Nemico mortale (`nemesi`)

Time limit:	0.5 seconds
Memory limit:	256 MiB
Difficulty	1

In a primary school teachers want to organize an art lab and they need to split the kids into groups. The situation here is that some kids steal colors from others. Each kid has another kid stealing her or his colors. It is not possible that two kids steal colors from the same kid but the same kid can steal colors from different kids. The situation is not symmetrical: if kid A steals colors from kid B, it is not necessarily true that kid B steals colors from kid A. As each group requires a teacher, your task is to help the teachers to split the kids into the minimal number of groups so that no color stealing occurs inside each group.

Implementation

You shall submit one file having extension `.c`, `.cpp` or `.pas`.

> ☞ Among the attachments of this task you will find a template (`nemesi.c`, `nemesi.cpp`, `nemesi.pas`) with a sample incomplete implementation.

You need to implement the following function:

C/C++	`void smista(int N, int nemico[]);`
Pascal	`procedure smista(N: longint; nemico: array of longint);`

- N is an integer representing the number of kids, who are numbered $0, 1, \ldots, N-1$.
- `nemico` is an array, indexed from 0 to $N-1$, such that `nemico`$[i] = j$ if and only if kid j steals colors from kid i.

You can use the following functions, already defined in the grader:

C/C++	`void nuovo_gruppo();`
Pascal	`procedure nuovo_gruppo();`

The above function creates a new group, which is initially empty. Kids can be added to the group through the function `aggiungi` (see below).

C/C++	`void aggiungi(int kid);`
Pascal	`procedure aggiungi(int kid);`

The above function adds kid number `kid` to the last created group.

The grader will call the function `smista`, and the latter must assign kids to groups using `nuovo_gruppo` and `aggiungi`.

Grader

In the directory for this problem there is a simplified version of the grader used during evaluation, which you can use to test your solutions locally. The sample grader reads data from `stdin`, calls the function that you should implement and writes to `stdout` in the following format.

The input file is made of 2 lines, containing:

- Line 1: integer N.
- Lines 2: values `nemico[`i`]` for $i = 0, \ldots, N-1$..

The output file is made of a G lines, where G represents the number of groups created by function `smista`. The ith line contains the kid numbers assigned to the ith group, listed in arbitrary order inside that group.

Constraints

- $2 \leq N \leq 100\,000$.
- Eack kid is assigned to one group.
- No kid steals colors to herself or himself.

Scoring

Your program will be tested against several test cases grouped in subtasks. For each test case you will get the following score

$$2^{\texttt{opt}-\texttt{eff}}$$

where `opt` is the optimal number of groups and `eff` is the number of groups obtained with your solution. For each subtask, its score is given by the product of its points below times the above factor for the worst test case in the subtask.

- **Subtask 1 [5 points]**: Examples.
- **Subtask 2 [13 points]**: Stealing is always symmetrical: if kid A steals colors from kid B, so does kid B from kid A.
- **Subtask 3 [30 points]**: Each kid can steal colors from at most another kid.
- **Subtask 4 [25 points]**: $N \leq 2000$.
- **Subtask 5 [27 points]**: No limitations.

Examples

input.txt	output.txt
5 1 0 1 0 3	0 4 2 3 1
5 1 2 3 4 0	1 4 3 2 0

Explanation

In the **first example** it is possible to split the kids into two groups: the first group is $\{0, 2, 4\}$ and the second is $\{1, 3\}$. The following calls are executed to this end: `nuovo_gruppo()`, `aggiungi(0)`, `aggiungi(4)`, `aggiungi(2)`, `nuovo_gruppo()`, `aggiungi(3)`, `aggiungi(1)`.

In the **second example**, three groups are obtained: $\{1, 4\}$, $\{3\}$ e $\{0, 2\}$.