Arbori de intervale

BUZATU GIULIAN & NIȚĂ ALEXANDROS

Problemă

Să rezolvăm problema: https://codeforces.com/edu/course/2/lesson/4/2/practice/contest/273278/problem/B

Problemă

Să rezolvăm problema: https://codeforces.com/edu/course/2/lesson/4/2/practice/contest/273278/problem/B

Hint: Cum putem reduce problema la una rezolvabilă cu un arbore de intervale?

Problemă

Să rezolvăm problema: https://codeforces.com/edu/course/2/lesson/4/2/practice/contest/273278/problem/B

Rezolvare: Putem să folosim un arbore de intervale în care pentru fiecare interval ținem minte suma elementelor acestuia. Atunci, singura funcție care se schimbă este cea de query. Dacă în fiul stâng avem suma (numărul de 1) mai mare decât al k-lea element pe care îl căutăm, vom merge pe intervalul din stânga continuând să căutăm pe k. Dacă suma este mai mică, căutăm pentru (k – suma din fiul stâng) în intervalul drept.

Implementare: https://github.com/Giulian617/Hai-la-olimpiada-2023-2024/blob/main/11-12/resources/EDU-SEGTREE-4-2B.cpp

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/maxq

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/maxq

Hint: Având în vedere că avem atât operații de update, cât și de query, instinctiv ne vom gândi cum putem folosi un AINT pentru a rezolva problema.

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/maxq

Rezolvare: În fiecare nod vom reține: suma pe intervalul pe care îl reprezintă nodul, suma subsecvenței maxime, suma maximă pe prefix și suma maximă pe sufix. Astfel, trebuie doar să avem grijă cum vom combina valorile din fii în nodul părinte.

Implementare: https://infoarena.ro/job_detail/2954397?action=view-source

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/distincte

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/distincte

Hint: Putem procesa query-urile offline.

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/distincte

Rezolvare: Sortăm query-urile după capătul din dreapta, după care pacurgem vectorul inițial. Vom folosi un AINT pentru sumă, iar la fiecare pas vom actualiza poziția curentă cu valoarea 1 în AINT, iar ultima apariție a valorii curente o vom actualiza cu valoarea 0. Totodată, vom reține că ultima apariție a valorii curente devine cea de pe poziția actuală. După aceea vom răspunde query-urilor care au capătul din dreapta la poziția curentă, făcând un query în AINT pe intervalul query-ului respectiv, iar suma de pe acest interval va reprezenta numărul de elemente distincte pentru query-ul dat.

Implementare: https://infoarena.ro/job_detail/2954554?action=view-source

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/mit

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/mit

Hint: Trebuie să facem update-urile până la nivelul frunzelor?

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/mit

Rezolvare: Pentru operațiile de tipul 1, știm cum să folosim un arbore de intervale astfel încât să răspundem eficient. Problema intervine când primim o operație de tipul 2, deoarece ar trebui să facem B-A update-uri pentru a actualiza toate valorile din interval. Pentru a eficientiza acest proces, putem observa că nu este nevoie să facem update până ajungem în frunze, ci putem folosi o tehnică numită Lazy Propagation. Astfel, în momentul în care intervalul dintr-un nod este complet inclus în intervalul pentru update, vom reține în el valoarea cu care dăm update pe acel interval. Așadar, acum putem avea valori de genul acesta în orice nod din arbore. Pentru a fi siguri că răspundem corect la operațiile de tipul 1, înainte să coborâm în fii, va trebui să propagăm valoarea lazy pe care o avem în nodul curent (atât la update, cât și la query).

Implementare: https://infoarena.ro/job_detail/2953992?action=view-source

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/hotel

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/hotel

Hint 1: Pare că un arbore de intervale ar putea fi folositor, dar un asemenea arbore poate să facă schimbări doar asupra unui element din vector. Sau oare?

