Structuri de date și algoritmi
- examen scris -

Notă

1. Subiectele se notează astfel: of - 1p; A – 2p; B - 1.5p; C1 - 1p; C2 – 1p; D - 3.5p.
2. Pentru cerința A, justificarea unei complexități presupune deducția acesteia.
3. Pentru cerințele B și C (C1, C2) se cer justificări, care vor fi punctate.
4. Problema de la D se va rezolva în Pseudocod. Se cer și se vor puncta: (1) descrierea ideii de rezolvare și comentarii despre soluția propusă; (2) scrierea reprezentării indicate în enunț; (3) (specificare și) implementare subalgoritm(i); (4) complexitate.
   Nu se acceptă cod C++. Nu se acceptă pseudocod fără comentarii despre soluția propusă.

A. Deduceți timpii mediu si defavorabil pentru următorul subalgoritm. Justificați rezultatul.

```
Subalgoritm h(n, A, B, C) este
|       {pre: n:Intreg; A, B, C:char}
|       daca n≠1 atunci
|       |       h(n-1, A, C, B)  O(n-1)
|       |       scrie n, A, B O(1)
|       |       h(n-1, C, B, A)  O(n-1)
|       | altfel
|       |       scrie 1, A, B  O(1)
|       sfdaca
sfh
```

T(n) = 2T(n-1)+O(1)
T(n-1)=2T(n-2)+O(1)
T(n-2)=2T(n-3)+O(1)

T(n)=2[2T(n-2)+O(1)] + O(1) = 2^2T(n-2) + 2O(1) + O(1)
T(n)=4[2T(n-3)+O(1)]+2O(1)+O(1) = 2^3T(n-3)+2^2O(1)+2O(1)+O(1)

T(n) = 2^k T(n-k) + S2^kO(1)
dar k = n-1 => T(n) = 2^(n-1)T(1)+ S2^(n-2)O(1)
T(n) = 2^(n-1)O(1) + (2^(n-1) - 1)O(1)
T(n) = O(2^n)

=> timpul defavorabil Theta(2^n) (ac cu timpul mediu deoarece alg executa ac nr de pasi indiferent de valoarea lui n)

B. Fie o tabelă de dispersie inițial vidă, cu 5 locații și funcția de dispersie $d(c) = c \bmod 5$, în care coliziunile sunt rezolvate prin înlănțuire, folosind arbori AVL pentru memorarea coliziunilor. Arătati ce se întâmplă la inserarea cheilor 25, 7, 18, 6, 3, 10, 8, 5. Justificati

C. Care este timpul defavorabil pentru sortarea unui vector cu n elemente folosind HeapSort? Justificati

a) $O(\log_2 n)$ b) $O(n)$ c) $O(n \log_2 n)$ d) $O(n^2)$

sortarea prin HeapSort are doua etape: construirea vectorului heap - O(n)
si extragerea maximului in O(log2n). Sunt n elemente de extras si pus in heap deci complexitatea finala este O(n log2n).

C. Presupunem o Stivă implementată secvențial pe un vector v. Ce se întâmplă în cazul în care memorăm vârful stivei la locația v[0], iar primul element introdus în stivă la ultima poziție utilizată în vector? Justificati

a) atât accesul, cât și stergerea se fac în timp liniar b) atât adăugarea, cât și ștergerea se fac în timp liniar

c) stiva nu poate fi folosită pentru a evalua expresii postfixate

accesul se face in timp constant O(1)
adaugarea si stregerea in O(n) pentru ca trebuie modificat vectorul prin mutarea elementelor
(c) nu este corect pentru ca din stiva se poate accesa doar ultimul elem

D. Se consideră un arbore binar conținând în noduri elemente distincte. Se cere să se scrie în Peudocod operația care să determine dacă două noduri *e* si *e'* se află sau nu pe același nivel în arbore. Arborele se reprezintă înlanțuit, cu alocare dinamică a nodurilor. Se va folosi o procedura nerecursivă. Indicați reprezentarea arborelui și precizați complexitatea operației. Folosiți comentarii pentru a ușura înțelegerea soluției. Ex: Pentru arborele de mai jos, *e*= 4, *e'*=40 ⇒ **da**; *e*= 5, *e'*=50 ⇒ **nu**