OLIMPIADA NAȚIONALĂ DE INFORMATICĂ, ETAPA NAȚIONALĂ
CLASA A VI-A
DESCRIEREA SOLUȚIILOR

COMISIA ȘTIINȚIFICĂ

Problema 1: Codificare

Propusă de: Ana Maria Arișanu, Colegiul Național "Mircea cel Bătrân", Râmnicu Vâlcea

Cerința 1:

Soluția 1: Pentru fiecare număr:

(1) se codifică conform regulii din enunț
(2) se compară numărul citit cu cel obținut prin codificare
(3) dacă numărul obținut prin codificare este mai mare, numărul citit se ia în calcul pentru minim și maxim.

Soluția 2: Se observă că un număr devine mai mare prin codificare doar dacă prima sa cifră este pară. Pentru fiecare număr:

(1) se determină prima sa cifră
(2) dacă prima cifră este pară, numărul citit se ia în calcul pentru minim și maxim.

În consecință, numărul de pași pe care-i efectuăm în total este $n \cdot 9$, unde 9 este numărul maxim de cifre.

Cerința 2:

Soluția 1: Fiecare număr de k cifre care are prima cifră cif se codifică (numerele sunt de la $\overline{cif00\cdots00}$ la $\overline{cif99\cdots99}$). Dacă numărul obținut prin codificare este palindrom, se numără.

Soluția 2: Fie $x = \overline{cifa_1\cdots a_{k-1}}$ un număr de k cifre ce are prima cifră cif.

Caz 1: $cif \neq 1$

Numărul obținut prin codificare este palindrom dacă și numai dacă au loc egalitățile $a_{k-1} = cif$, $a_1 = a_{k-2}$, $a_2 = a_{k-3}$ și așa mai departe. Rezultatul este $10^{\lfloor (k-1)/2 \rfloor}$.

Caz 2: $cif = 1$

Când $k = 2$, rezultatul este 10, numerele fiind $10, 11, 12, \ldots, 19$.

Când $k = 3$, rezultatul este 19, numerele fiind $100, 111, 122, \ldots, 199, 110, \cancel{111}, 112, 113, \ldots, 119$.

Când $k = 4$, rezultatul este 109, numerele fiind de forma $\overline{111a}$ (10 numere), $\overline{11aa}$ (cu $a \neq 1$, 9 numere) și $\overline{1aba}$ (cu $a \neq 1$, 90 de numere).

Când $k = 5$, rezultatul este 199, numerele fiind de forma $\overline{1111a}$ (10 numere), $\overline{111aa}$ (cu $a \neq 1$, 9 numere), $\overline{11aba}$ (cu $a \neq 1$, 90 de numere) și $\overline{1abba}$ (cu $a \neq 1$, 90 de numere).

Când $k = 6$, rezultatul este 1099, numerele fiind de forma $\overline{11111a}$ (10 numere), $\overline{1111aa}$ (cu $a \neq 1$, 9 numere), $\overline{111aba}$ (cu $a \neq 1$, 90 de numere), $\overline{11abba}$ (cu $a \neq 1$, 90 de numere) și $\overline{1abcba}$ (cu $a \neq 1$, 900 de numere).

Continuând raționamentul, rezultatul, în acest caz, este

$$\begin{cases} 2 \cdot 10^{\lfloor k/2 \rfloor} - 1 & \text{pentru } k \text{ impar,} \\ 10^{\lfloor k/2 \rfloor} + 10^{\lfloor (k-1)/2 \rfloor} - 1 & \text{pentru } k \text{ par.} \end{cases}$$

În consecință, numărul de pași pe care-i efectuăm în total este $\lfloor k/2 \rfloor$.

Problema 2: Esm

Propusă de: Dan Pracsiu, Liceul Teoretic "Emil Racoviță" Vaslui

Cerința 1: Se parcurge șirul și, pentru orice secvență de lungime 3 de forma (a_i, a_{i+1}, a_{i+2}), se verifică dacă $a_i \cdot a_{i+1} = a_{i+2}$.

Cerința 2: Considerăm $x = a_n$.

Deoarece numerele din șir sunt mai mici sau egale cu 100 000, construim un vector *poz* de lungime 100 000 în care, pentru fiecare $i \in \{1, 2, \ldots, 100\,000\}$, $poz_i = j$ are semnificația "cea mai din dreapta poziție unde apare numărul i în secvența $(a_1, a_2, \ldots, a_{n-1})$ este la poziția j". Dacă i nu apare în șir, atunci $poz_i = 0$.

Vom determina divizorii lui x și, pentru toate perechile de divizori (u, v) cu proprietatea că $u \cdot v = x$, aflăm cea mai din dreapta poziție P cu proprietatea că există o pereche (d_1, d_2) de divizori ai lui x astfel încât $poz_{d_1} \geq P$ și $poz_{d_2} \geq P$. De exemplu, dacă avem $n = 10$, $a = (7, 2, 3, 12, 8, 5, 4, 7, 6, 24)$, atunci $x = 24$ și $P = 7$, iar divizorii pereche ce au pozițiile după P sunt 4 și 6.

