assume that the height of the i-th tower is A_i. Let's say that tower j is visible from tower i if tower j is strictly higher than all towers between tower i and tower j (not including the i-th tower). More formally, let S be the range of all towers between i-th and j-th tower. This means that $S = [i+1, j-1]$ if $j > i$, and $S = [j+1, i-1]$ otherwise. The j-th tower is visible from the tower i if $\forall_{k \in S} A_j > A_k$.

Let B_i be the number of towers visible from tower i (not including tower i). Marichka calls a sequence of towers lucky if $A_i = B_i$ for all i. You task is to find the number of lucky sequences of n towers modulo prime number m.

Input

The first line contains 2 integers n and m.

Output

Print one number — the number of lucky sequences of n towers modulo m.

Constraints

$2 \le n \le 1000$,
$10^7 \le m \le 10^9$, m is prime.

Samples

Input (*stdin*)	Output (*stdout*)
`7 47774477`	`3`

Notes

Lucky sequences are $[1, 2, 2, 2, 2, 2, 1]$, $[2, 2, 3, 2, 3, 2, 2]$, $[2, 3, 2, 4, 2, 3, 2]$.

C. Zero Cycle

Limits: 2 sec., 256 MiB

Zenyk and Marichka have a graph with n vertices and m edges. i-th edge is directed from vertex v_i to vertex u_i, $v_i \neq u_i$. Let $(v_i \to u_i)$ denote such edge.

Marichka wants to assign a weight to each edge such that all weigths are non-zero integers in range $[-1000, 1000]$. Also Marichka wants the sum of edges on any cycle to be 0. Cycle is a sequence of edges $(a_1 \to a_2), (a_2 \to a_3), \ldots, (a_{k-1} \to a_k), (a_k \to a_1)$. If edges $(x \to y)$ and $(y \to x)$ both exist they also form a cycle. Help her with that task.

Input

First line contains two integers n and m — number of vertices and number of edges, respectively. Next m lines contain two integers each v_i and u_i which means that i-th edge goes from v_i to u_i.

Output

Print m numbers: i-th number should be the weight of i-th edge. Each number should be non-zero and in range $[-1000, 1000]$. If multiple answers exist you may print any one of them.

Constraints

$1 \leq n \leq 10^5$,
$1 \leq m \leq 2 \cdot 10^5$,
$1 \leq v_i, u_i \leq n$, $v_i \neq u_i$.

Samples

Input (*stdin*)	Output (*stdout*)
4 7 1 2 2 3 3 1 1 4 2 4 1 4 3 2	4 -7 3 47 -774 -447 7

D. Random GCD

Limits: 1 sec., 256 MiB

Zenyk has an array A of n integers. Marichka wants to find the greatest common divisor of this array.

To find it Zenyk wrote the following `C++` program:

```
random_shuffle(A.begin(), A.end());
int count = 0;
int result = A[0];
for(int i = 1; i < n; i++) {
    if (result == 1)
        break;
    result = gcd(result, A[i]);
    count ++;
}
```

Here we assume that after `random_shuffle` each permutation of array A is equiprobable and `gcd(a,b)` is the greatest common divisor of a and b. Zenyk is interested what is the expected value of `count` at the end of such a process.

Input

First line contains single integer n. Second line contains n integers A_i — elements of array.

Output

Print one value — the expected value of `count`. Answer is considered to be correct if its absolute or relative error is less than 10^{-7}.

Constraints

$1 \le n \le 10^6$,
$1 \le A_i \le 10^6$.

Samples

Input (*stdin*)	Output (*stdout*)
4 4 7 4 14	1.91666666666667

E. Longest Vector

Limits: 2 sec., 256 MiB

Zenyk has n 2-dimensional vectors but Marichka thinks that these vectors are not long enough. Zenyk wants to find such a subset of his vectors so that their sum is the longest possible.

Input

First line contains a single integer n. Next n lines contain 2 integers each x_i and y_i — coordinates of the i-th vector.

Output

Print one integer — squared length of the longest possible vector Zenyk can create.

