DESCRIEREA SOLUȚIILOR, OLIMPIADA NAȚIONALĂ DE INFORMATICĂ, CLASA A VI-A

Problema Iluminat

Propusă de: prof. Prof. Violeta Grecea – Colegiul Național de Informatică ,,Matei Basarab" Râmnicu Vâlcea

Cerințele 1 și 2. Rezolvarea acestor două cerințe presupune simularea stingerii becurilor: pe linia și coloana pe care se află numărul maxim se sting toate becurile.

Soluție brută. Constă în calcularea efectivă a maximului prin parcurgerea tuturor elementelor din matrice și apoi, parcurgerea liniei și a coloanei pe care se află acesta și atribuirea valorii 0 pentru toate elementelor întâlnite pe această linie și această coloană. Procedeul se repetă pentru fiecare etapă, până la etapa k de stingere a becurilor.

Apoi:

- Pentru cerința 1 se afișează maximul găsit la etapa k.
- Pentru cerința 2 se calculează suma elementelor de pe linia și coloana maximului, prin parcurgerea liniară a acestora. Maximul nu trebuie adăugat de 2 ori (o dată la parcugerea liniei și a doua oară la parcurgerea coloanei).

Această soluție nu va lua punctaj maxim, deoarece nu se încadrează în timp (complexitate temporală $\mathcal{O}(k \cdot n^2)$, punctajul maxim posibil fiind de 36 de puncte pentru ambele cerințe.

Soluție optimă. Presupune folosirea unor vectori astfel:

- doi vectori, *xLinie* și *xColoană*, fiecare având n^2 elemente, în care $xLinie_i$ = linia în care se află valoarea i în tablou, iar $xColoană_i$ = coloana în care se află valoarea i în tablou. Odată cu citirea datelor de intrare se completează și valorile corespunzătoare acestor doi vectori.
- doi vectori linie și coloana, cu maximum n elemente fiecare, în care:
  - $linie_i = 0$ dacă becurile de pe linia i nu au fost stinse (maximul nu s-a găsit încă pe linia i)
  - $linie_i = 1$ dacă becurile de pe linia i au fost stinse (maximul s-a găsit pe linia i)

  Același rol îl are vectorul coloană, referitor la coloanele pe care se găsește maximul.

Simularea stingerii becurilor presupune parcurgerea simultană, în ordinea descrescătoare a indicilor, a vectorilor *xLinie* și *xColoană*, până când se găsește o valoare i pentru care $linie_i = 0$ și $coloană[i] = 0$. Acesta este, pe rând maximul căutat la o etapă. Procedeul se repetă pentru fiecare etapă, până la etapa k de stingere a becurilor. Apoi:

- Pentru cerința 1 se afișează maximul maximul găsit la etapa k.
- Pentru cerința 2 se calculează suma elementelor de pe linia și coloana maximului, prin verificarea condiției ca linia și coloana corespunzătoare elementului să fie egale cu 0 (becurile nu sunt încă stinse).

Această abordare va lua punctaj maxim la aceste două cerințe. complexitatea temporală este $\mathcal{O}(n^2 + n \cdot k)$.

Cerința 3. Presupune calcule care să pornească de la tabloul inițial.

Soluție brută. Constă în parcurgerea fiecărei submatrice cu k linii și k coloane și calcularea sumei elementelor sale.

Soluție optimă. Constă în precalcularea unor sume parțiale, utilizând formula pentru sume partiale 2*D*:

$$cartier_{i,j} = cartier_{i,j-1} + cartier_{i-1,j} - cartier_{i-1,j-1} + cartier_{i,j}$$

Pe baza acestor sume parțiale se calculează maximul dintr-o submatrice de dimensiune *kxk* a tabloului. Această abordare a cerinței, de complexitate $\mathcal{O}(n^2)$, va lua punctaj maxim.

Problema Inundatie

Propusă de: prof. Cristian Frâncu – Nerdvana

Obervații. Următoarele observații sunt importante pentru toate cerințele:

- Apa va acoperi coloanele în ordinea de la stânga la dreapta. Este ordinea în care vârfurile coloanelor sunt atinse de apă;
- Odată ce apa atinge vârful unei coloane, deci acoperă acea coloană, nu este obligatoriu ca apa să se acumuleze deasupra acelei coloane. Dacă are unde se duce spre dreapta, înălțimile coloanelor fiind mai mici sau egale cu coloana curentă, apa se va deplasa.