De aici obținem că numărul de secvențe esm de forma $(a_i, a_{i+1}, \ldots, a_n)$ este egal cu P. Aceste secvențe sunt: $(a_1, a_2, \ldots, a_n), (a_2, a_3, \ldots, a_n), \ldots, (a_P, a_{P+1}, \ldots, a_n)$. Algoritmul are $n - 1$ pași pentru construirea vectorului *poz*, la care se adaugă $\sqrt{x}$ pași pentru determinarea divizorilor pereche ai lui x.

Observație importantă: Este posibil ca x să fie pătrat perfect, caz în care avem perechea de divizori (r, r) a lui x, unde $r = \sqrt{x}$. Deci va trebui să reținem ultimele două poziții ale lui r, nu doar cea mai din dreapta.

Cerința 3: Pentru determinarea tuturor secvențelor din șir care sunt esm vom proceda asemănător soluției de la cerința 2, în care, pentru fiecare $i \in \{3, \ldots, n\}$, trebuie să aflăm câte secvențe esm se termină cu $x = a_i$. Construim vectorul *poz* cu secvența $(a_1, a_2, \ldots, a_{i-1})$ și aflăm divizorii pereche ai lui a_i. Apoi, pentru a trece la aflarea numărului de secvențe care se termină cu a_{i+1}, actualizăm $poz_{a_i} = i$ (adică îl introducem pe a_i în *poz*) și aflăm câte secvențe esm se termină cu $x = a_{i+1}$.

În consecință, numărul de pași pe care-i efectuăm în total este $n \cdot \sqrt{\text{MAX}}$, unde MAX este valoarea maximă din șirul a.

Problema 3: Sim

Propusă de: Iulian Oleniuc, Facultatea de Informatică, Universitatea "Alexandru Ioan Cuza" Iași

Cerința 1: Reținem valorile citite într-o matrice *arrow*. Prima cerință presupune să construim o a doua matrice *count*, astfel încât în $count[x][y]$ să reținem numărul de cetățeni care în a doua zi iau micul dejun în cafeneaua (x, y). Parcurgem matricea *arrow* dată și, pentru fiecare poziție (x_1, y_1), calculăm în (x_2, y_2) poziția unde se ajunge din (x_1, y_1) urmând indicatorul $arrow[x_1][y_1]$, după care incrementăm valoarea $count[x_2][y_2]$. La final, calculăm valoarea maximă din matricea *count*.

Complexitatea algoritmului este $\mathcal{O}(n^2)$.

Cerința 2 – Subtask 1: Pentru $n \leq 30$ și $k = 10^3$, este suficient să efectuăm o simulare brută a celor k zile. În acest sens, vom folosi două matrice $count_C$ și $count_P$, în care vom reține câți cetățeni iau micul dejun în fiecare cafenea în ziua curentă și, respectiv, în ziua precedentă.

Pentru prima zi, toate celulele matricei $count_P$ vor fi inițializate cu valoarea 1. La începutul fiecărei noi zile, conținutul matricei $count_C$ este copiat în $count_P$, după care $count_C$ este reinițializată cu zerouri.

Calculul lui $count_C$ în funcție de $count_P$ este similar procedeului de la prima cerință. Valoarea $count_P[x_1][y_1]$ este adăugată la $count_C[x_2][y_2]$, dar nu înainte de a adăuga la răspunsul final valoarea

$$2 \cdot count_C[x_2][y_2] \cdot count_P[x_1][y_1].$$

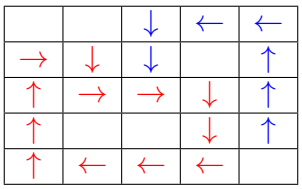
Această operație semnifică prima întâlnire dintre fiecare cetățean deja "ajuns" în (x_2, y_2) și fiecare cetățean ce urmează să "ajungă" în (x_2, y_2) din (x_1, y_1). Constanta 2 arată că fiecare întâlnire se contorizează de două ori – o dată pentru gradele de fericire ale celor din (x_2, y_2) și o dată pentru ale celor din (x_1, y_1).

De menționat că, pentru o implementare mai facilă, se poate recurge la tablouri tridimensionale; pentru acest subtask, limita de memorie o permite. Astfel, în loc de două matrice, putem folosi un tablou în care prima dimensiune să fie egală cu k, deci în care $count[i]$ să reprezinte matricea aferentă zilei i.

Complexitatea algoritmului este $\mathcal{O}(k \cdot n^2)$.

Cerința 2 – Subtask 2: În primul rând, observăm că, de la un punct încolo, traseul oricărui cetățean se va repeta, deoarece acesta va efectua o infinitate de pași, în timp ce orașul are un număr finit de cafenele. Mai mult, putem fi siguri că, în a n^2-a zi, nu va mai exista niciun cetățean care să nu fi intrat deja în partea repetitivă (în continuare, o vom numi "ciclu") a traseului său, deoarece lungimea maximă a acestuia este n^2.

