Laurea in "Informatica"
Corso di "Algoritmi e Strutture di Dati"
13 Giugno 2013

1. *Tempo disponibile 180 minuti. È ammesso ritirarsi entro 90 minuti.*
2. *Sono ammessi al più 3 scritti consegnati per l'A.A. 2012/13 (Giugno 2013-Febbraio 2014)*
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 e si determini se tale complessità sia polinomiale o esponenziale nella dimensione dell'input:

integer mistery (**integer** n)
    integer $k \leftarrow \lfloor n/2 \rfloor$
    for integer $i \leftarrow n$ **downto** $\lfloor n/2 \rfloor$ **do** $k \leftarrow k$
    if $n < 17$ **then**
        return k
    else
        return $k \cdot$ mistery $(\lfloor n/2 \rfloor) + 2 \cdot$ mistery $(k) + \lfloor n/2 \rfloor$

2. Data una lista L di interi, si vuole togliere da L tutti gli elementi che compaiono una sola volta e inserirli in una nuova lista M, mantenendo in entrambe le liste l'ordine originario degli elementi (p.e. se in ingresso $L = 2, 1, 5, 4, 5, 2, 7, 4, 5$, il risultato è $L = 2, 5, 4, 5, 2, 4, 5$ ed $M = 1, 7$). Si scriva lo pseudo-codice di un algoritmo efficiente utilizzando gli *operatori* delle liste visti a lezione.

3. Si scriva lo pseudo-codice di un algoritmo che, dati un intero k ed un albero binario T i cui nodi contengono valori interi, cancella in tempo *ottimo* ogni *foglia* di livello k contenente un intero uguale a quello contenuto nel *padre*, assumendo T realizzato con *puntatori*.

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

5. Si descriva l'algoritmo di Kruskal visto a lezione per calcolare il minimo albero di copertura di un grafo non orientato pesato, dimostrandone correttezza e complessità. Successivamente, si esegua (a mano) l'algoritmo sul grafo dell'Esercizio 4, assumendo che i pesi degli archi siano $w[1,3]=1$, $w[1,4]=3$, $w[1,5]=4$, $w[2,3]=6$, $w[3,4]=5$, $w[4,5]=2$, e mostrando ad ogni passo il contenuto delle strutture di dati usate.

6. Dato un insieme A, di n interi distinti, si vuole decidere se esistono due sottoinsiemi non vuoti e disgiunti S e T di A tali che il doppio del prodotto degli elementi in S sia uguale a tre volte la somma degli elementi in T. Scrivere lo pseudo-codice di un algoritmo non deterministico che richieda tempo polinomiale.