TABĂRA DE PREGĂTIRE A LOTULUI NAȚIONAL DE INFORMATICĂ
26 - 31 IULIE 2023, CÂMPULUNG MUSCEL

DESCRIEREA SOLUȚIILOR, CUPA SEPI, JUNIORI

Problema 1: All numbers

Propusă de: prof. Chește Ciprian, Liceul Tehnologic "Grigore C. Moisil", Buzău

Cunoștințe necesare.

- Descompunere în factori primi
- Numărul divizorilor
- Combinatorică

Fie $N = p_1^{e_1} p_2^{e_2} .... p_k^{e_k}$ cu $e_i \neq 0, 1 \leq i \leq K$ și $S = e_1 + e_2 + ... + e_k$

Vom descompune numărul N în factori primi și vom reține într-o structură de date factorii primi împreună cu exponenții acestora. Se cunoaște că numărul de divizori ai lui N pote fi calculat cu formula $(e_1 + 1)(e_2 + 1)...(e_k + 1)$ Pentru a obține un număr maxim de divizori cu aceeași sumă a exponenților S, trebuie ca exponenții să fie cât mai apropiați, adică să difere cât mai puțin. Doar în acest mod se poate ajunge la un produs maxim pentru că și $S + K$ este constantă. Vom calcula câtul C și restul R al împărțirii dintre suma exponenților și numărul exponenților. Vom calcula apoi numărul T care are aceeași factori primi ca si N ridicați la C, urmând ca restul R să fie distribuit între cei K factori primi. De exemplu pentru $N = 945 = 3^3 5^1 7^1$ avem $S = 5, K = 3$ deci $C = 1$ și $R = 2$. Restul $R = 2$ poate fi distribuit la cei 3 factori în 3 moduri diferite așadar se pot obține numerele $3^2 5^2 7^1$, $3^2 5^1 7^2$ și $3^1 5^2 7^2$ adică 1575, 2205 și 36754 care au fiecare 18 divizori.

Se observă de aici că trebuie generate toate combinările de K luate câte R printr-un mecanism de tip backtracking sau un algoritm asemănător și apoi generate numerele căutate plecând de la numărul T. Aceste numere vor fi apoi însumate modulo $10^9 + 7$.

Soluție alternativă.

Propusă de: stud. Ștefan-Cosmin Dăscălescu, Universitatea din București

Mai întâi, descompunem N în factori primi și calculăm exponenții, împreună cu suma lor asemănător cu prima soluție.

Dacă numărul exponenților este x și suma lor este y, vrem ca fiecare exponent să apară cel puțin la puterea y/x ori și cel mult la puterea $y/x + 1$ ori, numărul exponenților ce apar la puterea $y/x + 1$ fiind dat de $y\%x$.

Astfel, vom putea folosi o abordare bazată pe metoda programării dinamice, $dp[i][j][k][l]$ fiind suma tuturor numerelor optime care au folosit primii i exponenti, numărul total de exponenți folosit este j, mai putem folosi k exponenți cu puterea q si l exponenți cu puterea $q + 1$, unde q este y/x de mai sus.

Pentru a face actualizarea de la o stare la următoarea stare, avem două cazuri în funcție de valorile din k și l, putând fie să adăugăm numărul prim curent la puterea q, fie același număr prim la puterea $q + 1$. Trebuie avut grijă la calcularea puterilor numerelor prime din descompunerea lui N pentru evitarea calculelor ce nu sunt necesare.

Complexitatea finală a soluției va fi $O(\sqrt{N} + x^3 * y)$, unde x și y sunt numerele de mai sus, x fiind cel mult egal cu 12 (numărul factorilor primi distincți ai lui N) iar y fiind cel mult egal cu 40. Totuși, un x mare nu permite și un y mare și viceversa.

Problema 2: Kpartit

Propusă de: prof. Victor Manz, Colegiul Național de Informatică „Tudor Vianu", București

Problema poate fi rezolvată folosind metoda programării dinamice.