Spre exemplu, în matricea de mai jos, traseul cetățeanului $(4, 5)$ este compus dintr-un prefix nerepetitiv (albastru) și un ciclu (roșu).



Începând (cel târziu) cu a n^2-a zi, nu se va mai produce nicio întâlnire nouă între doi cetățeni, din moment ce amândoi se află fie în cicluri diferite, fie pe poziții diferite din același ciclu. Nu în ultimul rând, trebuie menționat că, odată ce doi cetățeni se întâlnesc într-o cafenea anume, aceștia nu se vor mai despărți niciodată, deoarece, din acel punct încolo, ei vor urma exact același traseu.

În concluzie, este de ajuns să analizăm configurația orașului în ziua n^2. Dacă, în acea zi, într-o cafenea anume se află k cetățeni, înseamnă că gradul de fericire al fiecăruia dintre ei este $k - 1$, deoarece s-a întâlnit, de-a lungul timpului, doar cu ceilalți $k - 1$ cetățeni. Așadar, cafeneaua respectivă contribuie cu $k \cdot (k - 1)$ la răspunsul final.

Restricția $n \leq 30$ ne permite să implementăm o soluție în complexitate $\mathcal{O}(n^4)$, în care simulăm, pentru fiecare cetățean în parte, traseul său pe parcursul celor n^2 zile, determinându-i poziția finală.

O soluție alternativă, tot în complexitate $\mathcal{O}(n^4)$, dar în practică mult mai rapidă, presupune simularea fiecărei zile ca la primul subtask, până când procesăm o zi în care nu s-a produs nicio întâlnire nouă. Corectitudinea soluției se bazează pe faptul că ultima zi în care se produce o astfel de întâlnire este cea în care, pentru ultima oară, un cetățean intră în partea ciclică a traseului său, unde sigur dă de altcineva. Această soluție obține 80 de puncte.

Cerința 2 – Subtask 3: La finalul simulării, singurele celule din matrice care ne interesează sunt cele care fac parte dintr-un ciclu. În ordinea descoperirii acestor cicluri, putem eticheta celulele respective cu numere strict pozitive, de la 1, pe rând, până la cel mult n^2.

Așadar, ne dorim să determinăm, pentru fiecare cetățean (x, y), care este eticheta $e(x, y)$ a cafenelei în care se va afla acesta în ziua n^2. După ce vom face rost de această informație, nu va mai trebui decât să calculăm frecvențele etichetelor în matricea e de $n \times n$ etichete.

Completarea matricei e se efectuează printr-un proces similar celui de la al doilea subtask, dar optimizat. În acest sens, nu vom simula traseul unui cetățean decât până în momentul în care vom ajunge într-o celulă deja vizitată, fie de către el însuși (caz în care am identificat un ciclu nou), fie de către cineva dinaintea lui.

În cazul în care am găsit un ciclu nou, cu etichete de la l până la r, vom determina mai întâi valorile $e(x, y)$ pentru celulele sale, printr-o simplă formulă bazată pe l, r și eticheta inițială a celulei (x, y).

Mai departe, parcurgem în sens invers prefixul nerepetitiv al traseului găsit. Dacă am ajuns (în acest sens invers) în (x_1, y_1) din (x_2, y_2), iar (x_2, y_2) face parte dintr-un traseu al cărui ciclu este etichetat cu valori de la l la r, atunci

$$e(x_1, y_1) = \begin{cases} r & \text{pentru } e(x_2, y_2) = l, \\ e(x_2, y_2) - 1 & \text{altfel.} \end{cases}$$

Complexitatea soluției devine $\mathcal{O}(n^2)$, căci fiecare celulă din matrice este vizitată o singură dată. De remarcat că, pentru punctajul maxim, răspunsul trebuie reținut pe 64 de biți.

Echipa

Problemele pentru această etapă au fost pregătite de:

- Prof. Pintea Adrian Doru, Inspectoratul Școlar Județean Cluj, Cluj-Napoca
- Prof. Pracsiu Dan, Liceul Teoretic "Emil Racoviță", Vaslui
- Prof. Boca Alina Gabriela, Colegiul Național de Informatică "Tudor Vianu", București
- Prof. Iordaiche Eugenia-Cristiana, Liceul Teoretic "Grigore Moisil", Timișoara
- Prof. Tîmplaru Roxana Gabriela, Colegiul "Ștefan Odobleja", Craiova
- Prof. Arișanu Ana Maria, Colegiul Național "Mircea cel Bătrân", Râmnicu Vâlcea
- Prof. Georgescu Camelia Alice, Colegiul Național "Mihai Viteazul", Ploiești
- Prof. Oprița Petru Simion, Liceul "Regina Maria", Dorohoi
- Stud. Buzatu Giulian, Facultatea de Matematică și Informatică, Universitatea București
- Stud. Oleniuc Iulian, Facultatea de Informatică, Universitatea "Alexandru Ioan Cuza" Iași