Rarest Insects

There are N insects, indexed from 0 to $N-1$, running around Pak Blangkon's house. Each insect has a **type**, which is an integer between 0 and 10^9 inclusive. Multiple insects may have the same type.

Suppose insects are grouped by type. We define the cardinality of the **most frequent** insect type as the number of insects in a group with the most number of insects. Similarly, the cardinality of the **rarest** insect type is the number of insects in a group with the least number of insects.

For example, suppose that there are 11 insects, whose types are $[5, 7, 9, 11, 11, 5, 0, 11, 9, 100, 9]$. In this case, the cardinality of the **most frequent** insect type is 3. The groups with the most number of insects are type 9 and type 11, each consisting of 3 insects. The cardinality of the **rarest** insect type is 1. The groups with the least number of insects are type 7, type 0, and type 100, each consisting of 1 insect.

Pak Blangkon does not know the type of any insect. He has a machine with a single button that can provide some information about the types of the insects. Initially, the machine is empty. To use the machine, three types of operations can be performed:

1. Move an insect to inside the machine.
2. Move an insect to outside the machine.
3. Press the button on the machine.

Each type of operation can be performed at most $40\ 000$ times.

Whenever the button is pressed, the machine reports the cardinality of the **most frequent** insect type, considering only insects inside the machine.

Your task is to determine the cardinality of the **rarest** insect type among all N insects in Pak Blangkon's house by using the machine. Additionally, in some subtasks, your score depends on the maximum number of operations of a given type that are performed (see Subtasks section for details).

Implementation Details

You should implement the following procedure:

```
int min_cardinality(int N)
```

- N: the number of insects.
- This procedure should return the cardinality of the **rarest** insect type among all N insects in Pak Blangkon's house.
- This procedure is called exactly once.

The above procedure can make calls to the following procedures:

```
void move_inside(int i)
```

- i: the index of the insect to be moved inside the machine. The value of i must be between 0 and $N - 1$ inclusive.
- If this insect is already inside the machine, the call has no effect on the set of insects in the machine. However, it is still counted as a separate call.
- This procedure can be called at most $40\ 000$ times.

```
void move_outside(int i)
```

- i: the index of the insect to be moved outside the machine. The value of i must be between 0 and $N - 1$ inclusive.
- If this insect is already outside the machine, the call has no effect on the set of insects in the machine. However, it is still counted as a separate call.
- This procedure can be called at most $40\ 000$ times.

```
int press_button()
```

- This procedure returns the cardinality of the **most frequent** insect type, considering only insects inside the machine.
- This procedure can be called at most $40\ 000$ times.
- The grader is **not adaptive**. That is, the types of all N insects are fixed before `min_cardinality` is called.

Example

Consider a scenario in which there are 6 insects of types $[5, 8, 9, 5, 9, 9]$ respectively. The procedure `min_cardinality` is called in the following way:

```
min_cardinality(6)
```

The procedure may call `move_inside`, `move_outside`, and `press_button` as follows.

Call	Return value	Insects in the machine	Types of insects in the machine
		$\{\}$	$[]$
`move_inside(0)`		$\{0\}$	$[5]$
`press_button()`	1	$\{0\}$	$[5]$
`move_inside(1)`		$\{0, 1\}$	$[5, 8]$
`press_button()`	1	$\{0, 1\}$	$[5, 8]$
`move_inside(3)`		$\{0, 1, 3\}$	$[5, 8, 5]$
`press_button()`	2	$\{0, 1, 3\}$	$[5, 8, 5]$
`move_inside(2)`		$\{0, 1, 2, 3\}$	$[5, 8, 9, 5]$
`move_inside(4)`		$\{0, 1, 2, 3, 4\}$	$[5, 8, 9, 5, 9]$
`move_inside(5)`		$\{0, 1, 2, 3, 4, 5\}$	$[5, 8, 9, 5, 9, 9]$
`press_button()`	3	$\{0, 1, 2, 3, 4, 5\}$	$[5, 8, 9, 5, 9, 9]$
`move_inside(5)`		$\{0, 1, 2, 3, 4, 5\}$	$[5, 8, 9, 5, 9, 9]$
`press_button()`	3	$\{0, 1, 2, 3, 4, 5\}$	$[5, 8, 9, 5, 9, 9]$
`move_outside(5)`		$\{0, 1, 2, 3, 4\}$	$[5, 8, 9, 5, 9]$
`press_button()`	2	$\{0, 1, 2, 3, 4\}$	$[5, 8, 9, 5, 9]$

At this point, there is sufficient information to conclude that the cardinality of the rarest insect type is 1. Therefore, the procedure `min_cardinality` should return 1.

In this example, `move_inside` is called 7 times, `move_outside` is called 1 time, and `press_button` is called 6 times.

Constraints

- $2 \leq N \leq 2000$

Subtasks

1. (10 points) $N \leq 200$
2. (15 points) $N \leq 1000$
3. (75 points) No additional constraints.

If in any of the test cases, the calls to the procedures `move_inside`, `move_outside`, or `press_button` do not conform to the constraints described in Implementation Details, or the

return value of `min_cardinality` is incorrect, the score of your solution for that subtask will be 0.

Let q be the **maximum** of the following three values: the number of calls to `move_inside`, the number of calls to `move_outside`, and the number of calls to `press_button`.

In subtask 3, you can obtain a partial score. Let m be the maximum value of $\frac{q}{N}$ across all test cases in this subtask. Your score for this subtask is calculated according to the following table:

Condition	Points
$20 < m$	0 (reported as "`Output isn't correct`" in CMS)
$6 < m \leq 20$	$\frac{225}{m-2}$
$3 < m \leq 6$	$81 - \frac{2}{3}m^2$
$m \leq 3$	75

Sample Grader

Let T be an array of N integers where $T[i]$ is the type of insect i.

The sample grader reads the input in the following format:

- line 1: N
- line 2: $T[0]\ T[1]\ \ldots\ T[N-1]$

If the sample grader detects a protocol violation, the output of the sample grader is `Protocol Violation: <MSG>`, where `<MSG>` is one of the following:

- `invalid parameter`: in a call to `move_inside` or `move_outside`, the value of i is not between 0 and $N-1$ inclusive.
- `too many calls`: the number of calls to **any** of `move_inside`, `move_outside`, or `press_button` exceeds $40\,000$.

Otherwise, the output of the sample grader is in the following format:

- line 1: the return value of `min_cardinality`
- line 2: q