Weryfikator wejścia

Limit pamięci: 256 MB

Zastanawialiście się kiedyś jak to się dzieje że podczas konkursów zadania "same" się sprawdzają? Niestety "samo" nic się nie dzieje... Autorzy zadań mają z tym sprawdzaniem całkiem sporo roboty. "Paczka" którą przygotowuje twórca składa się z: treści zadania, rozwiązań wzorcowych, rozwiązań które działają za wolno, rozwiązań błędnych, generatora testów, weryfikatora wyjścia, weryfikatora wejścia. W zależności od konkursu, może być tego dużo więcej! W tym zadaniu będziecie mogli posmakować trochę, jak to jest po drugiej stronie.

Twoje zadanie polega na napisaniu weryfikatora wejścia do zadania **Kumkwaty**:
https://szkopul.edu.pl/c/mistrz-programowania-2024/p/r2e/.
Ale jak to? Czy najtrudniejsze zadanie z poprzedniej rundy przechodzi teraz na poziom B? Absolutnie **NIE**!

Nie musisz rozwiązać zadania **Kumkwaty**. Twoim celem jest powiedzieć które podzadanie spełniają pewne wygenerowane dane wejściowe dla zadania **Kumkwaty**. By lepiej zrozumieć przejdź na sam dół tego zadania, do sekcji **Ocenianie**. Jest 7 podzadań. Musisz sklasyfikować do którego podzadania należą wygenerowane dane wejściowe.

Inaczej mówiąc na wejściu dostaniesz pewien test, pewne dane wejściowe do zadania **Kumkwaty** czyli liczbę kupek n oraz n liczb a_i oznaczających rozmiary kupek. Twoim zadaniem jest wypisać jaki jest numer najmniejszego podzadania które spełnia ten test[1]. Podzadania do zadania **Kumkwaty** są takie same jak do tego zadania.

Wejście

Wejście i podzadania są dokładnie takie same jak w zadaniu Kumkwaty. W pierwszym i jedynym wierszu wejścia znajduje się jedna liczba całkowita n $(1 \leq n \leq 1\,000\,000)$. W drugim wierszu wejścia znajduje się n liczb całkowitych a_i $(1 \leq a_i \leq 1\,000)$.

Wyjście

W pierwszym wierszu podaj jedną liczbę całkowitą - numer najmniejszego podzadania które spełniają podane wejściowe.
Podzadania są opisane na dole w sekcji **Ocenianie**.
Możesz założyć że otrzymałeś(aś) prawidłowe dane wejściowe i spełniają przynajmniej jedno podzadanie z sekcji **Ocenianie**.

Przykłady

Wejście dla testu `r3b0a`:

```
4
8 4 2 1
```

Wyjście dla testu `r3b0a`:

```
3
```

Wyjaśnienie:
Podane dane wejściowe:
* * Spełniają podzadanie 7: Dane wejściowe mogą być dowolne
* * **Nie**spełniają podzadania 6: Podane n wynosi 4 i **nie** jest nieparzyste.
* * Spełniają podzadanie 5: Podane n wynosi 4 i jest parzyste.
* * Spełniają podzadanie 4: Podane n wynosi 4 i jest mniejsze równe od 1000
* * Spełniają podzadanie 3: Podane n wynosi 4 i jest mniejsze równe od 100
* * **Nie**spełniają podzadania 2: Wszystkie 4 dane wejściowe muszą być mniejsze lub równe 2
* * **Nie**spełniają podzadania 1: Wszystkie 4 dane wejściowe muszą równe 1
Najmniejszy numer podzadania który jest spełniony to 3. Wypisujemy: 3

[1] W prawdziwym weryfikatorze musielibyście jeszcze sprawdzić czy na pewno na wejściu jest odpowiednia ilość liczb i czy wszystkie dane są liczbami a nie np. napisami. Na szczęście do tego są specjalne narzędzia, więc (Uffff!) nie musisz tego robić

Weryfikator wejścia

Limit pamięci: 256 MB

Wejście dla testu `r3b0b`:

```
5
1 1 1 1 1
```

Wyjście dla testu `r3b0b`:

```
1
```

Wyjaśnienie: Ten test spełniają podzadania 7, 6, 4, 3, 2 oraz 1, z któych najmniejsze jest 1

Wejście dla testu `r3b0c`:

```
6
2 2 1 2 1 1
```

Wyjście dla testu `r3b0c`:

```
2
```

Wyjaśnienie: Ten test spełniają podzadania 7, 5, 4, 3 oraz 2, z których najmniejsze jest 2

Ocenianie

Podzadanie	Ograniczenia	Limit czasu	Punkty
1	$a_i = 1$ dla każdego i	1 s (C++) / 4 s (Python)	2
2	$a_i \leq 2$ dla każdego i	1 s (C++) / 4 s (Python)	10
3	$n \leq 100$	1 s (C++) / 4 s (Python)	14
4	$n \leq 1\,000$	1 s (C++) / 4 s (Python)	27
5	n jest parzyste	1 s (C++) / 4 s (Python)	19
6	n jest nieparzyste	1 s (C++) / 4 s (Python)	19
7	brak dodatkowych ograniczeń	1 s (C++) / 4 s (Python)	9