RoAlgo Contest 5 - Editorial

Ștefan-Cosmin Dăscălescu, Traian Mihai Danciu, Eduard-Lucian Pîrțac

9 Septembrie 2023

1 Mulțumiri

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

- Traian Mihai Danciu, Ștefan-Cosmin Dăscălescu, Eduard-Lucian Pîrțac, autorii problemelor, laureați la concursurile de informatică și membri activi ai comunității RoAlgo.
- Alex Vasiluță, fondatorul și dezvoltatorul principal al Kilonova
- Roland Petrean, Raul Ardelean, Darius-Ramon Bejan, Andrei Chertes, Simion Lavrente, testerii concursului, care au dat numeroase sugestii și sfaturi utile pentru buna desfășurare a rundei, precum și ajutor la crearea unor teste.
- Simion Lavrente, creatorul wallpaper-ului rundei.
- Ștefan-Cosmin Dăscălescu, coordonatorul rundei si autorul solutiilor video.
- Comunității RoAlgo, pentru participarea la acest concurs.

2 Problema Palindrmoia

Autor: Ștefan-Cosmin Dăscălescu

2.1 Cerința 1

Pentru prima cerință, trebuie doar verificat pentru fiecare număr dacă este palindrom sau nu, folosind algoritmul clasic de a afla oglinditul numerelor sau folosind inversarea unui șir de caractere, dacă numărul este memorat ca un string.

2.2 Cerința 2

Pentru cea de-a doua cerință, trebuie aflată frecvența fiecărei cifre în număr, iar mai apoi trebuie verificat să avem cel mult o cifră care apare de un număr impar de ori. Dacă avem o asemenea cifră, trebuie să avem grijă și să avem cel puțin o cifră nenulă care apare de măcar două ori. Acest caz a fost tratat greșit de destui concurenți, fiind necesară o atenție sporită.

2.3 Cerința 3

Pentru cea de-a treia cerință, trebuie aflată frecvența fiecărei cifre în toate numerele, iar mai apoi, vom lua mereu doar numere de două cifre nenule, acesta fiind răspunsul maxim.

2.4 Cod sursă și soluție video

Soluție de 100

Soluție video

3 Problema ezuwu

Autor: Ștefan-Cosmin Dăscălescu

Pentru a rezolva această problemă, e necesar să observăm că putem memora folosind sume parțiale câți de u și câti de w avem, iar pentru fiecare query, este optim să aranjăm literele de w la mijloc, punând literele de u în mod echilibrat la marginile intervalului. Apoi, formula poate fi calculată ușor dacă știm numărul de u, n_u, respectiv numărul de w, n_w.

Astfel, răspunsul va fi $(n_u/2) \cdot n_w \cdot (n_u/2 + n_u \ mod \ 2)$.

3.1 Cod sursă și soluție video

Soluție de 100

Soluție video

4 Problema fulger

Autor: Ștefan-Cosmin Dăscălescu

Pentru a rezolva această problemă, este important să observăm că avem valorile ordonate crescător, deci putem mai întâi să considerăm toate subsecvențele și să verificăm manual pentru fiecare dintre ele, dar această abordare ne duce doar la un punctaj parțial, fiind prea înceată.

Pentru a ajunge la punctajul maxim, putem să ne folosim de faptul că dacă fixăm poziția de început a unei subsecvențe, poziția de final a unei subsecvențe de lungime minimă cu suma mai mare decât k va fi tot mai mare, ceea ce ne duce cu gândul la o implementare folosind metoda celor doi pointeri sau folosind căutarea binară, dacă precalculăm anterior sumele parțiale ale numerelor din șir. Pentru mai multe detalii de implementare, consultați sursele oficiale.