Prima soluție, mai puțin eficientă, dar mai ușor de intuit constă în actualizarea în n pași a unei matrice cu k linii și $NL = 26$ coloane. Un element al acesteia, $lmax[nrs][p]$, are semnificația: lungimea maximă a unui subșir *kpartit* de ordinul *nrs* care se termină cu cea de-a p-a literă din alfabet.

La fiecare pas $i, 1 \leq i \leq n$, $lmax[nrs][p]$ se obține ca fiind maximul dintre $lmax[nrs][p]$ de la pasul $i-1$, la care se adaugă 1 și $max\{1 + lmax[nrs-1][j], 0 \leq j < NL, j \neq p\}$.

Rezultatul cerut este diferența dintre n și maximul elementelor de pe linia k a matricei de la pasul n (care corespunde întregului șir s). Cazul particular în care trebuie afișat -1 corespunde situației în care toate elementele liniei k sunt nule.

Se observă că matricea este actualizată în $k * NL$ pași, pentru fiecare dintre cele n prefixe ale șirului s citit. Rezultă astfel că algoritmul are complexitatea timp $O(n * k * NL)$ (NL reprezentând dimensiunea alfabetului, adică 26 în cazul nostru). O astfel de soluție va obține 81 de puncte.

Pentru a obține punctajul maxim trebuie făcută observația că nu este necesară parcurgerea tuturor celor $NL = 26$ de litere, ci este suficient să reținem doar pozițiile din alfabet pentru care se obțin cele mai mari două valori de pe fiecare linie *nrs* a matricei.

Notăm cu $indicemax1[nrs]$ = indicele (poziția în alfabet a) unei litere care poate fi ultima intr-un subșir *kpartit* de ordinul *nrs* de lungime maximă și cu $indicemax2[nrs]$ = indicele unei litere care poate fi ultima intr-un subșir *kpartit* de ordinul *nrs* de lungime maximă **dacă nu îl luăm în considerare** pe $indicemax1[nrs]$.

Cu ajutorul acestora, la fiecare pas $i, 1 \leq i \leq n$, $lmax[nrs][p]$ se obține ca fiind maximul dintre $lmax[nrs][p]$ de la pasul $i-1$, la care se adaugă 1 și:

- $1 + lmax[nrs-1][indicemax1[nrs-1]]$, dacă $p \neq indicemax1[nrs-1]$
- $1 + lmax[nrs-1][indicemax2[nrs-1]]$, dacă $p = indicemax1[nrs-1]$

După calculul fiecărui element $lmax[nrs][p]$, trebuie actulizate (dacă este cazul) $indicemax1[nrs]$ și $indicemax2[nrs]$. Trebuie avut grijă ca de fiecare dată să se păstreze relația $indicemax1[nrs] \neq indicemax2[nrs]$.

O astfel de soluție, având complexitatea timp *O(n*k)* și complexitatea spațiu *O(n*NL* obține punctajul maxim.

Problema 3: Semnal

Propusă de: prof. Chește Ciprian, Liceul Tehnologic "Grigore C. Moisil", Buzău

Etapa de raționament. Solutia problemei se încadrează în metoda *stars and bars* care este o abordare de rezolvare a problemelor de combinatorică. Se poate folosi când trebuie să determinăm numărul de modalități de a grupa un număr dat de obiecte identice.

Mai multe detalii despre această metodă puteți găsi aici: Stars and bars

Vom face o analiză în funcție de amplitudinea semnalului digital pe care o vom nota cu H sau cu alte cuvinte de diferența dintre nivelul logic 0 și nivelul logic 1.

