იშვიათი მწერები

გვყავს N ცალი მწერი, რომლებიც გადანომრილია 0 to $(N-1)$-ის ჩათვლით და დაფრინავენ პაკ ბლანგკონის სახლში. თითოეულ მწერს აქვს თავისი **ტიპი**, რომელიც არის მთელი რიცხვი 0-დან 10^9 ჩათვლით. ბევრი მწერი შეიძლება ერთი და იგივე ტიპის იყოს.

დავუშვათ, მწერები დაჯგუფებულია ტიპების მიხედვით. განვსაზღვროთ სიმძლავრე **უფრო ხშირი** მწერების ტიპისა, როგორც მწერების რაოდენობა ჯგუფში, სადაც ყველაზე მეტი მწერია. ანალოგიურად, სიმძლავრე **იშვიათი** მწერების ტიპისა არის მწერების რაოდენობა ჯგუფში, სადაც მწერების ყველაზე მცირე რაოდენობაა.

მაგალითად, დავუშვათ, რომ არსებობს 11 მწერი, რომელთა ტიპებია $[5, 7, 9, 11, 11, 5, 0, 11, 9, 100, 9]$. ამ შემთხვევაში სიმძლავრე **უფრო ხშირი** მწერების ტიპისა არის 3. ის ჯგუფები, სადაც ყველაზე მეტი მწერია, არის ტიპი 9 და ტიპი 11, თითოეული შეიცავს 3 მწერს. სიმძლავრე **იშვიათი** მწერების ტიპისა არის 1. ყველაზე ნაკლები მწერების მქონე ჯგუფები არის ტიპი 7, ტიპი 0, და ტიპი 100, თითოეული შეიცავს 1 მწერს.

პაკ ბლანგკონმა არ იცის არცერთი მწერის ტიპი. მას აქვს აპარატი ერთი ღილაკით, რომელსაც შეუძლია გარკვეული ინფორმაციის მოწოდება მწერების ტიპების შესახებ. თავდაპირველად, აპარატი ცარიელია. აპარატის გამოსაყენებლად შეიძლება შესრულდეს სამი სახის ოპერაცია:

1. გადაიყვანოს მწერი აპარატის შიგნით.
2. გამოიყვანოს მწერი აპარატის გარეთ.
3. დააჭიროს ღილაკს აპარატზე.

თითოეული ტიპის ოპერაცია შეიძლება შესრულდეს მაქსიმუმ $40\,000$-ჯერ.

ღილაკზე დაჭერისას, აპარატია აცნობებს **უფრო ხშირი** მწერების ტიპის სიმძლავრეს, მხოლოდ აპარატის შიგნით არსებული მწერების გათვალისწინებით.

თქვენი ამოცანაა დაადგინოთ სიმძლავრე **იშვიათი** მწერის ტიპისა, ყველა N მწერს შორის პაკ ბლანგკონის სახლში აპარატის გამოყენებით. გარდა ამისა, ზოგიერთ ქვეამოცანებში თქვენი ქულა დამოკიდებულია შესრულებული მოცემული ოპერაციების ტიპების მაქსიმალურ რაოდენობაზე (დაწვრილებით იხილეთ ქვეამოცანების განყოფილებაში).

იმპლემენტაციის დეტალები

თქვენ უნდა შექმნათ შემდეგი პროცედურა:

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

- N: მწერების რაოდენობა.
- ამ პროცედურამ უნდა დააბრუნოს ბლანგკონის სახლში ყველა N მწერს შორის **იშვიათი** ტიპის მწერების სიმძლავრე.
- ეს პროცედურა გამოიძახება ზუსტად ერთხელ.

ზემოხსენებულ პროცედურას შეუძლია გამოიძახოს შემდეგი პროცედურები:

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

- i: აპარატში გადასაყვანი მწერის ინდექსი. i-ს მნიშვნელობა უნდა იყოს 0-სა $(N-1)$-ის ჩათვლით.
- თუ ეს მწერი უკვე არის აპარატის შიგნით, ზარი არ ახდენს გავლენას მანქანაში მწერების ნაკრებზე. თუმცა ის მაინც ცალკე ზარად ითვლება.
- ეს პროცედურა შეიძლება გამოძახებულ იქნეს მაქსიმუმ $40\ 000$-ჯერ.