Constraints

$1 \le n \le 2 \cdot 10^5$,
$-10^9 \le x_i, y_i \le 10^9$.

Samples

Input (*stdin*)	Output (*stdout*)
4 1 0 0 1 1 1 -1 -1	8
7 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000	98000000000000000000

F. Consecutive Triangles

Limits: 1 sec., 256 MiB

Marichka and Zenyk have n sticks. The length of the i-th stick is i (1-indexing).

Marichka wants to place all the sticks in a row so that it isn't possible to create a non-degenerate triangle using any triplet of consecutive sticks. Help her with that task.

Input

A single integer n.

Output

Print n integers — a permutation of sticks such that it's not possible to create a triangle using any consecutive triplet of sticks. If several answers exist print any of them.

Constraints

$3 \le n \le 10^5$.

Samples

Input (*stdin*)	Output (*stdout*)
`4`	`4 1 2 3`

G. Can You Do This?

Limits: 1 sec., 256 MiB

Given two integers a and b ($a < b$) find the smallest positive integer c such that a `OR` $c > b$ `OR` c.

Input

The only line contains two integers a and b.

Output

Print a single integer c — the answer. If no such integer exists, print `-1`.

Constraints

$0 < a < b < 10^{18}$.

Samples

Input (*stdin*)	Output (*stdout*)
`47 74`	`64`

Notes

`OR` stands for bitwise OR operation.

H. Moving Balls

Limits: 1 sec., 256 MiB

Zenyk has n balls placed on a line, the i-th ball is initially located in the p_i-th cell. No two balls can ever occupy a single cell. A robot $x \to y$ can go from cell x to cell y and push all the balls on its way. If one ball stands on the way of another ball, then it's also pushed. E. g. if you apply robot $1 \to 4$ to `1011000001` (1-based index) you'll get `0000111001`. (Here `1` indicates a cell occupied by a ball, while `0` – an empty one.) Note that some robots may go from right to left.

The goal is to have all the balls placed in a row next to each other, i. e. without empty spaces between them. If you have a sequence of robots $x_1 \to y_1, ..., x_k \to y_k$, you can use each robot as many times as you like and in any order to achieve the goal.

You have a sequence of pairs $(x_1, y_1), ..., (x_m, y_m)$. You have to assign a direction to each of them, and you'll get either robot $x_i \to y_i$ or $y_i \to x_i$ for each pair. Find the number of ways to do it such that the goal is achievable.

Input

The first line containts two integers n and m. The next line contains n distinct intigers p_i — the initial locations of all the balls. The following m lines contain pairs (x_i, y_i).

Output

The number of ways to assign directions modulo $10^9 + 7$.

Constraints

$1 \le n, m \le 2000$,
$1 \le p_i, x_i, y_i \le 10^6$,
$x_i \le y_j$.

Samples

Input (*stdin*)	Output (*stdout*)

`3 3`
`1 4 6`
`3 5`
`1 4`
`6 7`
	`5`

I. Shoot for the Stars

Limits: 1 sec., 256 MiB

Given a connected graph with n vertices and $n-1$ edges you have to delete at most $\frac{n}{2}$ (rounded down) edges such that each connected component of the resulting graph is a star.

A star is such a graph that contains at most one vertex that has more than one edge connected to it.

Input

The first line contains a single integer n.

The next $n-1$ lines contain a pair of integer x_i, y_i each which means that vertices x_i and y_i are connected with an edge.

Output

On the first line of the output print one integer k — the number of edges to be removed from the graph.

On the second line print n integers — indices of the edges (1-based) to be removed from the graph.

If there are multiple possible answers, print any one of them. If there is no answers at all print `-1`.

Constraints

$1 \le n \le 10^5$,
$1 \le x_i, y_i \le n$,
$0 \le k \le \frac{n}{2}$ (rounded down).

Samples

Input (*stdin*)	Output (*stdout*)
10 1 3 3 2 4 3 3 5 5 6 7 9 6 9 8 9 9 10	2 4 7

Notes

Note that you do not have to minimize the number of edges deleted. Any solution with at most $\frac{n}{2}$ edges deleted will be accepted.