- Pentru $H = 0$ se poate genera un singur semnal (o simplă linie dreaptă de lungime L). Acest rezultat este $comb(L+1, 0)$ pe care îl vom explica în continuare.
- Pentru $H = 1$ distingem subcazurile:
  (1) Dacă avem o singură trecere de la nivelul 0 la nivelul 1 și desigur o singură revenire atunci din lungimea L se scade 2, corespunzător celor două treceri de la un nivel la altul și avem apoi de așezat 2 linii verticale în L–$2+1$ poziții, adică $comb(L–1, 2)$
  (2) Dacă avem două treceri de la un nivel la altul atunci din L scădem 4 și vom avea de așezat 4 linii verticale în L–$4+1$ poziții adică $comb(L-3, 4)$
  (3) S.a.m.d. de unde vom găsi următoarea sumă de combinări $comb(L+1, 0) + comb(L-1, 2) + comb(L-3, 4) + ...$

- Pentru H = 2 raționamentul este asemănător și va conduce la suma de combinări $comb(L–3, 2) + comb(L-7, 4) + ...$

Sumele se vor calcula pentru toate valorile H care generează forme de undă, mai precis dacă L este număr impar atunci H poate fi $L/2$, iar dacă L este număr par, H poate fi $L/2–1$.

Exemplu. Pentru L = 10 avem:

- $H = 0$, $comb(11, 0) = 1$;
- $H = 1$, $comb(9, 2) + comb(7, 4) = 36 + 35 = 71$
- $H = 2$, $comb(7, 2) = 21$
- $H = 3$, $comb(5, 2) = 10$
- $H = 4$, $comb(3, 2) = 3$

În total sunt 106 semnale digitale distincte care se pot forma!

Etapa de implementare. În funcție de modul de calcul al combinărilor se pot distinge acum mai multe abordări de implementare:

- Combinări precalculate dinamic cu triunghiul lui Pascal dar cu valori limită de maxim 5000 – soluție de aproximativ $20 - 30$ puncte
- Combinări calculate clasic în care fiecare produs care apare în formula combinărilor se calculează de fiecare data împreună cu inversul modular necesar – soluție de aproximativa $30 - 40$ puncte
- Combinări în care fiecare produs este precalculat – soluție de aproximativ $40 - 50$ puncte.
- Combinări calculate cu teorema lui Lucas – soluție de $70 - 80$ puncte. Mai multe despre teorema lui Lucas puteți găsi aici: Teorema lui Lucas
- Combinări calculate în $O(1)$ după ce în prealabil au fost precalculate atât factoriale necesare cât și inversele modulare necesare – soluție de 100 puncte.

Soluție alternativă.

Propusă de: prof. Ionel-Vasile Piț-Rada, Colegiul Național Traian, Drobeta-Turnu Severin

Vom folosi următoarele notații:

- $f0[n]$ = numărul semnalelor de lungime n care se termină cu _ (orizontal jos)
- $f1[n]$ = numărul semnalelor de lungime n care se termină cu | (vertical urcare)
- $f2[n]$ = numărul semnalelor de lungime n care se termină cu $-$ (orizontal sus)
- $f3[n]$ = numărul semnalelor de lungime n care se termină cu | (vertical coborâre)

Ne interesează f0[L] + f3[L] Se observă relațiile : $f0[n] = f0[n-1] + f3[n-1]$, putem prelungi cu 1 pas orizontal în jos orice semnal care se termină cu orizontal jos sau coborâre verticală $f1[n] = f0[n-h]$, deoarece putem prelungi un semnal care se termină cu orizontal jos cu urcarea verticală de lungimea h $f2[n] = f1[n-1] + f2[n-1]$, putem prelungi cu 1 pas orizontal sus orice semnal care se termină cu orizontal sus sau urcare verticală $f3[n] = f2[n-h]$, deoarece putem prelungi un semnal care se termină cu orizontal sus cu coborâre verticală de lungimea h

Inițializare trebuie făcută cu atenție: f0[0] = 1,f1[0] = f2[0] = f3[0] = 0

Problema 4: Circles

Propusă de: stud. Andrei Onuț, Universitatea Yale, S.U.A.

