Zastosowanie sortowania

1. (*) Tomek ma zabawkę, z której wystają drewniane słupki różnej wysokości. Jednym uderzeniem młotka może wbić lub wysunąć wybrany słupek o 1.
   Napisz funkcję `int słupki(const std::vector<int> s)`, która dla danych początkowych wysokości słupków obliczy minimalną liczbę uderzeń młotka potrzebnych do wyrównania wysokości słupków.

2. [II OI](*) Napisz funkcję `bool trójkąt(const std::vector<int> v)`, która dla danego ciągu $\{x_1, x_2, \ldots, x_n\}$ dodatnich liczb całkowitych sprawdzi, czy taki ciąg zawiera trójkę elementów x_i, x_j, x_k (dla $i \neq j, j \neq k, i \neq k$) spełniających nierówność trójkąta, tzn.:

   $$2 \cdot \max(x_i, x_j, x_k) \leq x_i + x_j + x_k$$

   Podaj złożoność czasową i pamięciową swojego rozwiązania. (Uwaga: Jeżeli liczby całkowite mają ograniczony zakres, to można to zadanie rozwiązać w stałym czasie.)

3. Napisz funkcję `int przedział(const std::vector<int> x, int r)`, która dla danego ciągu $\{x_1, \ldots, x_n\}$, oraz dla liczby całkowitej $r \geq 0$ obliczy taką liczbę całkowitą c, że $|\{i : |x_i - c| \leq r\}|$ jest maksymalne.
   Przykład: `przedział({2, -2, 5, -1, 11, 8, 4, 5, 8, 7}, 2) == 6`.

4. (*) Dany jest ciąg liczb rzeczywistych $\{x_1, x_2, \ldots, x_n\}$. Napisz funkcję `std::vector<float> przekładaniec(const std::vector<float> x)`, której wynikiem jest taka permutacja $\{x_{p_1}, x_{p_2}, \ldots, x_{p_n}\}$ danego ciągu, dla której suma:

   $$\sum_{i=1}^{n-1} |x_{p_{i+1}} - x_{p_i}|$$

   jest największa.

5. Mamy n kubeczków z wodą, ponumerowanych od 1 do n ($n > 0$). W kubeczku nr i znajduje się x_i wody, przy czym $0 < x_1 \leq x_2 \leq \cdots \leq x_n$. Powtarzamy k razy (dla $k < n$) nastepującą czynność: wybieramy dwa (niepuste) kubeczki zawierające jak najmniej wody i wodę z jednego kubeczka dolewamy do drugiego.
   Napisz procedurę `float zlewki(const std::vector<float> x, int k)`, która dla danych liczb $\{x_1, \ldots, x_n\}$ oraz liczby k określa ile jest wody w najmniej wypełnionym kubeczku po wykonaniu k operacji przelewania.

   ?

6. Tomek chciałby popłynąć z kolegami jachtem w rejs trwający k dni. Potrzebuje zebrać grupę kolegów, którzy chcą i mogą z nim popłynąć. Od każdego ze swoich kolegów dowiedział się od kiedy do kiedy (włącznie) dysponuje wolnym czasem. Teraz zastanawia się, kiedy wyruszyć w rejs, tak żeby mogli żeglować jak największą grupą. W trakcie rejsu koledzy Tomka mogą się zmieniać (o ile obie osoby mają tego samego dnia wolne).
   Napisz funkcję `int rejs(int k, const std::vector<int> start, const std::vector<int> end)`, która mając daną liczbę k, początki oraz końce dostępności kolegów Tomka, zwróci maksymalną wielkość załogi (nie licząc Tomka).

   Jeżeli rejsu nie da się zorganizować, to poprawnym wynikiem jest 0. Na przykład: `rejs(20, {12, 48, 28, 55, 0, 25}, {36, 100, 70, 80, 65, 30}) == 3`.

7. Tomek chciałby popłynąć z kolegami jachtem w rejs. Potrzebuje zebrać grupę k kolegów, którzy chcą i mogą z nim popłynąć. Od każdego ze swoich kolegów dowiedział się od kiedy do kiedy (włącznie) dysponuje wolnym czasem. Teraz zastanawia się, kiedy wyruszyć w rejs, tak żeby mógł on trwać jak najdłużej. W trakcie rejsu załoga nie może się zmieniać.
   Napisz procedurę `int rejs(int k, const std::vector<int> start, const std::vector<int> end)`, która mając daną liczbę k oraz informacje o dostępności kolegów Tomka, zwróci maksymalną długość rejsu. Jeżeli rejsu nie da się zorganizować, to poprawnym wynikiem jest 0.

   Na przykład: `rejs(2, {12, 48, 28, 50, 0, 25}, {36, 100, 70, 80, 69, 30}) == 42`.

Last modified: Sunday, 27 November 2022, 4:43 PM