Cerința 1. Înălțimea maximă a unei coloane de apă când apa ajunge peste tot la nivelul coloanei maxime.

La acest moment nivelul apei este uniform și are exact înălțimea celei mai înalte coloane. Deci cea mai înaltă coloană de apă va fi deasupra înălțimii minime. Răspunsul este $hmin - h_{min}$, diferența dintre înălțimea maximă și cea minimă de la intrare.

Complexitate timp: $\mathcal{O}(N)$, complexitate memorie: $\mathcal{O}(N)$

Cerința 2. timpul în care apa ajunge la înălțimea de pe poziția *P*

Presupunem că apa tocmai a ajuns la coloana *P*. În acel moment există apă acumulată în diverse coloane spre stânga. Dacă însumăm numărul de pătrătele al fiecărei coloane de apă vom obține timpul necesar. Așadar pornim de la coloana *P*, de înălțime *H*, deplasându-ne către stânga. Deoarece apa a ajuns la coloana *P* înseamnă că spre stânga apa are cel puțin acea înălțime. Avem, deci, două posibilități:

(1) Coloana din stânga, h_i, este mai mică sau egală cu cea curentă. În acest caz nivelul apei este la înălțime *H*, iar coloana de apă are înălțime $H–h_i$. Vom însuma această înălțime la totalul de timp necesar.
(2) Coloana din stânga, h_i, este mai mare decât cea curentă. În acest caz vom actualiza *H* la acea înălțime h_i.

Continuăm, astfel, deplasarea către stânga însumând coloanele de apă și actualizând înălțimea *H*. Suma acestor coloane este timpul minim în care apa ajunge la înălțimea *P*.

Complexitate timp: $\mathcal{O}(N)$, complexitate memorie: $\mathcal{O}(N)$

Cerința 3. numărul înălțimii la care ajunge apa după *S* secunde

Soluție 1: căutare binară. O soluție este să folosim rezolvarea cerinței 1: vom căuta binar coloana la care ajunge apa. Vom porni cu un interval de căutare $[1\ N]$ și la fiecare pas ne vom întreba, pentru coloana de la jumatea acelui interval, în cât timp ajunge apa la acea coloană, folosind algoritmul de la cerința 2. Dacă timpul depășește *S* vom deplasa limita din dreapta a intervalului, în caz contrar pe cea din stânga.

Complexitate timp: $\mathcal{O}(N \cdot \log_2 N)$, complexitate memorie: $\mathcal{O}(N)$

Soluție 2. : parcurgere de la stânga la dreapta Pornim de la coloana 1, de înălțime *H*. Avem două variante:

(1) Coloana din dreapta, h_i, este mai mică sau egală cu cea curentă, *H*. În acest caz nu consumăm apă pentru a ne deplasa spre dreapta, deci nu avem nimic de făcut.

(2) Coloana din dreapta, h_i, este mai mare decât cea curentă, H. În acest caz apa se acumulează. Ne *înălțăm* cu o unitate față de H și ne deplasăm către stânga până găsim prima coloană la această înălțime (sau până depășim prima coloană). La fiecare deplasare adunăm unu la necesarul de apă. Dacă nu am depășit S, ne înălțăm încă o unitate față de H și repetăm procedeul de deplasare la stânga. Ne vom înălța până ce ajungem la înălțimea h_i, sau până ce depășim *bugetul* de S unități de apă. Dacă ajungem la înălțimea h_i, vom relua deplasarea către stânga.

Ultima coloană la care am reușit să ajungem este răspunsul la cerință.
Complexitate timp: $\mathcal{O}(N)$, complexitate memorie: $\mathcal{O}(N)$

Cerința 4. numărul celei mai din stânga înălțimi pe care o vom reduce cu o unitate pentru ca apa să ajungă cât mai repede la coloana cu numărul P.

