Editorial RoAlgo PreOJI 2025

1-8 Martie 2025

Cuprins

1 Multumiri

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

- Raul Ardelean, Ștefan Dăscălescu, Luca Mureșan, Stefan Vilcescu, autorii 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;
- Ștefan Alexandru Nuță, Ștefan Neagu, Raul Ardelean, Vlad Munteanu, Stefan Vilcescu, Tudor Iacob, Susan, Traian Danciu, 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 Caracterele Muzicale

AUTOR: ȘTEFAN VÎLCESCU

Pe durata acestui editorial, vom nota $n = len(s)$ si $m = len(t)$

2.1 Soluția de 16 puncte

Ne vom fixa intervalul $[l, r]$ din s pe care îl vom scoate, unde $1 \leq l \leq n$ și $l - 1 \leq r \leq n$, și vom verifica dacă se poate șterge intervalul $[l, r]$.
Pentru a verifica, vom face backtracking cu parametrii $(0, 0, l, r, n, m)$ la început. Vom numi funcția de backtracking *check*, iar parametrii acesteia înseamnă:
pozs = poziția curentă în șirul s.
pozt = poziția curentă în șirul t.
l, r = extremele intervalului ales.
n, m = lungimile șirurilor.
La un pas, vom verifica mai întâi dacă $pozt = m$. Dacă este adevărat, returnăm 1.
Altfel, dacă $pozs \geq l$ și $pozs \leq r$, vom seta $pozs = r + 1$.
Dacă $pozs = n$, returnăm 0.
Altfel, ne vom ține o variabilă x, inițializată cu 0.

Dacă $s_{pozs} = t_{pozt}$, vom face:

$$x = \max(x, check(pozs + 1, pozt + 1, l, r, n, m))$$

Apoi, vom face:

$$x = \max(x, check(pozs + 1, pozt, l, r, n, m))$$

La final, returnăm x.
Pentru toate intervalele $[l, r]$ pe care le fixăm pentru a le scoate, verificăm dacă $check(0, 0, l, r, n, m) = 1$. Dacă este adevărat, verificăm dacă răspunsul este mai mare decât $r - l + 1$.
Această soluție ar trebui să obțină 16 puncte.
Soluție de 16

2.2 Soluția de 34 de puncte

Vom încerca să optimizăm funcția $check(0, 0, l, r, n, m)$. Observăm faptul că, la pasul cu pozițiile $pozs$, respectiv $pozt$, este optim să alegem poziția poz, unde $poz > pozs$ și poz este minim, astfel încât $s_{poz} = t_{pozt}$.
Acest lucru este adevărat deoarece, dacă am alege o poziție după $poz1$, următoarele poziții pe care le vom alege vor fi mai mari decât $poz1$. Iar cum $poz1 > poz$, acestea vor fi mai mari și decât poz, deci tot va rămâne un subșir valid.
Soluție de 34

2.3 Soluția de 53 de puncte

În continuare, vom încerca să optimizăm funcția $check(0, 0, l, r, n, m)$. Ne vom ține doi vectori:

$pozl_i$ = poziția maximă $pozt$, astfel încât subsecența $[1, pozt]$ din t se află ca subșir în prefixul $[1, i]$ din s;

$pozr_i$ = poziția minimă $pozt$, astfel încât subsecența $[pozt, m]$ din t apare ca subșir în sufixul $[i, n]$;

Acești doi vectori se pot construi liniar, deoarece vom aplica observația de la punctul precedent, unde:

$pozl_i = pozl_{i-1}$, dacă s_i este diferit de $t_{pozl_{i-1}+1}$, altfel $pozl_i = pozl_{i-1} + 1$, pentru $1 \le i \le n$, și $pozl_0 = 0$.

$pozr_i = pozr_{i+1}$, dacă s_i este diferit de $t_{pozr_{i+1}-1}$, altfel $pozr_i = pozr_{i+1} - 1$, pentru $1 \le i \le n$, iar $pozr_{n+1} = m$.

