Codeforces Round #323 (Div. 2)

A. Asphalting Roads

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

City X consists of n vertical and n horizontal infinite roads, forming $n \times n$ intersections. Roads (both vertical and horizontal) are numbered from 1 to n, and the intersections are indicated by the numbers of the roads that form them.

Sand roads have long been recognized out of date, so the decision was made to asphalt them. To do this, a team of workers was hired and a schedule of work was made, according to which the intersections should be asphalted.

Road repairs are planned for n^2 days. On the i-th day of the team arrives at the i-th intersection in the list and if **none** of the two roads that form the intersection were already asphalted they asphalt both roads. Otherwise, the team leaves the intersection, without doing anything with the roads.

According to the schedule of road works tell in which days at least one road will be asphalted.

Input

The first line contains integer n $(1 \le n \le 50)$ — the number of vertical and horizontal roads in the city.

Next n^2 lines contain the order of intersections in the schedule. The i-th of them contains two numbers h_i, v_i $(1 \le h_i, v_i \le n)$, separated by a space, and meaning that the intersection that goes i-th in the timetable is at the intersection of the h_i-th horizontal and v_i-th vertical roads. It is guaranteed that all the intersections in the timetable are distinct.

Output

In the single line print the numbers of the days when road works will be in progress in ascending order. The days are numbered starting from 1.

Sample test(s)

input
2 1 1 1 2 2 1 2 2
output
1 4

input
1 1 1
output
1

Note

In the sample the brigade acts like that:

1. On the first day the brigade comes to the intersection of the 1-st horizontal and the 1-st vertical road. As none of them has been asphalted, the workers asphalt the 1-st vertical and the 1-st horizontal road;
2. On the second day the brigade of the workers comes to the intersection of the 1-st horizontal and the 2-nd vertical road. The 2-nd vertical road hasn't been asphalted, but as the 1-st horizontal road has been asphalted on the first day, the workers leave and do not asphalt anything;
3. On the third day the brigade of the workers come to the intersection of the 2-nd horizontal and the 1-st vertical road. The 2-nd horizontal road hasn't been asphalted but as the 1-st vertical road has been asphalted on the first day, the workers leave and do not asphalt anything;
4. On the fourth day the brigade come to the intersection formed by the intersection of the 2-nd horizontal and 2-nd vertical road. As none of them has been asphalted, the workers asphalt the 2-nd vertical and the 2-nd horizontal road.

B. Robot's Task

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Robot Doc is located in the hall, with n computers stand in a line, numbered from left to right from 1 to n. Each computer contains **exactly one** piece of information, each of which Doc wants to get eventually. The computers are equipped with a security system, so to crack the i-th of them, the robot needs to collect at least a_i any pieces of information from the other computers. Doc can hack the computer only if he is right next to it.

The robot is assembled using modern technologies and can move along the line of computers in either of the two possible directions, but the change of direction requires a large amount of resources from Doc. Tell the minimum number of changes of direction, which the robot will have to make to collect all n parts of information if initially it is next to computer with number 1.

It is guaranteed that there exists at least one sequence of the robot's actions, which leads to the collection of all information. Initially Doc doesn't have any pieces of information.

Input

The first line contains number n ($1 \le n \le 1000$). The second line contains n non-negative integers $a_1, a_2, ..., a_n$ ($0 \le a_i < n$), separated by a space. It is guaranteed that there exists a way for robot to collect all pieces of the information.

Output

Print a single number — the minimum number of changes in direction that the robot will have to make in order to collect all n parts of information.

Sample test(s)

input
3 0 2 0
output
1

input
5 4 2 3 0 1
output
3

input
7 0 3 1 0 5 2 6
output
2

Note

In the first sample you can assemble all the pieces of information in the optimal manner by assembling first the piece of information in the first computer, then in the third one, then change direction and move to the second one, and then, having 2 pieces of information, collect the last piece.

In the second sample to collect all the pieces of information in the optimal manner, Doc can go to the fourth computer and get the piece of information, then go to the fifth computer with one piece and get another one, then go to the second computer in the same manner, then to the third one and finally, to the first one. Changes of direction will take place before moving from the fifth to the second computer, then from the second to the third computer, then from the third to the first computer.

