Zadanie „Włamanie do galerii”

#max #min

W tym podrozdziale zajmiemy się rozwiązaniem prostego zadania z konkursu **Kodowanie z Kocurrem** – *Włamanie do galerii*. Oryginalny pomysł zadania pochodzi z serwisu Codeforces (zadanie 1041A).

Generalnie problem wygląda tak: w galerii znajdowało się K obrazów ponumerowanych kolejno począwszy od pewnej liczby naturalnej x. Oto zbiór numerów obrazów:

$$\{x, x+1, x+2, \ldots, x+K-2, x+K-1\}.$$

Liczby K oraz x nie są podane.

Po włamaniu w galerii pozostało n obrazów o numerach dość przypadkowo wybranych z powyższego zbioru:

$$a_1, a_2, \ldots, a_n.$$

Numery a_i podane są w losowym porządku i nie powtarzają się.

Mając daną liczbę n oraz numery a_i należy obliczyć, jaka jest najmniejsza możliwa ilość skradzionych obrazów.

Zauważmy, że na to pytanie możemy odpowiedzieć, jeśli będziemy znać najmniejszą i największą liczbę spośród numerów a_i, oznaczmy je odpowiednio min_a oraz max_a. Od liczby min_a do liczby max_a mamy tyle numerów obrazów, ile wynosi ich różnica plus 1:

$$max\_a - min\_a + 1.$$

W galerii pozostało n obrazów, zatem jeśli od powyższego wyrażenia odejmiemy jeszcze liczbę n, otrzymamy ilość brakujących numerów z tego przedziału. I to będzie właściwa odpowiedź na pytanie postawione w zadaniu – tyle przynajmniej obrazów skradziono. Być może złodzieje wzięli więcej obrazów: o numerach mniejszych od min_a lub większych od max_a, ale nie mamy żadnych możliwości obliczenia rzeczywistej ilości skradzionych płócien, a tylko mamy wyznaczyć tę najmniejszą ilość:

$$max\_a - min\_a + 1 - n.$$

Jak znaleźć te skrajne wartości? Na przykład, najpierw założymy, że min_a jest jakąś duuużą liczbą, większą od jakiegokolwiek dopuszczalnego numeru – z treści zadania wynika, że liczby a_i nie są większe od miliarda, zatem możemy przyjąć, że min_a jest równe miliard plus 1. Następnie będziemy wczytywać kolejno liczby a_i: jeśli wczytana liczba a_i jest mniejsza od min_a, to wtedy min_a otrzyma wartość a_i. I tak, aż do końca danych wejściowych. Ostatecznie zmienna min_a będzie miała wartość najmniejszej z liczb a_i. Analogicznie postępujemy z max_a, tylko jako jej wartość początkową przyjmiemy 0 i będziemy się pytać, czy kolejne a_i jest większe od dotychczasowej wartości max_a.

Obydwa powyższe poszukiwania możemy połączyć i wykonać bezpośrednio podczas wczytywania danych. Jako pierwszą daną wczytamy liczbę n. Ponieważ liczb a_i nie będziemy przechowywać, więc wystarczy nam jedna zmienna a, którą zadeklarujemy wewnątrz pętli wczytującej dane.

Nasz program może wyglądać tak:

```
#include <bits/stdc++.h>
using namespace std;

int main()
{
  int n;
  cin >> n;
  int min_a{ 1000000001 }, max_a{ };
  for(int i{ 1 }; i <= n; i++)
  {
    int a;
    cin >> a;
    if(a < min_a)
      min_a = a;
    if(a > max_a)
      max_a = a;
  }
  cout << max_a - min_a + 1 - n << '\n';
  return 0;
}
```

Jest OK, ale możemy to zgrabniej zapisać. Otóż mamy do dyspozycji dwie użyteczne funkcje: `min()` oraz `max()`. Każda z nich ma dwa argumenty i zwraca odpowiednio mniejszy oraz większy z nich. Zatem możemy się pozbyć instrukcji warunkowych i zapisać program w ten sposób:

```
#include <bits/stdc++.h>
using namespace std;

int main()
{
  int n;
  cin >> n;
  int min_a{ 1000000001 }, max_a{ };
  for(int i{ 1 }; i <= n; i++)
  {
    int a;
    cin >> a;
    min_a = min(a, min_a);
    max_a = max(a, max_a);
  }
  cout << max_a - min_a + 1 - n << '\n';
  return 0;
}
```

Proponujemy przetestować ten program na przykładach z treści zadania, a potem wysłać go na serwis Szkopuł do sprawdzenia.