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/hotel

Hint 1: Pare că un arbore de intervale ar putea fi folositor, dar un asemenea arbore poate să facă schimbări doar asupra unui element din vector. Sau oare?

Hint 2: Am putea reține în fiecare nod lungimea maximă din intervalul pe care îl reprezintă. La fiecate operație de tipul $update$, cum actualizăm această valoare?

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/hotel

Soluție: Conform Hint-ului 2, vom reține în fiecare nod din arborele de intervale lungimea maximă din intervalul pe care îl reprezintă. Cu toate acestea, apar unele probleme. În urma unei operații de tip 2, putem obține doi indici $j \in \left\{left, left+1, \dots, \left[\frac{left+right}{2}\right]\right\}$ și $k \in \left\{\left[\frac{left+right}{2}\right]+1, \left[\frac{left+right}{2}\right], \dots, right\right\}$ pentru care intervalele $\left[j, j+1, \dots, \left[\frac{left+right}{2}\right]\right], \left[\left[\frac{left+right}{2}\right]+1, \left[\frac{left+right}{2}\right]+2, \dots, k\right]$ sunt camere goale și formează cea mai lungă subsecvență de camere goale din intervalul $[left, right]$. Observăm, deci, că trebuie să reținem și lungimile maxime ale secvențelor de camere goale cu capătul stâng $left$ și cu capătul drept $right$, respectiv.

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/hotel

Soluție: Vom folosi o structură în care ținem minte aceste valori. Să le numim $lenStart$, $lenFinish$ și $longest$. Să vedem cum calculăm aceste valori. Lungimea secvenței maxime va fi dată de expresia:

$$max\begin{Bmatrix} aint[2 * node].longest, \\ aint[2 * node + 1].longest, \\ aint[2 * node].lenFinish, aint[2 * node + 1].lenStart \end{Bmatrix},$$

conform raționamentului de pe slide-ul precedent. Vom actualiza $aint[node].lenStart$ dacă este nevoie, i. e. dacă fiul din stânga este o secvență plină de camere nevizitate, caz în care folosim formula de mai jos. Analog pentru $aint[node].lenFinish$.

$aint[node].lenStart = aint[2 * node].lenStart + aint[2 * node + 1].lenStart.$

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/hotel

Soluție: Să rezolvăm, însă cazul de bază. Aceste este de dat de un interval care este complet inclus în intervalul pe care vrem să îl actualizăm. Astfel, vom schimba parametri în funcție de tipul de actualizare. Cu toate aceste, putem întâmpina probleme deoarece toți fiii direcți sau indirecți ai nodului actualizat nu vor fi, la rândul lor, actualizați. Deci, dacă primim o actualizare pe unul dintre aceste noduri, starea lui nu va fi cea corectă. Putem rezolva această problema adăugând un nou parametru structurii nodurilor, care să marcheze dacă a fost modificat anterior. Dacă a fost, vom trece această stare de „modificat” fiilor săi și îl vom marca ca nemodificat.

Implementare: https://infoarena.ro/job_detail/3223303?action=view-source

Temă

- https://infoarena.ro/problema/schi
- https://codeforces.com/problemset/problem/474/F
- https://codeforces.com/edu/course/2/lesson/5/2/practice/contest/279653/problem/D
- https://codeforces.com/edu/course/2/lesson/5/2/practice/contest/279653/problem/E

Probleme suplimentare

- https://codeforces.com/contest/1743/problem/F
- https://infoarena.ro/problema/zoo

Lectură suplimentară

- https://codeforces.com/edu/course/2/lesson/5
- https://codeforces.com/edu/course/2/lesson/5/2
- https://codeforces.com/edu/course/2/lesson/5/3
- https://codeforces.com/edu/course/2/lesson/5/4
- https://cp-algorithms.com/data_structures/segment_tree.html
- https://infoarena.ro/arbori-de-intervale
- https://sepi.ro/assets/upload-file/oni2024/OJI/11-12/11-12_editorial/Editorial_11_12_OJI_2024.pdf