Partea de Geometrie. Pentru fiecare traiectorie i: $1 \leq i \leq N$, va fi nevoie să calculăm valoarea $V(i)$. Să introducem, pentru simplitate, următoarele notații în cadrul traiectoriei i:

- $(x_{i,j}, y_{i,j}) = (x_j, y_j)$, pentru $j \in \{1, 2, 3\}$,
- $x_{C,i}$ = [abscisa centrului cercului ce descrie traiectoria i] = x_C,
- $y_{C,i}$ = [ordonata centrului cercului ce descrie traiectoria i] = y_C,
- $r(i)$ = [lungimea razei cercului ce descrie traiectoria i] = r.

Astfel, cu notațiile de mai sus, $V(i) = r^2 \cdot w(i)$. Presupunând că am avea o metodă prin care să determinăm valorile x_C și y_C, am putea calcula și valoarea r apoi.

Scriind de trei ori *ecuația carteziană a cercului*, care este listată și în cadrul secțiunii **Restricții** din enunț, obținem:

$$(x_1 - x_C)^2 + (y_1 - y_C)^2 = r^2, \tag{1}$$

$$(x_2 - x_C)^2 + (y_2 - y_C)^2 = r^2, \tag{2}$$

$$(x_3 - x_C)^2 + (y_3 - y_C)^2 = r^2. \tag{3}$$

Prin urmare, folosind, de exemplu, ecuația (1), obținem că $r = \sqrt{(x_1 - x_C)^2 + (y_1 - y_C)^2}$.

Acum, să ne concentrăm pe găsirea valorilor x_C și y_C, folosind, în continuare, cele trei ecuații de mai sus.

În cazul în care $x_1 = x_2$, obținem, în primul rând, că: $y_C = \frac{y_1 + y_2}{2}$ (din primele două ecuații). Apoi, folosind prima și cea de a treia ecuație, obținem că: $x_C = \frac{x_1^2 + y_1^2 - x_3^2 - y_3^2 - 2 \cdot y_1 \cdot y_C + 2 \cdot y_3 \cdot y_C}{2 \cdot (x_1 - x_3)}$. De remarcat că $x_1 \neq x_3$, deoarece cele trei puncte ce descriu traiectoria i sunt distincte și deja am afirmat că $x_1 = x_2$. Similar, se tratează cazurile $x_1 = x_3$ sau $x_2 = x_3$. De asemenea, folosind exact același raționament, putem rezolva și dacă $y_1 = y_2$, $y_1 = y_3$ sau $y_2 = y_3$, interschimând valorile x cu valorile y ale celor trei puncte.

Dacă se întâmplă simultan ca: $x_1 \neq x_2$, $x_1 \neq x_3$, $x_2 \neq x_3$, $y_1 \neq y_2$, $y_1 \neq y_3$ și $y_2 \neq y_3$, putem rezolva următorul sistem de două ecuații liniare pentru a afla cele două necunoscute x_C și y_C:

$$a \cdot x_C + b \cdot y_C = T,$$
$$c \cdot x_C + d \cdot y_C = Q,$$

unde:

- $T = x_1^2 + y_1^2 - x_2^2 - y_2^2$,
- $a = 2 \cdot (x_1 - x_2) \neq 0$,
- $b = 2 \cdot (y_1 - y_2) \neq 0$,
- $Q = x_1^2 + y_1^2 - x_3^2 - y_3^2$,
- $c = 2 \cdot (x_1 - x_3) \neq 0$,
- $d = 2 \cdot (y_1 - y_3) \neq 0$.

Expresiile T, a și b pot fi găsite scăzând primele două *ecuații ale cercului* una din cealaltă, iar expresiile Q, c și d pot fi găsite scăzând prima și cea de a treia *ecuație* una din cealaltă.

