Laborator Algoritmi și Structuri de Date
Tema 5

Tema săptămânii 5.

1 Probleme cu liste

(2p) 1. Să se construiască o listă (prin una din procedurile de inserat elemente în listă). După ce s-au citit toate elementele, să se parcurgă lista și să se insereze media aritmetică între elementele de pe fiecare două poziții consecutive.

Exemplu: Pentru lista 1 2 3 4 se vor insera mediile. Apelarea funcției standard de afișare a listei inițiale produce acum 1 1.5 2 2.5 3 3.5 4, alternativ 1 1 2 2 3 3 4 dacă lista e pe întregi.

(3p) 2. Să se construiască o listă sortată folosind inserția unui nou element pe poziția lui în lista sortată. Lista se menține sortată prin procedura de inserat care verifică cazurile particulare:

(a) când elementul de inserat este mai mic decât primul element și atunci se inserează la început.

(b) când trebuie căutată poziția unde trebuie inserat noul element (înainte de prima valoare mai mare -sau egală!- decât el)

(c) când am terminat de parcurs lista și atunci trebuie inserat la final

Exemplu:

inserare 4	4
inserare 2	2 4
inserare 5	2 4 5
inserare 4	2 4 4 5
inserare 4	2 4 4 4 5
inserare 1	1 2 4 4 4 5

(1+2p) 3. Să se construiască o listă (prin una din procedurile de inserat elemente în listă). După ce s-au citit toate elementele, să se parcurgă lista și să se inverseze ordinea elementelor în listă.

Exemplu: lista 1 2 3 4 devine lista 4 3 2 1.

Fără bonus: se inserează elementele într-o altă listă.

Cu bonus: se lucrează cu elementele listei citite și se inversează legăturile. În acest mod nu este nevoie de spațiu suplimentar de tip $O(n)$ ci doar spațiu constant $O(1)$ pentru extra variabile.

O variantă de implementare presupune parcurgerea listei cu trei pointeri `prev` și `current` și `next`: se leagă `curent` de `prev` pentru inversarea legăturii dintre ei și apoi toți pointerii trebuie să treacă un pas mai departe astfel:

(a) `prev` trece mai departe folosind `curent`

(b) `curent` trece mai departe folosind `next` salvat anterior (pentru că `curent->next` indică acum `prev`)

(c) `next` trece mai departe folosind `next->next`

(3+2p) 4. Să se interclaseze două liste sortate citite și stocate în prealabil. Interclasarea listelor este operația prin care din două liste sortate obținem o altă listă sortată care conține elementelor ambelor liste inițiale. Dacă cele două liste sunt de lungime n, respectiv m, se impune următoarea restricție: timpul de rulare pentru interclasare va fi $O(n + m)$.

Atenție că pentru inserarea fiecărui element din prima listă în lista a doua (atunci când o luăm mereu de la început în lista a doua) timpul de rulare este: de n ori se fac maximum m pași pentru a localiza poziția unde trebuie inserat elementul $= O(nm)$.

Exemplu: lista 4 6 8 interclasată cu 1 2 5 9 9 produce lista sortată 1 2 4 5 6 8 9 9.

Fără bonus: se parcurg elementele din cele două liste folosind pointeri de tipul `prim1` și `prim2` identificând la fiecare pas, care listă are elementul mai mic pe prima poziție. Acest prim element se inserează (la final!, folosind `ultim3`) într-o a treia listă și se șterge din lista care îl conținea.

Cu bonus: După citirea primelor două liste nu mai e voie să se apeleze `new` pentru alocare de noduri astfel încât să se lucreze doar prin modificarea legăturilor între noduri deja existente (cu efectul că nu vom mai putea ști care erau listele inițiale).

(3p) 5. Să se folosească liste pentru rezolvarea sumei de întregi pozitivi mari (mai mari decât putem reprezenta folosind tipurile de bază). Fiecare listă va conține cifrele unui număr mare, primul element fiind cea mai puțin semnificativă cifră. E permisă afișarea cifrelor în ordine inversă (parcurgerea listei).

Exemplu1: $542 + 34 = 576$ sau $(2 \to 4 \to 5) + (4 \to 3) = (6 \to 7 \to 5)$

Exemplu2: $19 + 11 = 30$ sau $(9 \to 1) + (1 \to 1) = (0 \to 3)$

Exemplu3: $999 + 1 = 1000$ sau $(9 \to 9 \to 9) + (1) = (0 \to 0 \to 0 \to 1)$

În exemplul 3 se ilustrează procesul prin care adunarea cifrei de transport (carry) produce lungirea listei.

(+1p) 6. Să se folosească liste pentru rezolvarea produsului dintre un întreg pozitiv mare (mai mare decât putem reprezenta folosind tipurile de bază) și un întreg pozitiv format dintr-o singură cifră $(0 - 9)$. E permisă afișarea cifrelor în ordine inversă (parcurgerea listei).

(+3p) 7. Să se folosească liste pentru rezolvarea produsului de întregi pozitivi mari (mai mari decât putem reprezenta folosind tipurile de bază). E permisă afișarea cifrelor în ordine inversă (parcurgerea listei).

Exemplu: $234 * 21 = 1914$ sau $(4 \to 3 \to 2) + (1 \to 2)$

$= (4 \to 3 \to 2) + 20 * 234$

$= (4 \to 3 \to 2) + (0 \to 8 \to 6 \to 4)$

$= (4 \to 1 \to 9 \to 4)$

Alternativă de implementare:

$= (4 \to 3 \to 2)$ adunat începând cu a doua poziție cu $(8 \to 6 \to 4)$

$= (4 \to 1 \to 9 \to 4)$

Alternativă de implementare:

$= (0 \to 0 \to 0 \to 0 \to 0)$

$+(4 \to 3 \to 2)$ începând cu poziția 1

$+(8 \to 6 \to 4)$ începând cu poziția 2

$= (4 \to 1 \to 9 \to 4 \to 0)$