Koneser naleśników

21Bukowina07. Grupa A. Dzień 3. Pamięć 256 MB. Czas 1 sek.

Mały Miś udał się na coroczny konkurs smażenia naleśników. W tym roku naleśnikarze przygotowali no to wydarzenie N dań naleśnikowych i położyli je wszystkie w rzędzie. Najciekawszą częścią konkursu nie jest jednak ogłoszenie zwycięzcy, a to co się stanie z naleśnikami po konkursie.

Tak się złożyło, że Mały Miś otrzymał możliwość zakupu dowolnego przedziału naleśników leżących w wyżej wymienionym rzędzie. Dla każdego naleśnika istnieje wartość z zakresu od -10^9 do 10^9, która oznacza zysk (lub stratę w przypadku liczb ujemnych) na danym naleśniku. Niestety Mały Miś nie zna się na nowatorskich wymysłach początkujących naleśnikarzy, więc nie wie ile będzie w stanie zarobić na każdym z nich. Na szczęście postanowił mu pomóc znajomy koneser naleśników, który był już na wszystkich konkursach i jest w stanie bardzo dobrze odróżniać dobre naleśniki od słabych.

Koneser nie jest w stanie podać najkorzystniejszego przedziału naleśników. Jest on jednak w stanie stwierdzić dla dowolnej pary przedziałów, czy pierwszy z nich jest korzystniejszy od drugiego. Niestety koneser nie ma zbyt dużo czasu i suma długości przedziałów (później nazywa *kosztem*) we wszystkich zapytaniach nie może przekraczać 10^7.

Pomóż Małemu Misiowi określić przedział dań do zakupienia o największej łącznej sumie smaczności.

Interakcja

To jest zadanie interaktywne. W tym zadaniu Twojemu programowi nie wolno korzystać ze standardowego wejścia, ani wyjścia. Grozi to utratą punktów za test. Twoje rozwiązanie może korzystać ze standardowego wyjścia błędu może to jednak znacząco wpływać na wykonanie programu, w szczególności zużywać czas. Udostępnione zostaną Ci następujące funkcje:

void nalesniki();

Ta funkcja zwraca liczbę **N** ($1 \leq$ **N** $\leq 10^5$) – liczbę dostępnych na sprzedaż dań naleśnikowych (czyli też liczbę uczestników).

bool lepszy(int, int, int, int);

Ta funkcja porównuje dwa przedziały – przyjmuje cztery wartości **a**, **b**, **c** i **d** ($1 \leq$ **a**,**b**,**c**,**d** $\leq$ **N**, **a**<**b**, **c**<**d**). Zwraca prawdę jeżeli suma wartości na przedziale indeksów od **a** do **b** jest ściśle większa od analogicznej sumy od indeksu **c** do **d**. Spowoduje to doliczenie do kosztu **b-a+d-c+2**. Funkcja przerwie działanie programu jeżeli argumenty nie spełniają założeń lub wywołanie przekroczyłoby maksymalny koszt.

void odpowiedz(int, int);

Ta funkcja przyjmuje wynik Twojego programu – przedział w postaci dwóch liczb **a** i **b** ($1 \leq$ **a**,**b** $\leq$ **N**), który to powinien reprezentować przedział o największej sumie wartości. Jeżeli ta funkcja zostanie wywołana drugi raz to Twój program zostanie przerwany bez wpływu na wynik (pamiętaj, że zanim do tego dojdzie możesz spowodować błąd lub zużyć za dużo czasu, co może zmniejszyć przyznaną liczbę punktów).

Testowanie

Do zadania jest dołączone dodatkowe pliki. Powinny one zostać Ci dostarczone (np. w zakładce z plikami na stronie). Tam znajdować się powinny:

- konlib.h – Plik nagłówkowy zawierający definicje udostępnionych Ci funkcji;
- konlib.cpp – Plik źródłowy zawierający przykładową implementacje biblioteki sprawdzającej; Zawartość tego pliku nie pokrywa się z plikiem używanym do końcowego oceniania rozwiązań;
- konb.cpp – Plik źródłowy programu realizującego przykładową, poprawną interakcję. Kod źródłowy w tym pliku **nie jest** poprawnym rozwiązaniem tego zadania, a został udostępniony dla celów demonstracyjnych.

W celu skompilowania dostarczonych plików za pomocą kompilatora g++ można użyć następującej komendy: `g++ konlib.cpp konb.cpp -o kon` (wszystkie trzy wymienione pliki muszą znajdować się w folderze, gdzie następuje kompilacja). W pierwszej linii standardowego wejścia do tak skompilowanego programu powinna się znaleźć liczba naleśników **N** ($1 \leq \mathbf{N} \leq 10^5$), a w drugiej **N** liczb oznaczających zyski za kolejne naleśniki (zyski z zakresu od -10^9 do 10^9).

Przykład

Wejście	Wyjście
10 -4 5 -3 7 2 8 -9 9 -10 9	19