Acum, ca să verificăm dacă subsecența $[l, r]$ o putem scoate, trebuie doar să verificăm dacă:

$$pozr_{r+1} \le pozl_{l-1} + 1$$

Deoarece ar apărea prefixul $[1, pozl_{l-1}]$ din t ca subșir în prefixul $[1, l - 1]$ din s și sufixul $[pozr_{r+1}, m]$ din t ca subșir în sufixul $[r + 1, n]$, rezultă că apare tot șirul t ca subșir în șirul s dacă am scoate intervalul $[l, r]$.

Dacă $pozr_{r+1} > pozl_{l-1} + 1$, atunci t nu ar apărea ca subșir, deoarece caracterul $t_{pozl_{l-1}+2}$ nu ar apărea în șirul s dacă am scoate intervalul $[l, r]$.

Soluție de 53

2.4 Soluția de 36 de puncte

Dacă t ar apărea ca subșir în s, atunci ar trebui ca $m \le n$, și cum $n \le m + 10$, se pot scoate doar intervale cu lungimea lor mai mică sau egală cu 10. Vom proceda la fel ca și la subtask-ul precedent, dar ne vom duce cu r-ul până la $\min(n, l + 10)$.
Soluție de 36

2.5 Soluția de 73 de puncte

Dacă am combina ultimele două subtask-uri, cu un $if(n \le m + 10)$, vom obține o soluție de 73 de puncte.
Soluție de 73

2.6 Soluția oficială

Observăm faptul că, dacă ne-am nota
$lmin_i$ = poziția minimă poz astfel încât $pozl_{poz} = i$, $0 \le i \le m$.
$rmax_i$ = poziția maximă poz astfel încât $pozr_{poz} = i$, $1 \le i \le m + 1$.
Atunci intervalul $[lmin_i + 1, rmax_{i+1} - 1]$ nu poate fi extins, deoarece, dacă ar fi extins, am scoate ori caracterul t_i, ori caracterul t_{i+1}. Așadar, trebuie doar să verificăm intervalele $[lmin_i + 1, rmax_{i+1} - 1]$, pentru $0 \le i \le m$.
Notă: Soluții care optimizează soluția de 53 de puncte cu căutare binară sau two pointers intră în timp, pentru a obține o soluție de 100 de puncte.
Soluție de 100

3 Matrix

AUTOR: ARDELEAN RAUL, LUCA MUREȘAN

3.1 Soluția oficială

Fie $A_1 \leq A_2 \leq \ldots \leq A_N$. Ne vom folosi de faptul că $\gcd(A_i, A_j) \leq A_{min(i,j)}$.
Este adevărat că $\gcd(A_N, A_N) = A_N \geq A_i \geq gcd(A_i, A_j)$ pentru fiecare $1 \leq i, j \leq N$. Asta înseamnă că A_N este egal cu elementul maxim din tabel.
Să setăm A_N elementul maxim din tabel și să îl ștergem din setul de elemente din tabel. Am șters $\gcd(A_N, A_N)$, deci setul conține acum toate $\gcd(A_i, A_j)$, pentru fiecare $1 \leq i, j \leq N$ și $1 \leq min(i, j) \leq N - 1$.
Din ultimele două inegalități:
$gcd(A_i, A_j) \leq A_{min(i,j)} \leq A_{N-1} = gcd(A_{N-1}, A_{N-1})$.
De îndată ce setul conține $gcd(A_{N-1}, A_{N-1})$, elementul maxim din setul de elemente curent este egal cu A_{N-1}. În măsura în care știm deja A_N, eliminăm $gcd(A_{N-1}, A_{N-1}), gcd(A_{N-1}, A_N), gcd(A_N, A_{N-1})$ din setul de elemente.
Acum setul conține toate $gcd(A_i, A_j)$, pentru fiecare $1 \leq i, j \leq N$ și $1 \leq min(i, j) \leq N - 2$.
Repetăm această operație pentru fiecare k de la $N - 2$ la 1, fixăm A_k elementul maxim din set și eliminăm din set
$gcd(A_k, A_k), gcd(A_i, A_k), gcd(A_k, A_i)$ pentru fiecare $k < i \leq N$.
Se poate demonstra corectitudinea acestui algoritm prin inducție

matematică. Pentru efectuarea operațiilor de ștergere și de obținere a elementului maxim se poate utiliza multiset sau map, astfel soluția are complexitatea $\mathcal{O}\left(N \cdot log(N)\right)$

3.1.1 Cod sursă

Soluție de 100