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

- i: მანქანის გარეთ გამოსაყვანი მწერის ინდექსი. i-ს მნიშვნელობა უნდა იყოს 0-სა და $(N-1)$-ის ჩათვლით.
- თუ ეს მწერი უკვე არის აპარატის გარეთ, ზარი არ ახდენს გავლენას აპარატში მწერების ნაკრებზე. თუმცა ის მაინც ცალკე ზარად ითვლება.
- ეს პროცედურა შეიძლება გამოძახებულ იქნეს მაქსიმუმ $40\ 000$-ჯერ.

```
int press_button()
```

- ეს პროცედურა დააბრუნებს **უფრო ხშირი** მწერების ტიპის სიმძლავრეს, მხოლოდ აპარატის შიგნით არსებული მწერების გათვალისწინებით.
- ეს პროცედურა შეიძლება გამოძახებულ იქნეს მაქსიმუმ $40\ 000$-ჯერ.

მაგალითი

განვიხილოთ სცენარი, რომელშიც არის $[5, 8, 9, 5, 9, 9]$ ტიპის 6 მწერი. პროცედურა `min_cardinality` გამოიძახება შემდეგნაირად:

```
min_cardinality(6)
```

პროცედურა გამოიძახებს `move_inside`, `move_outside`, და `press_button` შემდეგნაირად.

გამოძახება	დაბ.მნიშვ	მწერები აპარატში	მწერების ტიპი მანქანაში
		$\{\}$	$[]$
`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	$\{0,1,2,3,4\}$	$[5,8,9,5,9]$

ამ ეტაპზე საკმარისი ინფორმაციაა, რათა დავასკვნათ, რომ იშვიათი მწერების ტიპის სიმძლავრე არის 1. ამიტომ პროცედურა `min_cardinality`-მ უნდა დააბრუნოს 1.

ამ მაგალითში, `move_inside` გამოძახებულია 7-ჯერ, `move_outside` გამოძახებულია 1-ხელ და `press_button` გამოძახებულია 6-ჯერ.

შეზღუდვები

- $2 \leq N \leq 2000$

ქვეამოცანები

1. (10 ქულა) $N \leq 200$
2. (15 ქულა) $N \leq 1000$
3. (75 ქულა) შეზღუდვების გარეშე.

ნებისმიერი ტესტისას თუ `move_inside`, `move_outside`,ან `press_button` პროცედურების გამოძახება არ შეესაბამება იმპლემენტაციის დეტალებში აღწერილ დეტალებს ან

`min_cardinality` დაბრუნებული მნიშვნელობა არასწორია, თქვენი ქულა ამ ქვეამოცანისთვის იქნება 0.

დავუშვათ q არის **მაქსიმალური** შემდეგ მნიშვნელობებს შორის: `move_inside`-ის გამოძახებების რაოდენობა, `move_outside`-ის გამოძახებების რაოდენობა და `press_button`-ის გამოძახებების რაოდენობა.

მე-3 ქვეამოცანაში შეგიძლიათ მიიღოთ ნაწილობრივი ქულები. დავუშვათ m არის მაქსიმალური მნიშვნელობა $\frac{q}{N}$-სი ამ ქვეამოცანის ყველა ტესტს შორის. თქვენი ქულები ამ ქვეამოცანისთვის გამოითვლება შემდეგი ცხრილით:

პირობა	ქულა
$20 < m$	0 (CMS-ში შეტყობინებაა "`Output isn't correct`")
$6 < m \leq 20$	$\frac{225}{m-2}$
$3 < m \leq 6$	$81 - \frac{2}{3}m^2$
$m \leq 3$	75

სანიმუშო გრადერი

დავუშვათ T არის მასივი N მთელი რიცხვისა, სადაც $T[i]$ არის i მწერის ტიპი.

სანიმუშო გრადერი კითხულობს შემდეგნაირად:

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

თუ სანიმუშო გრადერი აღმოაჩენს პროტოკოლის დარღვევას, სანიმუშო გრადერი გამოიტანს `Protocol Violation: <MSG>`, სადაც`<MSG>` ერთ-ერთია შემდეგიდან:

- `invalid parameter`: `move_inside` ან `move_outside`-ს გამოძახებისას i არ არის 0-სა და $N-1$- შორის (ჩათვლით).
- `too many calls`: გამოძახებების რაოდენობა **ნებისმიერისა** `move_inside`, `move_outside`, ან `press_button` შორის აღემატება $40\,000$.
- გრადერი არის **not adaptive**. ანუ ყველა N მწერის ტიპი დაფიქსირდა `min_cardinality`-ის გამოძახებამდე.

წინააღმდეგ შემთხვევაში, სანიმუშო გრადერი გამოიტანს შემდეგი ფორმატით:

- სტრიქონი 1: `min_cardinality`-ის დაბრუნებული მნიშვნელობა.
- სტრიქონი 2: q.