In the third sample the optimal order of collecting parts from computers can look like that: 1->3->4->6->2->5->7.

C. GCD Table

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

The GCD table G of size $n \times n$ for an array of positive integers a of length n is defined by formula

$$g_{ij} = \gcd(a_i, a_j).$$

Let us remind you that the greatest common divisor (GCD) of two positive integers x and y is the greatest integer that is divisor of both x and y, it is denoted as $\gcd(x, y)$. For example, for array $a = \{4, 3, 6, 2\}$ of length 4 the GCD table will look as follows:

	1	2	3	4
1	4	1	2	2
2	1	3	3	1
3	2	3	6	2
4	2	1	2	2

Given all the numbers of the GCD table G, restore array a.

Input

The first line contains number n ($1 \le n \le 500$) — the length of array a. The second line contains n^2 space-separated numbers — the elements of the GCD table of G for array a.

All the numbers in the table are positive integers, not exceeding 10^9. Note that the elements are given in an arbitrary order. It is guaranteed that the set of the input data corresponds to some array a.

Output

In the single line print n positive integers — the elements of array a. If there are multiple possible solutions, you are allowed to print any of them.

Sample test(s)

input
4 2 1 2 3 4 3 2 6 1 1 2 2 1 2 3 2
output
4 3 6 2

input
1 42
output
42

input
2 1 1 1 1
output
1 1

D. Once Again...

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given an array of positive integers $a_1, a_2, ..., a_{n \times T}$ of length $n \times T$. We know that for any $i > n$ it is true that $a_i = a_{i - n}$. Find the length of the longest non-decreasing sequence of the given array.

Input

The first line contains two space-separated integers: n, T ($1 \le n \le 100$, $1 \le T \le 10^7$). The second line contains n space-separated integers $a_1, a_2, ..., a_n$ ($1 \le a_i \le 300$).

Output

Print a single number — the length of a sought sequence.

Sample test(s)

input
4 3 3 1 4 2
output
5

Note

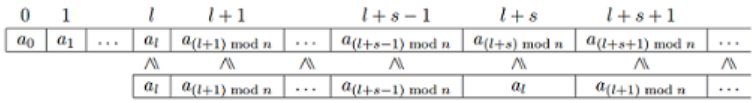
The array given in the sample looks like that: 3, **1**, 4, **2**, **3**, 1, **4**, 2, 3, 1, **4**, 2. The elements in bold form the largest non-decreasing subsequence.

E. Superior Periodic Subarrays

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given an infinite periodic array $a_0, a_1, ..., a_{n-1}, ...$ with the period of length n. Formally, $a_i = a_{i \bmod n}$. A periodic subarray (l, s) ($0 \le l < n$, $1 \le s < n$) of array a is an infinite periodic array with a period of length s that is a subsegment of array a, starting with position l.

A periodic subarray (l, s) is *superior*, if when attaching it to the array a, starting from index l, any element of the subarray is larger than or equal to the corresponding element of array a. An example of attaching is given on the figure (top — infinite array a, bottom — its periodic subarray (l, s)):



Find the number of distinct pairs (l, s), corresponding to the superior periodic arrays.

Input

The first line contains number n ($1 \le n \le 2 \cdot 10^5$). The second line contains n numbers $a_0, a_1, ..., a_{n-1}$ ($1 \le a_i \le 10^6$), separated by a space.

Output

Print a single integer — the sought number of pairs.

Sample test(s)

input
4 7 1 2 3
output
2

input
2 2 1
output
1

input
3 1 1 1
output
6

Note

In the first sample the superior subarrays are (0, 1) and (3, 2).

Subarray (0, 1) is superior, as $a_0 \ge a_0, a_0 \ge a_1, a_0 \ge a_2, a_0 \ge a_3, a_0 \ge a_0, ...$

Subarray (3, 2) is superior $a_3 \ge a_3, a_0 \ge a_0, a_3 \ge a_1, a_0 \ge a_2, a_3 \ge a_3, ...$

In the third sample any pair of (l, s) corresponds to a superior subarray as all the elements of an array are distinct.