Basen (rozwiązanie)

XIV OIJ, zawody I stopnia, tura otwarta
30 września – 16 grudnia 2019

Autor zadania: **Bartosz Kostka**
Opracowanie: **Krzysztof Bartoszek, Alicja Kluczek**

Zauważmy, że gdyby zegarek Bajtka zapisywał każdy przepłynięty metr, to pomiary tworzyłyby na zmianę ciąg rosnący i malejący. Na przykład, gdyby długość basenu wynosiła 5, to pomiary wyglądałyby następująco:

$$0, 1, 2, 3, 4, 5, 4, 3, 2, 1, 0, 1, 2, \ldots.$$

Płynąc od brzegu, nasze odległości rosną, a gdy wracamy – maleją. Moglibyśmy dokładnie stwierdzić, w którym momencie Bajtek zawraca – pomiary muszą albo muszą przestać rosnąć, a zacząć maleć, bądź odwrotnie – zacząć maleć, zamiast rosnąć. Każdy taki zwrot to kolejna długość basenu.

My mamy jednak wybrakowany zegarek, w którym niektórych pomiarów brakuje. Dodatkowo Bajtek mógł się zatrzymywać, dlatego niektóre pomiary mogą się powtarzać. Naszym zadaniem jest oszacować, jaką najmniejszą liczbę basenów przepłynął chłopiec.

Najpierw załóżmy, że kolejne pomiary się nie powtarzają. Okazuje się, że możemy zastosować powyżej opisaną metodę do obliczenia minimalnej liczby przepłyniętych długości basenu. Dopóki pomiary tworzą ciąg tego samego „typu" (czyli tworzą ciąg rosnący lub malejący), nie musieliśmy zawracać, zatem nie zwiększamy wyniku, zakładając, że Bajtek cały czas płynie w jednym kierunku. Dopiero kiedy kolejne pomiary zmienią nasz „typ" (przestają maleć, a zaczynają rosnąć, bądź odwrotnie), zwiększamy wynik, dodając kolejną długość basenu.

Zauważmy teraz, że to że niektóre długości basenu się nie powtarzają, nie utrudnia zbytnio zadania. Jeżeli jakiś pomiar się powtarza, zakładamy że Bajtek odpoczywał. Możemy zatem oczekiwać, że zamiast ciągów rosnących i malejących będziemy oczekiwać ciągów niemalejących (takich, gdzie każdy kolejny pomiar jest większy bądź równy poprzedniemu) oraz nierosnących (takich, gdzie każdy kolejny pomiar jest mniejszy bądź równy poprzedniemu).

Sprawdźmy zatem, jak będzie działał nasz pomysł na teście `bas0a` z treści zadania:

```
7
3 7 11 11 6 2 4
```

Mamy siedem pomiarów: $[3, 7, 11, 11, 6, 2, 4]$. Na początku wiemy, że zawsze płyniemy od brzegu basenu, gdzie mamy 0, zatem oczekujemy ciągu niemalejącego. Ciąg niemalejący mamy przez pierwsze cztery pomiary, tj. $[3, 7, 11, 11]$. Po drugiej jedenastce, kolejny pomiar jest mniejszy od poprzedniego, dlatego pomiędzy tymi pomiarami nastąpił zwrot. Kolejne dwa pomiary tworzą ciąg nierosnący ($[6, 2]$). W końcu, $2 < 4$, dlatego pomiędzy tymi dwoma pomiarami, też musiał nastąpić zwrot. Finalnie $[4]$ wyznacza ostatni ciąg niemalejący.

Innymi słowy, pomiędzy każdymi sąsiednimi pomiarami możemy wstawić znak mniejszości/równości/większości:

$$3 < 7 < 11 = 11 > 6 > 2 < 4$$

Pomijając znaki równości, za każdym razem kiedy zmienia się znak (z $<$ na $>$, bądź odwrotnie), musieliśmy zawrócić. Wynik zatem to liczba zwrotów plus jeden (początkowa długość basenu).

