Laurea in "Informatica"
Corso di "Algoritmi"
13 Luglio 2010

1. *Tempo disponibile 180 minuti. È ammesso ritirarsi entro 90 minuti.*
2. *Sono ammessi al più 3 scritti consegnati per l'A.A. 2009/10 (Giugno 2010-Febbraio 2011)*
3. *Non è possibile consultare appunti, libri o persone, né uscire dall'aula.*
4. *Le soluzioni degli esercizi devono:*
   a. *spiegare a parole l'algoritmo usato (anche con eventuali disegni)*
   b. *commentare l'eventuale procedura Pascal (dettagliando il significato delle variabili)*
   c. *giustificare la correttezza e tutti i passaggi matematici*
   d. *dimostrare la complessità (con equazioni di ricorrenza se necessario)*

1. Dato un albero binario B i cui nodi contengono interi, si vuole aggiungere ad ogni foglia, che contiene un valore pari, un figlio destro che contenga il valore del padre aumentato di 1. Si scriva una procedura Pascal ricorsiva di complessità ottima assumendo che B sia realizzato con puntatori.

2. Data una lista L di interi, si vuole modificarla cancellando tutti gli elementi adiacenti che hanno valori identici, mantenendo l'ordine iniziale degli elementi (p.e. se l'ingresso è L = 6, 6, 6, 2, 3, 1, 1, 7, 7, 4 allora il risultato è L = 2, 3, 4). Si scriva una procedura Pascal efficiente *utilizzando gli operatori* per le liste visti a lezione.

3. Si valuti l'ordine di grandezza della complessità $T(n)$ della funzione Pascal:

```
function PIPPO(n: integer): integer;
    var j, k: integer;
    begin
        for j := 2 to 4 do k := j;
        if n > 7 then PIPPO := n * PIPPO(n div k) + PIPPO(k div k) div k
                 else PIPPO := 7;
    end;
```

4. Si modifichi la procedura Pascal Mergesort vista a lezione, per ordinare un vettore A di n interi, utilizzando sempre la tecnica *divide-et-impera*, ma dividendo il vettore in 3 sottosequenze, ciascuna di circa $n/3$ elementi, in modo che la complessità resti $O(n \log n)$ (per semplicità, si puo' assumere che tutti gli n elementi siano distinti e che n sia una potenza di 3).

5. Si scriva la procedura Pascal *Depth-First Search (DFS)* vista a lezione. 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 i vettori di adiacenza siano ordinati in modo *crescente* e mostrando il contenuto dei vettori di adiacenza.

6. Dati un insieme A di interi ed un intero k, si vuole decidere se esiste una partizione di A in due sottoinsiemi B e C tali che il prodotto degli elementi di B sia uguale a k volte la somma degli elementi di C. Si scriva una procedura Pascal non deterministica di complessità polinomiale.