Pentru a rezolva sistemul de mai sus, putem folosi, de exemplu, *metoda substituției*: se află y_C din prima ecuație (în funcție de x_C) și apoi se înlocuiește această expresie în cea de a doua ecuație. Astfel, se află valoarea lui x_C. Apoi, se află și valoarea lui y_C.

De asemenea, pentru a determina $w(i)$ se poate folosi fie calcul de determinanți (ca și în cazul calculării, de exemplu, a ariei unui triunghi), fie se compară valorile pantelor celor două drepte suport ale segmentelor determinate de punctele (x_1, y_1) și (x_2, y_2), respectiv de punctele (x_2, y_2) și (x_3, y_3).

Meet-in-the-middle. Pentru a afla răspunsul la problemă, vom utiliza tehnica *Meet-in-the-middle*, despre care puteți afla mai multe pe *USACO Guide*.

Conform restricțiilor din enunț, știm că $A \cup B = \{1, 2, \ldots, N\}$ și $A \cap B = \emptyset$. Astfel, fiecare modalitate (A, B) prin care Sam și John își pot împărți între ei cele N traiectorii, respectând restricția bunicilor cu privire la numărul L, poate fi unic identificată folosind doar mulțimea A.

Să introducem: $M = \lfloor \frac{N}{2} \rfloor \geq 1$, unde $\lfloor \frac{N}{2} \rfloor$ reprezintă valoarea părții întregi a împărțirii numărului N la 2. Considerăm cele două *jumătăți* nevide ale șirului V: $l = [V_1, V_2, \ldots, V_M]$ (prima *jumătate*) și $r = [V_{M+1}, V_{M+2}, \ldots, V_N]$ (a doua *jumătate*). Astfel, $1 \leq |l| \leq |r| = \lfloor \frac{N+1}{2} \rfloor$.

Evident, mulțimea A are cel puțin un element, iar elementele ce constituie A pot face parte fie doar din mulțimea $\{1, 2, \ldots, M\}$, fie doar din mulțimea $\{M+1, M+2, \ldots, N\}$, fie din ambele mulțimi. Cu acest gând, atât pentru l, cât și pentru r, vom calcula în complexitatea de timp: $O(2^{\lfloor \frac{N+1}{2} \rfloor} \cdot \lfloor \frac{N+1}{2} \rfloor)$, utilizând, de exemplu, *reprezentarea în baza* 2 *a numerelor naturale*, toate perechile de forma $(sum(\sigma), |\sigma|)$, unde: σ este un subșir (format nu neapărat din elemente consecutive) al șirului l (sau al șirului r, pentru calculele ce implică elementele din a doua *jumătate* a șirului V), iar $sum(\sigma) =$ [suma elementelor din σ]. Trebuie să fim atenți la faptul că σ poate fi și vid, caz în care considerăm că $sum(\sigma) = |\sigma| = 0$.

Pentru fiecare subșir σ al șirului r, vom *insera* pe linia $|\sigma|$ a unei *colecții* (mai formal, o listă) bidimensionale valoarea $sum(\sigma)$. După această etapă, elementele fiecărei linii i (cu $0 \leq i \leq |r| = \lfloor \frac{N+1}{2} \rfloor$) din *colecție* vor trebui să ajungă în ordine non-descrescătoare. Linia i conține $\binom{|r|}{i} = \binom{\lfloor \frac{N+1}{2} \rfloor}{i}$ elemente. Astfel, [numărul maxim de elemente din cadrul unei linii a *colecției*] = $\binom{|r|}{\lfloor \frac{|r|}{2} \rfloor} = max_n$ (notație), din *Triunghiul lui Pascal*. Cele $2^{\lfloor \frac{N+1}{2} \rfloor}$ elemente ale colecției pot fi sortate în complexitatea de timp: $O(2^{\lfloor \frac{N+1}{2} \rfloor} \cdot log(2^{\lfloor \frac{N+1}{2} \rfloor})) = O(2^{\lfloor \frac{N+1}{2} \rfloor} \cdot \lfloor \frac{N+1}{2} \rfloor)$, înainte de a *insera* pe linia corespunzătoare fiecare element.