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".

bas.cpp

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

using namespace std;

int main() {
  // Wyłączenie synchronizacji.  Przyśpiesza operacje wczytywania/wypisywania.
  ios_base::sync_with_stdio(0);

  // Deklarujemy i wczytujemy liczbe pomiarów.
  int N;
  cin >> N;

  // Zmienna określająca ile przepłynęliśmy już długości basenu (wynik).
  // Pierwszy basen zaczynamy już przepływać na początku.
  int przeplyniete = 1;
  // Zmienna utrzymująca, jaka była poprzednio wskazana wartość na zegarku Bajtka.
  // -1 jest dla nas wygodną wartością początkową, bo wszystkie pomiary są dodatnie.
  int poprzedni = -1;
  // Zmienna określająca, czy aktualnie oddalamy się od brzegu, czy zbliżamy do niego.
  // Jak zaczynamy, to odpływamy od brzegu.  Dlatego rosnący jest równe true (prawdzie).
  bool rosnacy = true;

  // Musimy obsłużyć N pomiarów.
  for (int i=0; i<N; i++) {
    // Deklarujemy zmienną na pomiar i wczytujemy ją.
    int p;
    cin >> p;
    // Jeżeli do tej pory oddalaliśmy się od brzegu, ale to się teraz zmieniło
    // (to jest nowy pomiar jest mniejszy od poprzedniego),
    if (rosnacy == true && poprzedni > p) {
      // to znaczy, że musieliśmy zawrócić, zatem dodajemy nowy basen
      przeplyniete++;
      // i zmieniamy kierunek na przeciwny.
      rosnacy = false;
    // Analogicznie jeżeli do tej pory zbliżaliśmy się do brzegu,
    // a nowy pomiar jest większy od poprzedniego,
    } else if (rosnacy == false && poprzedni < p) {
      // to znaczy, że też musieliśmy odbić od brzegu, zatem zwiększamy wynik
      przeplyniete++;
      // i zmieniamy kierunek na przeciwny.
      rosnacy = true;
    }
    // Pamiętamy, aby zaktualizować ostatni pomiar.
    poprzedni = p;
  }

  // Finalnie, wypisujemy wynik - liczbę przepłyniętych basenów.
  cout << przeplyniete << "\n";
}
```


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".

bas.py

```python
# Wczytujemy liczbę pomiarów.
N = input()

# Wczytujemy tablice pomiarów.
# Pobrane pomiary konwertujemy z napisów na liczby.
P = [int(x) for x in input().split()]

# Zmienna określająca ile przepłynęliśmy już długości basenu (wynik).
# Pierwszy basen zaczynamy już przepływać na początku.
przeplyniete = 1
# Zmienna utrzymująca, jaka była poprzednio wskazana wartość na zegarku Bajtka.
# -1 jest dla nas wygodną wartością początkową, bo wszystkie pomiary są dodatnie.
poprzedni = -1
# Zmienna określająca, czy aktualnie oddalamy się od brzegu, czy zbliżamy do niego.
# Jak zaczynamy, to odpływamy od brzegu.  Dlatego rosnący jest równe true (prawdzie).
rosnacy = True

# Dla każdego pomiaru:
for p in P:
  # jeżeli do tej pory oddalaliśmy się od brzegu, ale to się teraz zmieniło
  # (to jest nowy pomiar jest mniejszy od poprzedniego),
  if rosnacy == True and poprzedni > p:
    # to znaczy, że musieliśmy zawrócić, zatem dodajemy nowy basen
    przeplyniete += 1
    # i zmieniamy kierunek na przeciwny.
    rosnacy = False
  # Analogicznie jeżeli do tej pory zbliżaliśmy się do brzegu,
  # a nowy pomiar jest większy od poprzedniego,
  elif rosnacy == False and poprzedni < p:
    # to znaczy, że też musieliśmy odbić od brzegu, zatem zwiększamy wynik
    przeplyniete += 1
    # i zmieniamy kierunek na przeciwny.
    rosnacy = True
  # Pamiętamy, aby zaktualizować ostatni pomiar.
  poprzedni = p

# Finalnie, wypisujemy wynik - liczbę przepłyniętych basenów.
print(przeplyniete)
```


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".