Pilze zählen (mushrooms)

Pilzexperte Andrew möchte Pilzsorten in Singapur untersuchen.

Dafür hat er n Pilze gesammelt, durchnummeriert von 0 bis $n-1$. Jeder Pilz gehört zu einer von zwei Sorten, die mit A und B bezeichnet werden.

Andrew weiss, dass **Pilz 0 zur Sorte A gehört**. Weil jedoch beide Sorten gleich aussehen, kennt er die Sorten der Pilze 1 bis $n-1$ nicht.

Glücklicherweise hat Andrew eine Maschine in seinem Labor, die ihm dabei helfen kann. Die Maschine wird benutzt, indem man zwei oder mehr Pilze in einer Reihe hineinlegt (in beliebiger Reihenfolge). Sie bestimmt dann die Anzahl Paare **benachbarter** Pilze, die unterschiedlicher Sorte sind. Falls du zum Beispiel die Sorten $[A, B, B, A]$ (in dieser Reihenfolge) in die Maschine legst, gibt sie dir 2 zurück.

Da die Anwendung der Maschine sehr teuer ist, ist die Anzahl Benutzungen begrenzt. Ausserdem darf die Gesamtanzahl Pilze, die du über alle Benutzungen hinweg hineinlegst, $100\,000$ nicht überschreiten. Hilf Andrew, mit der Maschine herauszufinden, wie viele Pilze der Sorte A er gesammelt hat.

Implementierungsdetails

Du sollst die folgende Funktion implementieren:

```
int count_mushrooms(int n)
```

- n: die Anzahl Pilze, die Andrew gesammelt hat.
- Diese Funktion wird genau einmal aufgerufen und soll die Anzahl Pilze der Sorte A zurückgeben.

Innerhalb von `count_mushrooms` kannst du folgende Funktion aufrufen:

```
int use_machine(int[] x)
```

- x: ein Array mit einer Länge zwischen 2 und n (inklusive) mit den Nummern der Pilze, die in die Maschine gelegt werden, in der gewünschten Reihenfolge.
- Die Elemente von x müssen **unterschiedliche** Ganzzahlen von 0 bis $n-1$ (inklusive) sein.
- Sei d die Länge des Arrays x. Dann gibt die Funktion die Anzahl der Indizes j zurück, für die $0 \leq j \leq d-2$ gilt und für die Pilze $x[j]$ and $x[j+1]$ eine unterschiedliche Sorte haben.

- Die Funktion kann höchstens $20\ 000$ mal aufgerufen werden.
- Die Summe der Längen aller x, die an die Funktion `use_machine` in all ihren Aufrufen übergeben werden, darf nicht grösser sein als $100\ 000$.

Beispiele

Beispiel 1

Angenommen, es gibt 3 Pilze der Sorten $[A, B, B]$, in dieser Reihenfolge. Der Grader ruft die Funktion `count_mushrooms` wie folgt auf:

```
count_mushrooms(3)
```

Diese Funktion kann nun `use_machine([0, 1, 2])` aufrufen, was (in diesem Fall) 1 zurückgibt. Danach kann sie `use_machine([2, 1])` aufrufen, was 0 zurückgibt.

Nach diesen zwei Funktionsaufrufen lässt sich eindeutig bestimmen, dass es genau 1 Pilz der Sorte A gibt. Somit sollte die Funktion `count_mushrooms` 1 zurückgeben.

Beispiel 2

Angenommen, es gibt 4 Pilze der Sorten $[A, B, A, A]$, in dieser Reihenfolge. Der Grader ruft die Funktion `count_mushrooms` wie folgt auf:

```
count_mushrooms(4)
```

Diese Funktion kann nun `use_machine([0, 2, 1, 3])` aufrufen, was 2 zurückgibt. Danach kann sie `use_machine([1, 2])` aufrufen, was 1 zurückgibt.

Nach diesen zwei Funktionsaufrufen lässt sich eindeutig bestimmen, dass es genau 3 Pilze der Sorte A gibt. Somit sollte die Funktion `count_mushrooms` 3 zurückgeben.

Beschränkungen

- $2 \leq n \leq 20\ 000$

Bewertung

Du erhältst keine Punkte, falls deine Lösung in einem Testfall `use_machine` mit Argumenten aufruft, die nicht den oben erwähnten Bedingungen entsprechen. Ebenfalls erhältst du keine Punkte, wenn `count_mushrooms` in einem Testfall einen falschen Wert zurückgibt.

Ansonsten sei Q die maximale Anzahl Aufrufe von `use_machine` unter allen Testfällen. Deine

Lösung erhält wie folgt Punkte:

Bedingung	Punkte
$20\,000 < Q$	0
$10\,010 < Q \leq 20\,000$	10
$904 < Q \leq 10\,010$	25
$226 < Q \leq 904$	$\frac{226}{Q} \cdot 100$
$Q \leq 226$	100

In gewissen Testfällen ist der Grader adaptiv, d.h. er antwortet nicht gemäss einer vorher festgelegten Folge von Pilzsorten, sondern die Antworten des Graders können von den vorherigen Aufrufen von `use_machine` abhängen. Es ist jedoch garantiert, dass nach jeder Interaktion alle bisherigen Antworten mit mindestens einer Folge von Pilzsorten konsistent sind.

Beispiel-Grader

Der Beispiel-Grader liest ein Array s von Ganzzahlen, welches die Pilzsorten beschreibt. Für alle $0 \leq i \leq n-1$ bedeutet $s[i] = 0$, dass Pilz i von der Sorte A ist, und $s[i] = 1$, dass er von der Sorte B ist.

Der Beispiel-Grader liest die Eingabe im folgenden Format:

- Zeile 1: n
- Zeile 2: $s[0]$ $s[1]$ $\ldots$ $s[n-1]$

Der Beispiel-Grader gibt das Ergebnis im folgenden Format aus:

- Zeile 1: der Rückgabewert von `count_mushrooms`.
- Zeile 2: die Anzahl Aufrufe von `use_machine`.

Beachte, dass der Beispiel-Grader nicht adaptiv ist.