Найрідкісніші комахи

У будинку Пака Бланґкона бігають N комах, пронумерованих від 0 до $N-1$. Кожна комаха має **тип**, який є цілим числом від 0 до 10^9 включно. Кілька комах можуть мати один і той же тип.

Припустимо, комахи згруповані за типами. Ми визначаємо потужність **найпоширенішого** типу комах, як кількість комах у групі з найбільшою кількістю комах. Подібним чином, потужність **найрідкіснішого** типу комах – це кількість комах у групі з найменшою кількістю комах.

Наприклад, припустимо, що є 11 комах, типи яких $[5, 7, 9, 11, 11, 5, 0, 11, 9, 100, 9]$. У цьому випадку потужність **найпоширенішого** типу комах становить 3. Групи з найбільшою кількістю комах — це тип 9 і тип 11, кожна з яких складається з 3 комах. Потужність **найрідкіснішого** типу комах становить 1. Групи з найменшою кількістю комах — це тип 7, тип 0, і тип 100, кожна з яких складається з 1 комахи.

Пак Бланґкон не знає типу жодної комахи. У нього є пристрій з однією кнопкою, який може надати деяку інформацію про типи комах. Спочатку пристрій порожній. Щоб використовувати пристрій, можна виконати три види операцій:

1. Засунути комаху у пристрій.
2. Забрати комаху з пристрою.
3. Натиснути кнопку на пристрої.

Кожен тип операції можна виконати щонайбільше $40\ 000$ разів.

Кожного разу, коли натискається кнопка, пристрій повідомляє потужність **найпоширенішого** типу комах, враховуючи лише комах усередині пристрою.

Ваше завдання - за допомогою пристрою визначити кількість **найрідкісніших** видів комах серед усіх N комах у домі Пака Бланґкона. Крім того, у деяких підзадачах ваш результат залежить від максимальної кількості виконаних операцій певного типу (докладніше див. у розділі Підзадачі).

Деталі реалізації

Ви повинні реалізувати таку функцію:

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

- N: кількість комах.

- Ця процедура має повернути потужність **найрідкіснішого** типу комах серед усіх N комах у домі Пака Бланґкона.
- Ця процедура викликається рівно один раз.

Наведена вище функція може викликати такі функції:

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

- i: номер комахи, яку потрібно засунути всередину пристрою. Значення i має бути від 0 до $N-1$ включно.
- Якщо ця комаха вже в пристрої, виклик не впливає на набір комах у пристрої. Однак він все ще зараховується як окремий виклик.
- Цю процедуру можна викликати щонайбільше $40\ 000$ разів.

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

- i: індекс комахи, яку потрібно забрати з пристрою. Значення i має бути від 0 до $N-1$ включно.
- Якщо ця комаха не в пристрої, виклик не впливає на набір комах у пристрої. Однак він все ще зараховується як окремий виклик.
- Цю процедуру можна викликати щонайбільше $40\ 000$ разів.

```
int press_button()
```

- Ця процедура повертає потужність **найпоширенішого** типу комах, враховуючи лише комах усередині пристрою.
- Цю процедуру можна викликати щонайбільше $40\ 000$ разів.
- Градер **не адаптивний**. Тобто, типи всіх N комах зафіксовані перед викликом `min_cardinality`.

Приклад

Розглянемо сценарій, у якому 6 комах типів $[5, 8, 9, 5, 9, 9]$ відповідно. Функція `min_cardinality` викликається таким чином:

```
min_cardinality(6)
```

Функція може викликати `move_inside`, `move_outside`, і `press_button` наступним чином.

Виклик	Повернене значення	Комахи у пристрої	Тип комах в пристрої
		$\{\}$	$[]$
`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	$\{0, 1, 2, 3, 4\}$	$[5, 8, 9, 5, 9]$

На даний момент є достатньо інформації, щоб зробити висновок, що потужність найрідкіснішого типу комах становить 1. Таким чином, функція `min_cardinality` повертає значення 1.

У цьому прикладі, `move_inside` викликається 7 разів, `move_outside` викликається 1 раз, а `press_button` викликається 6 разів.

Обмеження

- $2 \leq N \leq 2000$

Підзадачі

1. (10 балів) $N \leq 200$
2. (15 балів) $N \leq 1000$
3. (75 балів) Без додаткових обмежень.

Якщо в будь-якому з тестових випадків виклики функцій `move_inside`, `move_outside`, або `press_button` не відповідають обмеженням, описаним у Деталях реалізації, або значення, що

повертається `min_cardinality` є неправильним, оцінка вашого рішення для цієї підзадачі буде 0.

Нехай q буде **максимальним** з наступних трьох значень: кількість викликів `move_inside`, кількість викликів `move_outside`, і кількість викликів `press_button`.

У підзадачі 3 ви можете отримати частковий бал. Нехай m буде максимальним значенням $\frac{q}{N}$ серед усіх тестів у цій підзадачі. Ваш бал за цю підзадачу розраховується відповідно до наступної таблиці:

Умова	Бали
$20 < m$	0 (повідомлено як "`Output isn't correct`" в CMS)
$6 < m \leq 20$	$\frac{225}{m-2}$
$3 < m \leq 6$	$81 - \frac{2}{3}m^2$
$m \leq 3$	75

Приклад градера

Нехай T - це масив з N цілих чисел, де $T[i]$ - тип i-ї комахи.

Градер зчитує вхідні дані в такому форматі:

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

Якщо градер виявляє порушення протоколу, результатом градера є `Protocol Violation: <MSG>`, де `<MSG>` - одне з наступного:

- `invalid parameter`: під час виклику `move_inside` або `move_outside`, значення i не знаходиться від 0 до $N-1$ включно.
- `too many calls`: кількість викликів **будь-якого** з `move_inside`, `move_outside`, або `press_button` перевищує $40\ 000$.

Інакше градер виводить у наступному форматі:

- 1-й рядок: повернуте значення `min_cardinality`
- 2-й рядок: q