Laurea in "Informatica"
Corso di "Algoritmi e Strutture di Dati"
18 Settembre 2012

1. *Tempo disponibile 180 minuti. È ammesso ritirarsi entro 90 minuti.*
2. *Sono ammessi al più 3 scritti consegnati per l'A.A. 2011/12 (Giugno 2012-Febbraio 2013)*
3. *Non è possibile consultare appunti, libri o persone, né uscire dall'aula.*
4. *Le soluzioni degli esercizi devono:*
   - *spiegare a parole l'algoritmo usato (anche con l'aiuto di eventuali disegni)*
   - *fornire e commentare lo pseudocodice (dettagliando a parole il significato delle variabili)*
   - *giustificare la correttezza e la complessità (con tutti i passaggi matematici necessari)*
5. *Un esercizio può ammettere più soluzioni: a soluzioni computazionalmente più efficienti e/o concettualmente più semplici sono assegnati punteggi maggiori*

1. Si calcoli la complessità $T(n)$ della seguente procedura ricorsiva, con $n > 0$, specificando se tale complessità è polinomiale o esponenziale nella *dimensione* dell'input:

```
integer mistery(integer [ ] A, integer n)
    integer k ← n + 2
    for integer i ← 2 to n do
        ⌊ k ← k – 1
    if n > 5 then
        | return ⌊n/k⌋ · mistery(A, ⌊n/k⌋) + mistery(A, ⌊n/k⌋)
    else
        ⌊ return ⌊n/k⌋
```

2. Dati un vettore A di n interi ed un intero x, si vuole decidere se esistono due elementi di A la cui somma sia x. Si scriva lo pseudo-codice di un algoritmo di complessità $O(n \log n)$, spiegandone il funzionamento (*nota*: si consiglia di utilizzare algoritmi già noti).

3. Si scriva lo pseudo-codice di una funzione che verifichi in tempo $O(n)$ se un albero binario T di n nodi è un albero di ricerca, usando gli operatori degli alberi binari.

4. Si scriva lo pseudo-codice della procedura *Depth-First Search (DFS)* vista a lezione e poi la si esegua sul grafo non orientato $G = (N, A)$, $N = \{1, 2, 3, 4, 5\}$, $A = \{[1,2], [1,3], [1,5], [2,5], [3,4], [3,5]\}$ a partire dal nodo 3, assumendo che gli insiemi di adiacenza dei nodi siano ordinati in senso crescente.

5. Dato un insieme di k stringhe, dette *primitive,* ed una stringa $X = x_1 \ldots x_n$, si vuole decidere se X è ottenibile come *concatenazione* di primitive. Per esempio, date le primitive {01, 10, 011, 101}, per $X = 0111010101$ la risposta è sì (due possibili soluzioni sono 011-10-10-101 e 011-101-01-01) mentre per $X = 0110001$ la risposta e no. Scrivere lo pseudo-codice di un algoritmo che risolva il problema utilizzando la *programmazione dinamica* (*nota*: ce n'è uno che richiede tempo $O(mn)$, dove m è la somma delle lunghezze delle stringhe primitive).

6. Dato un insieme A di n interi distinti, si vuole decidere se esiste una partizione di A in tre sottoinsiemi *non vuoti* S, T e U tali che la cardinalità di S sia il doppio di quella di T ed il prodotto degli elementi di S sia il triplo della somma degli elementi di U. Si scriva lo pseudo-codice di un algoritmo *non deterministico* che richieda tempo *polinomiale*.