Olimpiada Națională de Informatică, Etapa Națională, Clasa a V-a Descrierea soluțiilor

Comisia științifică

15 aprilie 2025

Problema 1. Cartonase

Propusă de: prof. Marius Nicoli, Colegiul Național "Frații Buzești", Syncro Soft, Craiova

Cerința 1

Pentru cerința 1 este suficient să parcurgem elementele vectorului care preced poziția poz dată și să ne oprim la prima valoare care este mai mare decât $v[poz]$, afișând poziția respectivă. Dacă nu sunt astfel de valori înaintea poziției poz, atunci soluția va fi poz.

Cerința 2

Pentru a rezolva cerința 2, traversăm vectorul element cu element și la poziția curentă decidem dacă o afișăm sau nu.

O primă abordare este să parcurgem iarăși elementele de la început până la poziția curentă și să verificăm dacă toate sunt mai mici sau egale cu valoarea poziției curente. Timpul de calcul obținut este de ordinul $O(N^2)$ și această soluție nu se va încadra în timp pentru toate datele de test.

Ținând însă cont că toate elementele vectorului sunt distincte și au valori de la 1 la N, observăm că poziția curentă i este una care trebuie afișată dacă și numai dacă maximul din vector dintre elementele aflate până la poziția i este egal cu i (valoarea poziției curente).

Astfel, scriem un algoritm de calcul al maximului dintr-un vector, iar la poziția curentă i este suficient să-l afișăm pe i dacă și numai dacă maximul de până acum este i. Această abordare are timp de calcul de ordin $O(N)$.

Cerința 3

Soluția optimă pentru cerința 3 este asemănătoare cu cea descrisă pentru cerința 2. De data aceasta însă, la poziția curentă i este necesar să păstrăm atât maximul cât și al doilea maxim.

Observăm că putem decide să afișăm poziția i dacă primul maxim este strict mai mare decât i și al doilea maxim este mai mic sau egal cu i. Avem, de asemenea, timp de calcul de ordin $O(N)$.

Problema 2. Casute

Propusă de: Dan-Constantin Spătărel, București

Cazurile 1 și 2 (50 de puncte)

Dacă $N \leq 3000$ atunci putem folosi un vector de dimensiune $N + 1$ (deoarece vectorii sunt indexați de la 0, nu de la 1 ca în problemă) pentru a ține evidența numerelor naturale aflate în fiecare căsuță pe parcursul executării celor Q operații.

Cu ajutorul unui vector putem rezolva fiecare dintre cele 3 operații relativ ușor, astfel:

1. Operațiile de primul tip se rezolvă cu o atribuire în cadrul unei structuri repetitive de tip for.
2. Răspunsul pentru de al doilea tip de operație se obține accesând elementul de pe pozția poz din vector.
3. Răspunsul pentru de al treilea tip de operație se obține prin determinarea maximului și prin contorizarea numărului de apariții al acestuia în vector, în intervalul st inclusiv dr exclusiv.

Complexitatea spațiu: $O(N)$
Complexitatea timp: $O(QN)$

Cazurile 1 și 3 (50 de puncte)

Avem de răspuns numai la operații de tip 2.

Observăm că rezultatul unei operații de tip 2 depinde doar de ultima operație de tip 1 care a afectat căsuța poz înainte de executarea operației de tip 2.

Deoarece $Q <= 3000$ putem stoca toate operațiile, în ordinea în care trebuie executate (de exemplu cu ajutorul a 5 vectori cu denumirile: tip, st, dr, nr și poz).

Putem găsi răspunsul pentru fiecare operație de tip 2 astfel: căutăm operația anterioară, cea mai recentă, de tip 1, cu proprietatea: $st_1 \leq poz_2 < dr_1$ - răspunsul este nr_1. Dacă nu există nicio astfel de operație atunci răspunsul este 0.