Mergând de la P către stânga vom număra distanțele dintre înălțimi despărțite de apă. Distanța maximă, D, este cea unde vom reduce coloana din dreapta, deoarece reducând acea coloană cu unu timpul se scurtează cu D. Procedăm astfel:

- Pornim de la coloana numărul P de înălțime H, parcurgând înălțimile către stânga. Ne oprim la prima înălțime h_i mai mare sau egală cu H și reținem distanța dintre cele două înălțimi, $D = P–i+1$, precum și poziția înălțimii din dreapta acelei distanțe, P. Distanța D este chiar diferența de timp cu care se scurtează timpul în care apa ajunge la coloana P, dacă reducem coloana P cu unu.
- Actualizăm înălțimea curentă H ca fiind h_i și P ca fiind i și continuăm deplasarea către următoarea înălțime mai mare sau egală cu H, obținând o nouă distanță D. La fiecare pas reținem D dacă este mai mare sau egal cu cel de până acum, împreună cu numărul coloanei din dreapta distanței D, adică P. Este necesar mai mare sau egal deoarece dorim cea mai din stânga coloană pe care o putem reduce.

Trebuie să dăm, însă, atenție specială cazului când avem mai multe coloane de aceeași înălțime despărțite de înălțimi mai mici, de exemplu cazul $6, 3, 3, 3, 3, 3, 6, 3, 3, 6$. Reducând cel mai din dreapta 6 la 5 vom reduce timpul cu 2 secunde, distanța până la următorul 6. Dar dacă reducem următorul 6 la 5 nu vom câștiga nimic, de fapt vom pierde o secundă.

Pentru a trata acest caz, după fiecare deplasare și calcul de distanță D, dacă h_i are aceeași înălțime cu H ne vom deplasa la stânga suplimentar până ce găsim o coloană strict mai înaltă decât H.

Complexitate timp: $\mathcal{O}(N)$, complexitate memorie: $\mathcal{O}(N)$

Problema Siruri

Propusă de: prof. Prof. Flavius Boian – Colegiul Național Spiru Haret" Târgu-Jiu

Cerința 1. O soluție eficientă este marcarea într-un vector de frecvență a cifrelor existente pentru fiecare număr și oprirea în cazul în care o cifră este deja marcată.

Cerința 2. Se verifică dacă ultima cifră a numărului curent este egală cu prima cifră a numărului următor. Dacă acest lucru este adevărat se lipesc cele două numere, se elimină cifrele care apar de mai multe ori, păstrându-se doar prima apariție și se repetă procedeul atât timp cât este posibil.

Verificarea dacă numărul nou format se va uni cu următorul se face după ce în număr cifrele rămân o singură dată. Dacă un număr nu are ultima cifră egală cu a următorului, acesta se transformă prin ștergerea cifrelor care apar de mai multe ori și păstrarea celei mai din stânga apariții a acestora.

Cerința 3. Problema se reduce la a determina, în șirul nou format, câte numere cu număr maxim de cifre există. Se parcurge șirul, se reține numărul cu număr maxim de cifre și numărul de apariții al unui astfel de număr. Rezolvarea cerinței se poate face printr-o singură parcurgere a șirului. Complexitate timp: $\mathcal{O}(N * L_{max})$, complexitate memorie: $\mathcal{O}(N)$, unde L_{max} reprezintă numărul maxim de cifre ale unui număr.

Echipa

Problemele pentru această etapă au fost pregătite de:

- prof. Dan Pracsiu – Liceul Teoretic ,,Emil Racoviță"
- prof. Alina Pintescu – Colegiul Național ,,Gheorghe Șincai"
- prof. Violeta Grecea – Colegiul Național de Informatică ,,Matei Basarab"
- prof. Cerasela Daniela Cardaș – Colegiul Național ,,A.T. Laurian"
- prof. Georgeta Balacea – Colegiul Național Vasile Alecsandri
- prof. Dorin-Mircea Rotar – Colegiul Național ,,Samuil Vulcan"
- prof. Flavius Boian – Colegiul Național ,,Spiru Haret"
- prof. Costineanu Raluca – Colegiul Național ,,Ștefan cel Mare"
- stud. Liviu Armand Gheorghe – Universitatea București, Facultatea de Matematică și Informatică
- stud. Ioan Cristian Pop – Universtiatea Politehnica București
- prof. Rodica Pintea – Liceul Teoretic ,,Radu Vlădescu"
- stud. Theodor Gabriel Tulbă Lecu – Universitatea Politehnica Bucuresti
- Cristian Frâncu – Nerdvana