4.1 Cod sursă și soluție video

Soluție de 100 cu two pointers

Soluție de 100 cu căutare binară

Soluție video

5 Problema suporter

Autor: Ștefan-Cosmin Dăscălescu

Această problemă reprezintă o aplicație clasică a programării dinamice, unde $dp[i][j][p]$ reprezintă scorul maxim dacă după primele i etape, ținem cu echipa j și am schimbat echipa favorită de p ori. Această soluție poate fi implementată fie în $O(n^2 \cdot m \cdot k)$ făcând tranzițiile între perechile de echipe, sau în $O(n \cdot m \cdot k)$ dacă optimizăm abordarea calculând mai întâi cele mai mari două scoruri după fiecare etapă, mai apoi calculând tranzițiile doar bazându-te pe cele două maxime.

5.1 Cod sursă și soluție video

Soluție de 100

Soluție video

6 Problema ursul

Autor: Traian Mihai Danciu

6.1 Soluții parțiale

Pentru subtaskul 1, putem să calculăm suma iterativ. Pentru a calcula C_n^k, vom calcula $n \cdot (n-1) \cdot \cdots \cdot (n-k+1)$ și $k \cdot (k-1) \cdot (k-2) \cdot \cdots \cdot 1$ iterativ. Fie primul număr a și al doilea număr b. Rezultatul va fi $\frac{a}{b} \mod 998\,244\,353$. Din moment ce 998 244 353 este prim, putem afla inversul modular al lui b față de 998 244 353, cu formula $b^{998\,244\,353\,-\,2}$. Deci C_n^k va fi $a \cdot b^{998\,244\,351}$. Complexitate: O($N^3$) pentru fiecare test.

Pentru subtaskul 2, putem să precalculăm combinările folosind formula: $C_n^k = C_{n-1}^k + C_{n-1}^{k-1}$. Astfel putem să aflăm C_n^k în timp constant, complexitatea totală fiind O(N) pentru fiecare test.

Pentru subtaskul 3, putem să precalculăm $fact[i] = i! \mod 998\,244\,353$ și $invfact[i]$ = inversul modular al lui $i!$ față de 998 244 353. Ne vom folosi de următoarea recurență: $fact[i] = fact[i-1] \cdot i \mod 998\,244\,353$, iar $fact[0] = 1$. De asemenea, putem calcula $invfact[i]$ astfel: $invfact[i] = invfact[i+1] \cdot (i+1) \mod 998\,244\,353$.

Pentru subtaskul 4, putem precalcula toate răspunsurile.

6.2 Soluția completă

Autorii soluției: Traian Mihai Danciu, Roland Petrean

Problema ne cere $S = \sum_{i=0}^{n} C_n^i \cdot (n+i)$.

Iar $S = \sum_{i=0}^{n} C_n^i \cdot (n+i) = (\sum_{i=0}^{n} C_n^i \cdot n) + (\sum_{i=0}^{n} C_n^i \cdot i) =$
$(\sum_{i=0}^{n} C_n^i \cdot n) + (\sum_{i=0}^{n} C_n^i \cdot i)$.

Pentru a determina C_n^i ne putem gândi că din mulțimea $\{1, 2, \ldots, n\}$ alegem câte i elemente. Dar pentru a determina suma, pentru fiecare element, putem alege dacă îl luăm, sau nu, deci sunt $2 \cdot 2 \cdot \cdots \cdot 2$, de n ori, deci 2^n variante, astfel am determinat toata suma.

Pentru a determina $C_n^i \cdot i$, putem să ne gândim că din mulțimea $\{1, 2, .., n\}$ alegem i elemente, și putem alege 1 element "special", în i moduri. Dar putem face acest lucru și în alt mod, pentru a determina toată suma în mod eficient: mai întâi alegem elementul "special", în n moduri, iar apoi, pentru fiecare dintre celelalte elemente, alegem daca le luăm sau nu, în $2 \cdot 2 \cdot 2 \cdot \cdots \cdot 2$, de $n-1$ ori, deci 2^{n-1} moduri. Așa că vom avea $n \cdot 2^{n-1}$.

Deci, în total, avem
$S = 2^n \cdot n + 2^{n-1} \cdot n = n \cdot (2^n + 2^{n-1}) = n \cdot (2 \cdot 2^{n-1} + 2^{n-1}) = n \cdot 3 \cdot 2^{n-1}$.