Dacă observăm că există o echivalență între procesul de inițializare a căsuțelor cu valoarea 0 și operația fictivă $tip = 1$ $st = 1$ $dr = N + 1$ $nr = 0$, atunci o alternativă, pentru a evita cazul particular de mai sus, este să adăugăm, înainte de citirea operațiilor din fișierul de intrare, această operație.

Complexitatea spațiu: $O(Q)$
Complexitatea timp: $O(Q^2)$

Cazurile 1, 2 și 3 (75 de puncte)

Putem combina cele două soluții de mai sus, într-o singură sursă, astfel: Dacă $N \leq 3000$ atunci rezolvăm problema folosind prima soluție, altfel rezolvăm problema folosind a doua soluție.

Toate cazurile (100 de puncte)

Dacă $N = 10^9$, atunci orice fel de soluție care va încerca să rețină valorile din fiecare căsuță va obține verdictul limită de memorie depășită. Se impune astfel să economisim memoria utilizată.

Putem observa că toate cele N căsuțe pot fi împărțite în intervale maximale (care nu mai pot fi extinse) de căsuțe consecutive cu proprietatea că orice operație de tip 1:

- fie nu modifică niciuna dintre căsuțele din interval;
- fie modifică toate căsuțele din interval.

De aceea, dacă am identifica aceste intervale, am putea reține pentru fiecare dintre ele un singur număr natural: valoarea din fiecare dintre căsuțele din interval.

Să analizăm următorul exemplu: $N = 50$ și 4 operații de tipul 1 (inclusiv operația suplimentară echivalentă cu inițializarea):

Intervale	1...5	6...16	17...24	25...31	32...35	36...41	42...50	
index	1	6	17	25	32	36	42	51
tip=1 st=**1** dr=**51** nr=0	0	0	0	0	0	0	0	
tip=1 st=**6** dr=**42** nr=6	0	6	6	6	6	6	0	
tip=1 st=**25** dr=**36** nr=4	0	6	6	4	4	6	0	
tip=1 st=**17** dr=**32** nr=5	0	6	5	5	4	6	0	

Intervalele, vectorul index și valorile după fiecare dintre cele 4 operații

Observăm că toate intervalele pot fi descrise cu ajutorul vectorului *index* (descris în tabelul de mai sus) prin două elemente consecutive ale sale. Mai mult dect atât, observăm că elementele vectorului *index* sunt toate valorile *st* și *dr* de la toate operațiile de tip 1, sortate crescător.

În acest moment putem rezolva ușor operațiile de tip 2, identificând din ce interval face parte *poz*.

Operațiile de tip 3 sunt mai dificil de rezolvat, deoarece trebuie să identificăm atât intervalele care contribuie la rezultatul unei operații cât și lungimea intersecției dintre aceste intervale și intervalul operației.

O modalitate facilă de a simplifica rezolvarea, atât petru operațiile de tip 3 cât și pentru cele de tip 2 este să adăugăm în plus la vectorul index atât valorile *st* și *dr* de la toate operațiile de tip 3 cât și valorile *poz* de la toate operațiile de tip 2. Acest proces va avea ca efect fragmentarea suplimentară a intervalelor, astfel încât ele își vor pierde proprietatea de maximalitate însă cu următoarele beneficii:

- valorile *poz* ale operațiilor de tip 2 se vor afla numai la începutul unui interval;
- intervalele formate sunt complet incluse în intervalul *st* inclusiv *dr* exclusiv al oricărei operații de tip 3.

Algoritm

Folosind tehnica descrisă în a doua soluție, vom stoca toate operațiile, în ordinea în care trebuie executate.

Într-un vector numit *index* vom pune laolaltă toate valorile *st*, *dr* și *poz* de la toate operațiile, inclusiv valorile speciale 1 și $N + 1$ corepunzătoare operației fictive $tip = 1$ $st = 1$ $dr = N + 1$ $nr = 0$, echivalentă cu procesul de inițializare a căsuțelor.

Vom sorta elementele vectorului *index* și apoi vom elimina dublurile (elementele care se repetă).

