Insetti rari

Nella casa di Mister Blancone vivono N insetti, numerati da 0 a $N-1$. Ogni insetto ha un **tipo**, che è un intero compreso tra 0 e 10^9 inclusi. Più insetti possono avere lo stesso tipo.

Dato un sottoinsieme (non vuoto) di tali insetti, raggruppiamoli in base al tipo. Definiamo la cardinalità del tipo di insetto **più frequente** come il numero di insetti in uno dei gruppi con il numero maggiore di insetti. Definiamo la cardinalità del tipo di insetto **più raro** come il numero di insetti in uno dei gruppi con il numero minore di insetti.

Per esempio, consideriamo 11 insetti, con tipi $[5, 7, 9, 11, 11, 5, 0, 11, 9, 100, 9]$. In questo caso, la cardinalità del tipo **più frequente** è 3, e corrisponde a quella dei gruppi dei tipi 9 e 11. La cardinalità del tipo **più raro** è 1, e corrisponde a quella dei gruppi dei tipi 7, 0 e 100.

Mister Blancone non conosce i tipi degli insetti, ma ha lo Strumento™ che può fornirgli alcune informazioni a riguardo. All'inizio lo Strumento è vuoto, e per usarlo sono possibili tre operazioni:

1. Inserire un insetto nello Strumento.
2. Rimuovere un insetto dallo Strumento.
3. Premere l'unico pulsante presente sullo Strumento, di modo che esso riporti la cardinalità del tipo di insetto **più frequente** nel sottoinsieme di insetti che ha all'interno.

Ciascun genere di operazione può essere eseguito al più $40\,000$ volte.

Determina la cardinalità del tipo di insetto **più raro** tra tutti gli N insetti, usando lo Strumento! Nota che, in alcuni subtask, il tuo punteggio potrebbe dipendere dal massimo numero di operazioni di un dato genere che eseguirai.

Dettagli di implementazione

Devi implementare la seguente funzione:

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

- N: il numero di insetti.
- La funzione deve restituire la cardinalità del tipo di insetto **più raro** tra tutti gli N insetti.
- Questa funzione è chiamata esattamente una volta.

Tale funzione può effettuare chiamate alle seguenti altre funzioni:

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

- i: l'indice dell'insetto da inserire nello strumento, con i compreso tra 0 e $N - 1$.
- Se l'insetto i è già dentro lo strumento, la chiamata non modifica l'insieme di insetti dentro lo strumento. Tuttavia, è comunque contata come una chiamata separata.
- Questa funzione può essere chiamata al massimo $40\,000$ volte.

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

- i: l'indice dell'insetto da rimuovere dallo strumento, con i compreso tra 0 e $N - 1$.
- Se l'insetto i è già fuori dallo strumento, la chiamata non modifica l'insieme di insetti dentro lo strumento. Tuttavia, è comunque contata come una chiamata separata.
- Questa funzione può essere chiamata al massimo $40\,000$ volte.

```
int press_button()
```

- Questa funzione restituisce la cardinalità del tipo di insetto **più frequente** nel sottoinsieme di insetti che sono all'interno dello strumento, o 0 se lo strumento è vuoto.
- Questa funzione può essere chiamata al massimo $40\,000$ volte.

Il grader **non è adattivo**, vale a dire, i tipi di tutti gli N insetti è fissato prima della chiamata a `min_cardinality`.

Caso di esempio

Considera uno scenario con 6 insetti di tipi $[5, 8, 9, 5, 9, 9]$. La funzione `min_cardinality` viene chiamata nel modo seguente:

```
min_cardinality(6)
```

La funzione potrebbe chiamare `move_inside`, `move_outside`, e `press_button` come segue:

Chiamata	Valore di ritorno	Insetti nello strumento	Tipi degli insetti nello strumento
		$\{\}$	$[]$
`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	$\{0, 1, 2, 3, 4\}$	$[5, 8, 9, 5, 9]$

A questo punto, ci sono informazioni sufficienti per concludere che la cardinalità del tipo più raro è 1. Pertanto, la funzione `min_cardinality` può restituire 1.
In questo esempio, `move_inside` è stata chiamata 7 volte, `move_outside` è stata chiamata 1 volta, e `press_button` è stata chiamata 6 volte.

Assunzioni

- $2 \leq N \leq 2000$.

Subtask

1. (10 punti) $N \leq 200$.
2. (15 punti) $N \leq 1000$.
3. (75 punti) Nessuna limitazione aggiuntiva.

Se, in un caso di test, una chiamata alla funzione `move_inside` o `move_outside` ha parametro fuori dal range ammesso, oppure il valore di ritorno di `min_cardinality` non è corretto, il punteggio della soluzione per il relativo subtask è 0.

Altrimenti, per i primi due subtask otterrai punteggio completo, mentre nell'ultimo subtask puoi ottenere un punteggio parziale.

Sia q il massimo numero di chiamate a una delle tre funzioni `move_inside`, `move_outside`, `press_button`. Sia m il massimo valore di $\frac{q}{N}$ tra tutti i casi di test di questo subtask. Il punteggio del subtask è calcolato così:

Condizione	Punti
$20 < m$	0 (riportato come "`Output isn't correct`" da CMS)
$6 < m \leq 20$	$\frac{225}{m-2}$
$3 < m \leq 6$	$81 - \frac{2}{3}m^2$
$m \leq 3$	75

Grader di esempio

Sia T un array di N interi, dove $T[i]$ è il tipo dell'insetto i.

Il grader di esempio legge l'input secondo il seguente formato:

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

Se il grader di esempio rileva una violazione di protocollo, l'output sarà `Protocol Violation: <MSG>`, dove `<MSG>` è uno tra:

- `invalid parameter`: in una chiamata a `move_inside` o `move_outside`, $i < 0$ o $i \geq N$.
- `too many calls`: il numero di chiamate a **una qualunque** tra `move_inside`, `move_outside`, o `press_button` supera $40\,000$.

Altrimenti, l'output del grader di esempio segue il seguente formato:

- riga 1: il valore di ritorno di `min_cardinality`
- riga 2: q