# Liste dublu înlănțuite

1) Se consideră o listă dublu înlănțuită, sortată crescător. Să se implementeze un algoritm care inserează un nou element în listă, astfel încât lista să rămână sortată crescător.

Indicații: Algoritmul va parcurge lista, până când elementul următor este mai mare decât valoarea dată, va crea un nou element cu această valoare, astfel încât să indice spre elementul următor, apoi va face elementul curent să indice spre elementul nou. Să se testeze algoritmul pentru o listă cu `10` elemente de tip întreg, în care primul element este `5`, iar fiecare element următor este cu `5` mai mare decât elementul curent. Valoarea elementului care se inserează în listă se citește de la tastatură. La sfârșit, să se afișeze pe ecran lista sortată crescător, cu noul element inserat în poziția corespunzătoare.

2) Se consideră o listă dublu înlănțuită, cu elemente generate aleator, și un element oarecare, care se citește de la tastatură. Să se scrie un algoritm care șterge din listă acest element.

Indicații: Lista va conține un număr de elemente care poate fi schimbat prin program, folosind directiva `#define`, iar valorile elementelor se vor inițializa cu valori aleatoare, între `0` și `100`. După inițializarea cu valori aleatoare, elementele listei se afișează pe ecran. Apoi se citește de la tastatură elementul care va fi șters. Algoritmul caută în listă elementul care va fi șters, îl șterge (dacă îl găsește) și apoi reface legăturile. La sfârșit, se afișează pe ecran lista obținută.

3) Se consideră două liste dublu înlănțuite, cu elemente generate aleator. Să se scrie un algoritm care concatenează listele și elimină elementele duplicate.

Indicații: Listele vor conține un număr de elemente care poate fi schimbat prin program, folosind directiva `#define`, iar valorile elementelor se vor inițializa cu valori aleatoare, între `0` și `20`. După inițializarea cu valori aleatoare, elementele celor două liste se afișează pe ecran. După concatenarea listelor, se parcurge rezultatul, pentru fiecare element verificând numai elementele care îl preced. Dacă există printre acestea un element cu aceeași valoare, se elimină elementul curent, prin refacerea legăturilor, și se continuă procedura. Se poate implementa folosind două bucle

`while`. Prima buclă este `while (ElementCurent)`, unde `ElementCurent = CapListă1->următor`, iar `ElementTemporar = CapListă1`, iar a doua buclă este `while (ElementTemporar != ElementCurent)`. La sfârșit, se afișează pe ecran lista obținută prin concatenare și eliminarea duplicatelor.

4) Se consideră două liste dublu înlănțuite, cu elemente generate aleator. Să se scrie un algoritm care generează intersecția celor două liste.

Indicații: Listele vor conține un număr de elemente care poate fi schimbat prin program, folosind directiva `#define`, iar valorile elementelor se vor inițializa cu valori aleatoare, între `0` și `20`. După inițializarea cu valori aleatoare, elementele celor două liste se afișează pe ecran. Se vor șterge din prima listă toate elementele care nu se află în a doua listă. Elementele se vor compara dupa câmpul valoare. La sfârșit, se afișează pe ecran intersecția celor două liste, care va fi chiar prima listă, care a fost modificată.