UNIVERSITATEA DIN BUCUREȘTI
FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ
DOMENIUL DE STUDII INFORMATICĂ – ANUL I

EXAMEN LA DISCIPLINA "PROGRAMAREA ALGORITMILOR"
– model de subiect –

Subiectul I (3 p.)
a) Scrieți o funcție *divizori* care primește un număr variabil de parametri numere naturale și returnează pentru fiecare număr primit ca parametru lista divizorilor săi primi sub forma unui dicționar cu perechi de forma *număr*: *lista divizorilor*. De exemplu, pentru apelul *divizori*(50, 21) funcția trebuie să furnizeze dicționarul {50: [2,5], 21: [3,7]}. **(1.5 p.)**

b) Înlocuiți punctele de suspensie din instrucțiunea *litere_10* = [...] cu o expresie astfel încât lista să fie inițializată cu primele 10 litere mici din alfabetul englez. **(0.5 p.)**

c) Considerăm o funcție recursivă a cărei complexitate este dată de următoarea relație de recurență:

$T(1) = T(2) = 1$
$T(n) = T(n/3) + 2$, pentru $n \geq 1$

Determinați complexitatea funcției respective. (**1 p.**)

Subiectul 2 (3 p.) – metoda Greedy – complexitate maximă $\mathcal{O}(n \log_2 n)$
Considerăm *n* spectacole $S_1, S_2, \ldots, S_n$ pentru care cunoaștem intervalele lor de desfășurare $[s_1, f_1)$, ..., $[s_n, f_n)$, toate dintr-o singură zi. Având la dispoziție o singură sală, în care putem să planificăm un singur spectacol la un moment dat, să se determine numărul maxim de spectacole care pot fi planificate fără suprapuneri. Un spectacol S_j poate fi programat după spectacolul S_i dacă $s_j \geq f_i$. Justificați, pe scurt, corectitudinea programului și complexitatea sa.

Subiectul 3 (3 p.) – metoda programării dinamice – complexitate maximă $\mathcal{O}(n^2)$
Să se determine un subșir crescător de lungime maximă al unui șir *t* format din *n* numere întregi.

Subiectul 4 (3 p.) – metoda backtracking
Să se afișeze toate permutările mulțimii $A = \{1, 2, \ldots, n\}$, unde *n* este un număr natural nenul .