Examen la Analiza Algoritmilor
Set 1

Timp de rezolvare: 90 de minute

1. (1,5p) Construirea unui heap binar se poate face optim în:
   a) $O(n^2)$; b)$O(n \log n)$; c) $O(n)$
2. (1,5p) Dacă găsim o soluție deterministă și polinomială pentru o problemă considerată până acum în NP\P, atunci putem spune că:
   a) $P \subseteq NP$; b) $P=NP$; c) $NP=NPC$
3. (1,5p) Dacă o problemă este în clasa de complexitate NSPACE(n), atunci ea este sigur și în clasa:
   a) SPACE(n**2); b) NSPACE((log n)**2; c) NTIME(n)
4. (1,5p) Fie o problemă A și știm că PCP $\leq_T$ A (PCP = Problema corespondențelor lui Post). Atunci A este sigur:
   a) decidabilă; b) în mulțimea NP-hard; c) NP-completă
5. (1,5p) Orice algoritm de sortare prin comparație de chei are complexitatea:
   a) $O(n^2)$; b) $\Theta(n \log n)$; c) $\Omega(n \log n)$

6. (4p) Dați un exemplu de mulțime nerecursivă și demonstrați pe scurt că aceasta este într-adevăr nerecursivă.

7. (4p) Demonstați că problema GAP este în clasa de complexitate SPACE((log n)**2).

8. (3p) a) Determinați acoperirea minimală cu vârfuri pentru un graf bipartit complet cu n + m vârfuri.
   (3.5p) b) Ce algoritm de aproximare cunoașteți care rezolvă problema anterioară? Descrieți acest algoritm (puteți folosi pseudocod sau în cuvinte) și menționați factorul său de aproximare.

9. (5p) Scrieți un algoritm nedeterminist pentru următoarea problemă:
Fie un alfabet Σ și un set de șiruri de caractere $S = \{s_1, s_2, .. s_n\} \subseteq \Sigma^*$. Să se determine dacă există un alt șir de caractere $s \subseteq \Sigma^*$, de dimensiune maximă k, care să conțină toate șirurile de caractere din S ($s_i \subseteq s \ \forall\ i=1..n$).

10. (3p) Verificaţi posibilitatea aplicării teoremei master şi calculaţi limite asimptotice de complexitate pentru recurenţa: $T(n) = 6*T(n/2) + n^2 + n*(\log n)^{100}$

11. (10p) Se consideră tipul de date LIST<N>, pentru care avem definiți constructorii:

```
[] : → LIST<N>
[a] : N → LIST<N>
cons(a, l) : N x (LIST<N> \ {[]}) → LIST<N>
```

și axiomele:

```
head(l): LIST<N> \ {[]} → N
(H1)  head([a]) = a
(H2)  head(cons(a, x)) = a
last(l): LIST<N> \ {[]} → N
(L1)  last([a]) = a
(L2)  last(cons(a, x)) = last(x)

append(l1, l2): LIST<N> x LIST<N> → LIST<N>
(A1)  append([], l2) = l2
(A2)  append([a], l2) = cons(a, l2)
(A3)  append(cons(a, x), l2) =
                  cons(a, append(x, l2))

sorted(l): LIST<N> → BOOLEAN
(S1)  sorted([]) = true
(S2)  sorted([a]) = true
(S3)  sorted(cons(a, x)) = sorted(x) && a <= head(x)
```

Verificați prin inducție structurală dacă următoarea proprietate este adevărată:

```
P(l) = ( l != [] → sorted(append(l, [a])) == sorted(l) && a >= last(l) )
          ∀ l ∈ LIST<N>
```

Total: 40p