Editorial RoAlgo PreOJI 2025

1-8 Martie 2025

Cuprins

1 Multumiri

Acest concurs nu ar fi putut avea loc fără următoarele persoane:

- Andrei Iorgulescu, Ștefan Dăscălescu, Traian Danciu, Vlad Munteanu, autorii și propunătorii problemelor și laureați la concursurile de informatică și membri activi ai comunității RoAlgo;
- Alex Vasiluță, fondatorul și dezvoltatorul principal al Kilonova;
- Ștefan Alecu, creatorul acestui șablon LaTeXpe care îl folosim;
- Andrei Chertes, David Curcă, Tudor Iacob, Susan, testerii concursului, care au dat numeroase sugestii și sfaturi utile pentru buna desfășurare a rundei;
- Ștefan Dăscălescu, Andrei Iorgulescu și Luca Mureșan, coordonatorii rundelor;
- Comunității RoAlgo, pentru participarea la acest concurs.

2 Minimax

Autor: Stefan Dascalescu

2.1 Soluția oficială

Un algoritm foarte simplu care este suficient de rapid atunci când k este mic este acela de a simula procesul de aflare a cifrei minime și maxime de k ori, crescând numărul dat cu produsul obținut. Din păcate, acest algoritm va fi mult prea încet pentru a obține punctajul maxim.
Pentru a optimiza algoritmul de mai sus, va trebui să observăm că nu are sens să mai rulăm algoritmul dacă la un moment dat, vom ajunge să avem cifra minimă egală cu 0, deoarece produsul va rămâne mereu constant, deci și numărul în sine nu se va schimba. Întrebarea care se pune acum este de ce acest algoritm este suficient de rapid?
O primă idee la care ne putem gândi este aceea că produsul maxim pe care îl putem avea nu este niciodată mai mare decât 81, așa că pentru orice număr, vom putea ajunge la o cifră a sutelor egală cu 0, ceea ce ne duce cu gândul la un algoritm care va avea nevoie de cel mult 900 de operații dacă ne raportăm la acest raționament. Totuși, se poate demonstra că acest număr de operații este foarte mic, în cel mai rău caz acesta fiind cel mult 60, deci și complexitatea algoritmului nostru este una optimă.

2.1.1 Cod sursă

Soluție de 100

3 Divizor

Autor: Stefan Dascalescu

3.1 Soluția oficială

Această problemă ne duce cu gândul la diverse cazuri pentru care putem obține ușor un răspuns. În primul rând, dacă avem un număr par mai mare sau egal cu 4, putem să-l scriem ca o sumă de două elemente egale, iar cel mai mare divizor comun să fie diferit de 1.
De asemenea, dacă avem un număr impar neprim n, putem să găsim unul din divizorii săi proprii și să-l scriem ca suma dintre x și $n - x$.
Singurul caz în care nu putem performa acest algoritm este dacă numărul în sine este prim.
Astfel, dacă intervalul nostru conține un număr compus, atunci vom folosi acest număr pentru a putea găsi perechea dată. Altfel, vom afișa -1.
Complexitatea algoritmului este dată de complexitatea algoritmului de aflare a primalității unui număr, după filtrarea cazurilor de bază.

3.1.1 Cod sursă

Soluție de 100