

## **MAJSTOR**

Broj bodova koje je Sven osvojio dobivamo izvedbom algoritma za bodovanje opisanog u tekstu zadatka. Za svaku rundu uspoređujemo Svenov simbol sa svim ostalim simbolima i dodajemo odgovarajući broj bodova u sumu.

Broj bodova koje je Sven mogao osvojiti dobivamo tako da za svaku rundu isprobamo koji od tri simbola donosi najviše bodova i taj broj bodova dodamo u sumu.

To je najlakše implementirati tako da napravimo funkciju bodovi koja za parametre prima redni broj runde i znak koji je Sven pokazao i vraća broj osvojenih bodova u toj rundi. Ako su A i B traženi brojevi onda ih možemo izračunati sljedećim algoritmom:

Za rundu r od 1 do R radi

$A = A + \text{bodovi}(r, \text{Sven}[r])$

$B = B + \max\{\text{bodovi}(r, 'S'), \text{bodovi}(r, 'P'), \text{bodovi}(r, 'K')\}$

## **NIZOVI**

Neka je smušenost(P, K) smušenost nizova koje dobijemo izbacivanjem prvih P i zadnjih K brojeva iz oba niza. Rješenje koje za svaki par (P, K) računa smušenost(P, K) postupkom opisanim u tekstu zadatka je složenosti  $O(N^3)$  i presporo je za velike N, te dobiva 30 bodova.

Potrebno je primijetiti kako da se smušenost(P, K) u općenitom slučaju može izraziti kao:

$$\text{smušenost}(P, K) = A(P) \cdot B(N-K) + B(P) \cdot A(N-K) + \text{smušenost}(P+1, K+1)$$

Drugim riječima, smušenost možemo izračunavati tako da krenemo iz sredine nizova i širimo interval (L, R) prema van s obje strane. Smušenost proširenog intervala (L-1, R+1) dobivamo kao smušenost neproširenog intervala uvećanog za umnoške na granicama:  $A(L-1) \cdot B(R+1) + B(L-1) \cdot A(R+1)$ .

Složenost ovog algoritma je  $O(N^2)$  jer sredinu intervala možemo odabrati na  $2 \cdot N - 1$  način, a svaki interval najviše  $N/2$  puta proširiti za 1 na obje strane.

## **TABLICA**

Direktna implementacija algoritma opisanog u tekstu zadatka je vremenske složenosti  $O(K \cdot N^2)$  i prostorne složenosti  $O(N^2)$  i donosi 30 bodova.

Algoritam možemo ubrzati tako da prvo pročitamo sva premještanja i, za svaki broj koji trebamo u nekom trenutku premjestiti, pamtimo trenutnu poziciju i ciljnu poziciju.

Radimo jedno po jedno premještanje tako da prvo iz trenutne i ciljne pozicije broja koji premještamo izračunamo broj potrebnih rotacija redaka i stupaca, te ispišemo njihov zbroj.

Zatim svaki broj koji se nalazi u istom retku kao i broj koji premještamo pomaknemo odgovarajući broj polja u desno, a nakon toga svaki broj koji se nalazi u istom stupcu kao i broj koji premještamo pomaknemo odgovarajući broj polja prema dolje.

Vremenska složenost ovog algoritma je  $O(K^2)$ , a prostorna složenost  $O(K)$ .

## **CVJETIĆI**

Simulacija izrastanja biljaka crtanjem po dvodimenzionalnoj tablici je vremenske i prostorne složenosti  $O(N^2 + N \cdot M)$ , gdje je  $M$  ograničenje na koordinate, te donosi 30 bodova.

Neka je  $A(x)$  ukupan broj biljaka kojima na horizontalnom segmentu na koordinati  $x$  može, ali još nije, izrastao cvjetić. Niz  $A$  početno sadrži sve nule, a mijenja se izrastanjem biljke  $(L, R)$  na sljedeći način:

- Na koordinatama  $L$  i  $R$  narasti će cvjetići i to, po definiciji niza  $A$ , njih točno  $A(L) + A(R)$ , pa taj broj ispisujemo na izlaz. Nakon izrastanja tih cvjetića više nema biljaka kojima može izrasti cvjetić na tim koordinatama, pa postavljamo  $A(L) = 0$  i  $A(R) = 0$ .
- Na koordinatama  $[L+1, R-1]$  horizontalni segment nove biljke još nema cvjetiće, pa za svaku koordinatu  $x$  iz tog intervala postavljamo  $A(x) = A(x) + 1$ .

Direktna implementacija gornjeg algoritma je složenosti  $O(N \cdot M)$  gdje je  $M$  ograničenje na koordinate, te donosi 45 bodova.

Za sve bodove potrebno je implementirati neku strukturu podataka koja efikasno izvodi sljedeće operacije na nizu  $A$ :

- Postavi  $A(x) = 0$
- Postavi  $A(x) = A(x+1)$  za sve  $x$  iz  $[L-1, R+1]$

Neke od struktura koje možemo iskoristiti su: stablo intervala ( $O(\log N)$  po operaciji), logaritamska struktura na nizu diferencija od  $A$  ( $O(\log N)$  po operaciji), te podjela na blokove ( $O(\sqrt{N})$  po operaciji).

U priloženim izvodnim kôdovima implementirano je rješenje koje dijeli niz  $A$  na blokove veličine 256. Vidi tekst zadatka JAGODA (5. kolo HONI 2009.) za detalje o strukturi.