Similar cu prima soluție, vom folosi un vector de aceeași lungime ca și vectorul *index* pentru a stoca valorile celulelor din fiecare interval în parte.

Operațiile de primul tip se rezolvă astfel:

- cu ajutorul unei structuri repetitive se caută poziția elementului *st* în vectorul *index*;
- cu ajutorul unei structuri repetitive se caută poziția elementului *dr* în vectorul *index*;
- în cadrul celei de-a doua structuri revetitive se modifică vectorul de valori.

Răspunsul pentru de al doilea tip de operație se obține astfel:

- cu ajutorul unei structuri repetitive se caută poziția elementului *poz* în vectorul *index*;
- rezultatul operației se regăsește în vectorul de valori la poziția găsită la pasul anterior.

Răspunsul pentru de al treilea tip de operație se obține astfel:

- cu ajutorul unei structuri repetitive se caută poziția elementului *st* în vectorul index;
- cu ajutorul unei structuri repetitive se caută poziția elementului *dr* în vectorul index;
- în cadrul celei de-a doua structuri revetitive se calculează valoarea maximă din vectorul de valori;
- atunci când actualizăm maximul, resetăm numărul său de apariții la lungimea intervalului curent;
- de fiecare dată când găsim o nouă apariție maximului, creștem numărul său de apariții cu lungimea intervalului curent.

Operația de eliminare a dublurilor din vectorul index poate fi opțională. În funcție de detaliile de implementare, unele implementări pot lua punctaj maxim deși nu elimină dublurile.

Complexitatea spațiu: $O(Q)$

Complexitatea timp: $O(Q^2)$

Problema 3. Perechi

Propusă de: stud. Jonathan Mogovan, Universitatea "Babeș-Bolyai", Cluj-Napoca, Cluj

Cerința 1

Pentru cerința 1 este suficient să găsim cel mai din dreapta element care respectă condiția. Parcurgem șirul și identificăm cel mai mic număr Y pentru care suma $X + Y$ este divizibilă cu K, alegând poziția celui mai din dreapta Y. Complexitate: $O(N)$.

Cerința 2

Pentru cerința 2, o abordare naivă verifică toate perechile posibile în două bucle și marchează valorile din perechile valide pentru a nu le reutiliza. Complexitatea de timp este: $O(N^2)$.

Soluția optimă împarte numerele în funcție de resturile lor la K cu ajutorul unui vector de frecvență și le grupează în perechi de forma $(R, K - R)$, unde $1 \leq R \leq K/2$. Se analizează separat cazurile pentru restul 0 și pentru K par. Pentru fiecare pereche de forma de mai sus, adunăm la rezultat diferența dintre valoarea frecvenței mai mari și valoarea frecvenței mai mici.
Complexitate: $O(N + K)$.

Echipa

Problemele pentru această etapă au fost pregătite de:

- Pintea Adrian-Doru, Inspectoratul Școlar Județean, Cluj-Napoca, Cluj, Inspector
- Beiland Arnold, Liceul Teoretic, Carei, Satu Mare, Profesor
- Dumitrașcu Dan Octavian, Colegiul Național "Dinicu Golescu", Câmpulung, Argeș, Profesor
- Iordaiche Eugenia Cristiana, Liceul Teoretic "Grigore Moisil", Timișoara, Timiș, Profesor
- Nicoli Marius, Colegiul Național "Frații Buzești", Craiova, Dolj, Profesor
- Șandor Nicoleta Lenuța, Colegiul Național "Mihai Eminescu", Satu Mare, Satu Mare, Profesor
- Tîmplaru Roxana Gabriela, Colegiul "Ștefan Odobleja", Craiova, Dolj, Profesor
- Spătărel Dan-Constantin, SC SPĂTĂREL TUTORING SRL, București, București, Programator/specialist în domeniu
- Mogovan Jonathan, Universitatea "Babeș-Bolyai", Cluj-Napoca, Cluj, Student