Krzysztof Diks, Lech Duraj
Tłumaczenie

Dostępna pamięć: 1500 MB, limit czasu: 4 s. *IOI 2015, dzień pierwszy, 28.07.2015*

Zespoły

W klasie jest N uczniów, ponumerowanych od 0 do $N-1$. Każdego dnia nauczyciel zadaje uczniom pewną liczbę projektów. Każdy z projektów musi zostać zrealizowany przez pewien zespół uczniów, tego samego dnia. Projekty mogą różnić się między sobą trudnością – o każdym projekcie wiadomo, jaka jest dokładna liczba uczniów w zespole, który powinien nad nim pracować.

Każdy uczeń ma własne zdanie co do wielkości zespołów, w jakich może się znaleźć. Ściślej mówiąc, uczeń numer i może należeć tylko do zespołu, który ma co najmniej $A[i]$ i co najwyżej $B[i]$ uczniów. Każdego dnia ucznia można włączyć do co najwyżej jednego zespołu, przy czym niektórzy uczniowie mogą pozostać bez przydziału.

Nauczyciel wybrał już projekty na najbliższe Q dni. Dla każdego z tych dni określ, czy możliwe jest podzielenie uczniów na zespoły tak, aby nad każdym projektem pracował jeden zespół.

Przykład

Załóżmy, że jest $N = 4$ uczniów i $Q = 2$ dni pracy. Życzenia uczniów podane są w poniższej tabeli:

uczeń	*0*	*1*	*2*	*3*
A	*1*	*2*	*2*	*2*
B	*2*	*3*	*3*	*4*

Pierwszego dnia są $M = 2$ projekty, wymagające zespołów o wielkości odpowiednio $K[0] = 1$ oraz $K[1] = 3$. Takie dwa zespoły można zbudować, przydzielając ucznia 0 do jednoosobowego zespołu, a pozostałych trzech uczniów łącząc w kolejny zespół.

Drugiego dnia znowu są $M = 2$ projekty, lecz tym razem potrzebne są zespoły o rozmiarach $K[0] = 1$ oraz $K[1] = 1$. Takiego przydziału zrealizować nie można, ponieważ tylko jeden uczeń chce pracować w zespole o wielkości 1.

Zadanie

Dane są opisy wszystkich uczniów: liczba N oraz tablice A i B, a także ciąg Q zapytań, z których każde dotyczy jednego dnia. Zapytanie zawiera liczbę M projektów do realizacji oraz ciąg K o długości M zawierający wielkości zespołów do sformowania. Dla każdego zapytania, Twój program musi odpowiedzieć, czy da się odpowiednio przydzielić uczniów do zespołów.

Musisz zaimplementować funkcje `init` *oraz* `can`*:*

- `init(N, A, B)` – *Program sprawdzający wywoła tę funkcję dokładnie raz.*
  - `N`*: liczba uczniów.*
  - `A`*: tablica długości* `N`*:* `A[i]` *jest minimalną możliwą wielkością zespołu dla ucznia i.*

  - `B`*: tablica długości* `N`*:* `B[i]` *jest maksymalną możliwą wielkością zespołu dla ucznia* i*.*
  - *Funkcja ta nie powinna zwracać żadnej wartości.*
  - *Możesz założyć, że* $1 \leqslant A[i] \leqslant B[i] \leqslant N$ *dla* $i = 0, \ldots, N-1$*.*

- `can(M, K)` *– Po jednokrotnym wywołaniu* `init`*, program sprawdzający wywoła tę funkcję* Q *razy, raz dla każdego dnia.*
  - `M`*: liczba projektów zadanych tego dnia.*
  - `K`*: tablica długości* `M` *zawierająca wymagane rozmiary zespołów dla kolejnych projektów.*
  - *Funkcja powinna zwracać 1, jeśli możliwe jest sformowanie odpowiednich zespołów, lub 0 w przeciwnym wypadku.*
  - *Możesz założyć, że* $1 \leqslant$ `M` $\leqslant N$ *oraz że dla każdego* $i = 0, \ldots,$ `M` $- 1$ *zachodzi* $1 \leqslant$ `K[i]` $\leqslant N$*. Zwróć uwagę, że suma wszystkich* `K[i]` *może przekroczyć* N*.*

Podzadania

Niech S *oznacza sumę wartości* `M` *we wszystkich wywołaniach* `can(M, K)`*.*

podzadanie	**liczba punktów**	N	Q	**dodatkowe warunki**
1	*21*	$1 \leqslant N \leqslant 100$	$1 \leqslant Q \leqslant 100$	*brak*
2	*13*	$1 \leqslant N \leqslant 100\,000$	$Q = 1$	*brak*
3	*43*	$1 \leqslant N \leqslant 100\,000$	$1 \leqslant Q \leqslant 100\,000$	$S \leqslant 100\,000$
4	*23*	$1 \leqslant N \leqslant 500\,000$	$1 \leqslant Q \leqslant 200\,000$	$S \leqslant 200\,000$

Przykładowy program sprawdzający

Przykładowy program sprawdzający czyta dane z wejścia w następującej postaci:

- *wiersz 1:* `N`
- *wiersze 2, . . . ,* `N + 1:` `A[i] B[i]`
- *wiersz* `N + 2:` `Q`
- *wiersze* `N + 3, . . . , N + Q + 2:` `M K[0] K[1] ... K[M - 1]`

Dla każdego zapytania program wypisuje wartość zwróconą przez funkcję `can`*.*