În continuare, să presupunem că am fixat cu ajutorul unui subșir σ (notații: $sum(\sigma) = sum_a$, $|\sigma| = cnt_a$) al șirului l valorile traiectoriilor din mulțimea $\{1, 2, \ldots, M\}$ ce fac parte dintr-o posibilă mulțime *soluție* A, pe care încercăm să o numărăm. Considerând că S este suma: $S = V_1 + V_2 + \ldots + V_N$, atunci dorim să numărăm câte subșiruri σ' (notații: $sum(\sigma') = sum_b$, $|\sigma'| = cnt_b$) ale șirului r respectă proprietatea:

$$|(sum_a + sum_b) \cdot (cnt_a + cnt_b) - (S - (sum_a + sum_b)) \cdot (N - (cnt_a + cnt_b))| \leq L.$$

Prelucrând inegalitatea de mai sus, obținem că:

$$-L - S \cdot (v - N) - sum_a \cdot N \leq sum_b \cdot N \leq L - S \cdot (v - N) - sum_a \cdot N,$$

unde: $v = (cnt_a + cnt_b)$ și v poate fi fixat: $cnt_a \leq v < N$ și $1 \leq v$ (mulțimea A este nevidă).

Astfel, pentru a număra câte subșiruri σ' satisfac inegalitatea de mai sus, se va *căuta binar* pe linia $(v - cnt_a)$ în *colecție*, în complexitatea de timp: $O(log(max_n))$, pentru fiecare pereche (σ, v).

Complexitatea totală de timp a algoritmului este: $O(2^{\lfloor \frac{N+1}{2} \rfloor} \cdot \lfloor \frac{N+1}{2} \rfloor + 2^{\lfloor \frac{N+1}{2} \rfloor} \cdot N \cdot log(max_n))$, adică: $O(2^{\lfloor \frac{N+1}{2} \rfloor} \cdot N \cdot log(max_n))$. În cazul în care $N = 34$, înseamnă că $max_n = \binom{17}{8} = 24\,310$.

Soluție alternativă - viitor stud. Cristian Verde. Problema presupune aflarea razei cercului circumscris celor 3 puncte, pe care o putem calcula găsind coordonatele centrului acestuia. Centrul este localizat la intersecția mediatoarelor laturilor triunghiului format de punctele (x_1, y_1), (x_2, y_2) și (x_3, y_3). Știm că mediatoarea unei laturi este perpendiculară pe aceasta, de unde aflăm panta (dacă $d1$ și $d2$ sunt drepte perpendiculare, atunci fie sunt una paralelă cu Ox și cealaltă cu Oy, fie respectă relația $a_{d1} \cdot a_{d2} = 1$, unde a_{d1} și a_{d2} sunt pantele celor 2 drepte), și faptul că trece prin mijlocul laturii, lucru ce ne permite să calculăm b-ul ecuației $y = a \cdot x + b$ pentru mediatoare, iar acum problema se reduce la o intersecție de drepte si distanța dintre centru și unul dintre puncte. Calculez $w(i)$ tot cu determinanți, mai exact luând semnul expresiei $(x_3 - x_2) \cdot (y_1 - y_2) - (x_1 - x_2) \cdot (y_3 - y_2)$, ce ne oferă semiplanul față de dreapta reprezentată de punctele (x_1, y_1) și (x_3, y_3) în care se află punctul (x_2, y_2).

