Próg kwalifikacyjny (rozwiązanie)

XIV OIJ, zawody I stopnia, tura ukryta
14 października 2019 – 13 stycznia 2020

Autor zadania: **Karol Pokorski**
Opracowanie: **Dawid Jamka, Kacper Walentynowicz**

Olimpiada Informatyczna Juniorów

Rozpoczniemy od bardzo prostej obserwacji: zadania zawsze opłaca się rozwiązywać od tego dającego najwięcej punktów. Jeśli na przykład Bajtazar chce rozwiązać dwa zadania, nie ma sensu, aby były to inne zadania niż dwa najwyżej punktowane. Posortujemy zatem wszystkie zadania od tego z największą punktacją do tego z najmniejszą punktacją i załóżmy, że Bajtazar zawsze będzie chciał rozwiązać pewien początkowy fragment (*prefiks*) tak otrzymanej listy.

Dla naszego przykładu `kwa0a` z zadania:

```
6
3 2 1 1 5 1
...
```

zadania będziemy rozwiązywali w kolejności $V = [5, 3, 2, 1, 1, 1]$.

Zastanówmy się teraz jak szybko dowiadywać się, dla każdego d, ile punktów zdobędziemy rozwiązując d najbardziej punktowanych zadań. Najprostszą metodą jest po prostu sumowanie, za każdym razem, wartości pierwszych d elementów ciągu V. Możemy jednak te sumy (zwane *sumami prefiksowymi*) obliczyć raz, na początku programu, i zapamiętać w osobnej tablicy (lub liście) $sumy[0..n-1]$. Dla każdego d, wartość $sumy[d]$ będzie sumą pierwszych $d+1$ liczb z tablicy V, czyli liczbą punktów, jakie otrzymamy za rozwiazanie zadań $V[0], V[1], \ldots, V[d]$. Dla naszego przykładu ($V = [5, 3, 2, 1, 1, 1]$), ciągiem sum prefiksowym będzie $sumy = [5, 8, 10, 11, 12, 13]$.

Ciąg sum prefiksowych możemy łatwo uzyskać obliczając go od lewej do prawej. Każda kolejna suma prefiksowa (poza pierwszą, równą $V[0]$) jest poprzednią sumą, powiększoną o wartość następnego elementu tablicy V. Możemy zatem użyć następującego prostego algorytmu:

```
sumy[0] = V[0]
for i in 1, 2, ..., dlugosc V - 1
  sumy[i] = V[i] + sumy[i-1]
```

Teraz musimy szybko odpowiadać na zapytania (potencjalne progi kwalifikacyjne). Zauważmy, że teraz każde zapytanie P sprowadza się do znalezienia najmniejszej liczby w ciągu sum prefiksowych, która jest nie mniejsza od P. Zauważmy, że ciąg sum prefiksowych jest posortowany. Nie musimy zatem przeglądać całego ciągu, a możemy użyć algorytmu **wyszukiwania binarnego**. Algorytm ten jest dokładnie opisany np. w kursie podstaw algorytmiki portalu MAIN2: `https://www.main2.edu.pl/main2/courses/show/7/5/`.

To nie jest jedyne rozwiązanie tego zadania. Inne rozwiązanie bazuje na metodzie dwóch wskaźników, która była używana w zadaniu *Liczbowy proces*.

kwa.cpp

```cpp
#include "bits/stdc++.h"

using namespace std;

int main() {
  ios_base::sync_with_stdio(0);

  // Deklarujemy i wczytujemy liczbę zadań.
  int N;
  cin >> N;

  // Deklarujemy wektor na wartości punktowe zadań.
  vector <long long> V(N);
  // Wczytujemy punktację zadań.
  for (int i=0; i<N; i++) cin >> V[i];

```


Olimpiada finansowana jest ze środków Ministerstwa Edukacji Narodowej w ramach zadania publicznego „Organizacja i przeprowadzenie olimpiad i turniejów w latach szkolnych 2019/2020, 2020/2021, 2021/2022".

```cpp
  // Sortujemy liczbę punktów od największej do najmniejszej.
  sort(V.begin(), V.end(), greater<long long>());

  // Liczymy sumy prefiksowe.
  // Warto zwrócić uwagę, że wartości tych sum nie mieszczą się
  // w typie int, dlatego używamy long longów.
  vector <long long> sumy(N);
  sumy[0] = V[0];
  for (int i=1; i<N; i++) {
    sumy[i] = sumy[i-1] + V[i];
  }

  // Deklarujemy i wczytujemy liczbę zapytań
  // (potencjalnych progów punktowych).
  int Q;
  cin >> Q;

  // Dla każdego zapytania,
  while (Q--) {
    // deklarujemy i wczytujemy jego wartość
    long long P;
    cin >> P;
    // i wykorzystujemy wyszukiwanie binarne.
    // Ustawiamy początek i koniec przedziału, w którym wyszukujemy,
    int start = 0, koniec = N-1;
    // a następnie póki przedział ten jest niepusty,
    while(start <= koniec) {
      // wybieramy środkowy element z tego przedziału,
      int srodek = (start+koniec)/2;
      // i porównujemy jego wartość z naszym zapytaniem,
      // odpowiednio zmieniając jeden z końców przedziału.
      if (sumy[srodek] < P) start = srodek+1;
      else koniec = srodek-1;
    }
    // Finalnie wypisujemy liczbę zadań, powiększoną o jeden
    // (otrzymujemy wskaźnik na tablicę zadań, numerowaną od 0).
    cout << start+1 << "\n";
  }
}
```

kwa.py

```python
def main():
  # Wczytujemy liczbę zadań.
  N = int(input())

  # Wczytujemy punktację zadań.
  V = list(map(int, input().split()))

  # Sortujemy liczbę punktów od największej do najmniejszej.
  V.sort(reverse=True)

  # Liczymy sumy prefiksowe.
  sumy = [V[0]]
  for i in range(1, N):
    sumy.append(sumy[-1] + V[i])

  # Wczytujemy liczbę zapytań
  # (potencjalnych progów punktowych).
  Q = int(input())
```


Olimpiada finansowana jest ze środków Ministerstwa Edukacji Narodowej w ramach zadania publicznego „Organizacja i przeprowadzenie olimpiad i turniejów w latach szkolnych 2019/2020, 2020/2021, 2021/2022".

```
19
20    # Dla każdego zapytania,
21    for _ in range(Q):
22      # wczytujemy jego wartość
23      P = int(input())
24      # i wykorzystujemy wyszukiwanie binarne.
25      # Ustawiamy początek i koniec przedziału, w którym wyszukujemy,
26      start, koniec = 0, N-1
27      # a następnie póki przedział ten jest niepusty
28      while start <= koniec:
29        # wybieramy środkowy element z tego przedziału,
30        srodek = (start+koniec)//2
31        # i porównujemy jego wartość z naszym zapytaniem,
32        # odpowiednio zmieniając jeden z końców przedziału.
33        if sumy[srodek] < P: start = srodek+1
34        else: koniec = srodek-1
35      # Finalnie wypisujemy liczbę zadań, powiększoną o jeden
36      # (otrzymujemy wskaźnik na tablicę zadań, numerowaną od 0).
37      print(start+1)
38
39
40  main()
```


Olimpiada finansowana jest ze środków Ministerstwa Edukacji Narodowej w ramach zadania publicznego „Organizacja i przeprowadzenie olimpiad i turniejów w latach szkolnych 2019/2020, 2020/2021, 2021/2022".