Structuri de date
Tema 5

1. Pentru un arbore binar scrieţi o funcţie care determină numărul de noduri frunză dintr un arbore.

2. Scrieţi o funcţie care contorizează numărul de noduri dintr-un arbore binar.

3. Scrieţi o funcţie care determină înălţimea unui arbore binar.

4. Scrieţi o funcţie care tipăreşte un arbore în forma cu paranteze, adică sub forma
   (Key(Left_Tree, Right_Tree))
   unde Left_Tree şi Right_Tree sunt afişările corespunzătoare subarborilor stâng şi drept.

5. Scrieţi o funcţie care va elimina un nod dintr-un arbore binar, folosind următoarea metodă, aplicabilă în cazul când nodul ce urmează să fie şters are ambii subarbori nevizi. Mai întâi, se determină predecesorul imediat al nodului de şters, predecesor raportat la traversarea în inordine, prin parcurgerea la fiul stâng şi apoi cât se poate în dreapta. Acest predecesor imediat are proprietatea că are cel mult un fiu, şi deci poate fi şters din poziţia sa fără dificultate. Apoi va fi instalat în arbore în poziţia ocupată de nodul curent ce trebuia iniţial şters. Demonstraţi că, urmând acest procedeu, se păstrează proprietatea de arbore de căutare.

6. Scrieţi o funcţie care determină lăţimea maximă a unui arbore binar.

7. Scrieţi o funcţie care converteşte un arbore binar într-o listă dublu înlănţuită, în care nodurile sunt puse în ordinea dată de traversarea în inordine a arborelui.

8. Fie x1, x2, …, xn (xi≠xj, i≠j) secvenţa de parcurgere în preordine a unui arbore binar. Arătaţi că, dacă cunoaştem secvenţa de parcurgere în inordine a arborelui binar $x_{\sigma 1}, x_{\sigma 2}, ..., x_{\sigma n}$, atunci putem reconstrui în mod unic arborele binar. Scrieţi o funcţie

   *Btree * BtreeRecover(int x[], int x1[], int n);*

   care implementează acest algoritm.

9. Dându-se secvenţa x1, x2, …, xn (xi≠xj, i≠j) şi $x_{\sigma 1}, x_{\sigma 2}, ..., x_{\sigma n}$ rezultatul traversării în inordine şi postordine, este posibilă refacerea în mod unic a arborelui binar de start? Dacă nu, găsiţi un contraexemplu.

10. Aceeaşi problemă ca la ex. 9, dar cele două secvenţe conţin traversările în preordine şi postordine.

11. (Problema dicţionarului) De la intrare se construieşte un dicţionar prin citirea perechilor de cuvinte:
cuvânt_română cuvânt_engleză
unde cele două cuvinte reprezintă un termen în limba română şi traducerea lui în limba engleză. După terminarea citirii, se vor efectua operaţii de căutare care constau în citirea unui cuvânt în limba română şi afişarea echivalentului în limba engleză. Folosind un arbore de căutare binară, se cere:
a) Scrieţi un program care efectuează aceste operaţii.
b) Implementaţi cazul când se caută folosind cuvântul în limba engleză.

12. (Problema alfabetului) Un cercetător a găsit un dicţionar explicativ al unei limbi necunoscute, care are intrări de forma:
cuvânt1=explicaţie1
cuvânt2=explicaţie2
...
cuvânt n=explicaţie n
Ştiind că şirul format (cuvânt1, cuvânt2,…, cuvânt n) este în ordinea corespunzătoare alfabetului necunoscut, se cere să se determine acest alfabet (ordinea literelor în acest alfabet).

13. (Arborele cu memorie suplimentară) Pentru a traversa arborele binar fără recursivitate şi fără stivă auxiliară, se foloseşte o tehnică specială, ce presupune existenţa unui câmp pointer suplimentar, ce pointează spre nodul părinte al fiecărui nod. Folosind această implementare:
a) scrieţi o metodă care construieşte un asemenea arbore
b) scrieţi o funcţie are traversează arborele recursiv, folosind un indice al legăturii curente.