Partea a doua. Având valorile vectorului V, ne mai rămâne de calculat numărul de moduri de a partiționa elementele sale în două submulțimi ce respectă relația din enunț. Rescriind, obținem:

$$|(V(a_1)+V(a_2)+\ldots+V(a_k)+V(b_1)+V(b_2)+\ldots+V(b_{N-k}))\cdot k-(V(b_1)+V(b_2)+\ldots+V(b_{N-k}))\cdot N|<=L$$

sau

$$|sum_V\cdot k-(V(b_1)+V(b_2)+\ldots+V(b_{N-k}))\cdot N|<=L$$

Cu alte cuvinte, vrem:

$$sum_V\cdot k+L>=sum_B\cdot N>=sum_V\cdot k-L, \tag{4}$$

unde B este o submulțime de $N-k$ elemente a lui V.

Soluție de complexitate $O(2^N)$ - 53 de puncte. Fixăm, utilizând o "mască pe biți", toate submulțimile posibile (scrierea fiecărui număr în baza 2 poate reprezenta o submulțime - dacă cifra de pe poziția i este 1 atunci elementul $V(i+1)$ se regăsește în ea). Așadar, printr-un for de la 1 la 2^N-2, avem

$sum_i = sum_{i\ xor\ 2^{first\_bit(i)}} + V(first\_bit(i)+1)$, unde $first\_bit(i) = 31 - \_\_builtin\_clz(i)$,

iar numărul de cifre de 1 din scrierea binară al lui i (pentru aflarea lui $N-k$) este $\_\_builtin\_popcount(i)$. Mai rămâne doar să contorizăm câte dintre acestea satisfac condiția de mai sus pentru o mulțime B.

Soluție de complexitate $O(2^{\frac{N}{2}}\cdot N)$ - 100 de puncte. Întocmai ca Andrei, folosesc tehnica "Meet-in-the-middle". Îmi împart elementele șirului în primele $[\frac{N}{2}]$ și ultimele $[\frac{N+1}{2}]$, pe care folosesc abordarea din soluția de complexitate $O(2^N)$, introduc valorile din *sum* în vectori STL în funcție de numărul de elemente ale submulțimii și îi sortez. Pentru fiecare i de la 1 la $N-1$, trebuie să calculăm în câte feluri putem combina o submulțime cu x elemente din primele $[\frac{N}{2}]$ și $i-x$ din ultimele $[\frac{N+1}{2}]$, astfel încât să respecte condiția (4) ($i=N-k$, iar $\frac{N}{2}>=x>=0$ și $\frac{N+1}{2}>=i-x>=0$). Având sumele deja sortate, putem aplica metoda Two Pointers, complexitatea acestei bucăți a algoritmului, la fel ca cea a sortării, fiind $O(2^{\frac{N}{2}}\cdot N)$.

Echipa

Problemele pentru această etapă au fost pregătite de:

- Nicoli Marius, Colegiul Național "Frații Buzești", Syncro Soft, Craiova
- Lica Daniela, Centrul Județean de Excelență Prahova
- Manz Victor, Colegiul National de Informatica "Tudor Vianu", București
- Dăscălescu Ștefan-Cosmin, Facultatea de Matematică și Informatică, Universitatea București
- Măgureanu Livia, Colegiul Național de Informatică "Tudor Vianu", București
- Chește Ciprian, Liceu Tehnologic "Grigore C. Moisil", Buzău
- Verde Flaviu-Cristian, Colegiul Național de Informatică "Tudor Vianu", București
- Onuț Andrei, Universitatea Yale, S.U.A.
- Popa Bogdan Ioan, Facultatea de Matematică și Informatică, Universitatea București
- Perju Verzotti Luca, Colegiul National de Informatica "Tudor Vianu", București
- Panaete Adrian, Colegiul Național "A.T. Laurian", Botoșani
- Cerchez Emanuela, Colegiul Național "Emil Racoviță", Iași
- Șerban Marinel, Colegiul Național "Emil Racoviță", Iași
- Piț-Rada Ionel-Vasile, Colegiul Național Traian, Drobeta-Turnu Severin
- Costineanu Raluca, Colegiul Național "Ștefan cel Mare", Suceava
- Bogdan Vlad Mihai, Facultatea de Matematică și Informatică, Universitatea București