6.3 Soluție alternativă

Autorul soluției: Andrei Chertes

Definim $k = \sqrt{998\ 244\ 353}$. Putem precalcula doi vectori, $p1$ și $p2$, unde $p1[i] = 2^i, 0 \leq i < k$ și $p2[i] = 2^{i \cdot k}$, atunci răspunsul pentru 2^n va fi $p1[n \mod k] \cdot p2[\lfloor \frac{n}{k} \rfloor] \mod 998\ 244\ 353$. Restul se rezolvă la fel ca la soluția principală. Cod sursă

6.4 Cod sursă și soluție video

Soluție de 100

Soluție video

7 Problema Bloxorz

Autor: Eduard-Lucian Pîrtac

7.1 Subtask 1

Soluție:

Nici măcar nu trebuie să simulăm mutările, trebuie doar să folosim puțină matematică.

Dacă începem dintr-o poziție verticală și ajungem într-o poziție verticală, înseamnă că mutările vor fi de către 3 pătrate fiecare. Ne mutăm doar într-o direcție, asta înseamnă că vom avea doar o coordonată care se schimbă. Să notăm cu a și b coordonata inițială respectiv coordonata finală. Condițiile ca să existe drum sunt $a = b \pmod 3$ și să nu existe niciun # între a și b. Astfel, numărul de mutări va fi 2· abs($a - b$) / 3, deoarece facem câte 2 mutări între 2 poziții verticale consecutive.

7.2 Subtask 2 și 3

Avem 2 variante, dfs (fill recursiv) sau bfs (lee cu coadă). Cea din urmă va obține punctaj maxim, în timp ce prima va obține 70 de puncte. Motivul este că recursivitatea reușește să "spargă" stiva pe testele mari (am încercat să o optimizez, dar nu merge mai bine de 1 GB :skull:). Puteți să combinați subtask-ul 1 cu subtask-ul 2 pentru 80 de puncte, dar nu e greu de modificat soluția ca să folosească bfs :).

7.3 Cod sursă și soluție video

Soluție cu dfs

Soluție cu lee cu coadă

Soluție video

8 Problema ABgraf

Autor: Ștefan-Cosmin Dăscălescu

Pentru a rezolva această problemă e important să observăm ce cazuri pot apărea când schimbăm tipul unui nod (schimbul din A în B sau din B în A se tratează identic). Pentru asta vom calcula mai întâi componentele conexe din cele două subgrafuri, memorând pentru fiecare nod componenta conexă în care se află.

Mai intâi, se poate schimba un nod din A în B, făcând subgraful format din nodurile de tipul B conex, lucru ce este adevărat dacă acel nod schimbat este adiacent cu noduri din toate componentele conexe pe care graful B le avea.

Un alt caz este dat de schimbarea unui nod din A în B, făcând subgraful format din nodurile de tipul A conex. Acest lucru este realizabil dacă A avea doar două componente conexe, iar acel nod înlocuit este izolat.

Nu în ultimul rând, dacă subgraful A sau B sunt conexe, răspunsul este DA.

8.1 Cod sursă și soluție video

Soluție de 100

Soluție video

9 Problema uwu

Autor: Ștefan-Cosmin Dăscălescu

Pentru a rezolva această problemă avem nevoie să putem afla eficient câte subșiruri uwu avem, lucru ce nu îl putem realiza folosind o metodă de tipul brute force ca la problema ezuwu. Pentru a putea optimiza soluția, putem folosi fie o abordare folosind arbori de intervale, în care în fiecare nod ținem date legat de câti de u, w, uw, wu, uwu avem, combinând cu atenție numărul de subșiruri de fiecare tip, fie o abordare bazată pe sume parțiale și diverse calcule ce se pot face folosind acele valori.

Deși abordarea cu sume parțiale ne duce la $O(1)$ per query, ambele abordări

sunt suficient de rapide pentru a lua punctajul maxim.

9.1 Cod sursă și soluție video

Soluție de 100 cu arbori de intervale

Soluție de 100 cu sume partiale

Soluție video