Laurea in "Informatica"
Corso di "Algoritmi e Strutture Dati"
17 Gennaio 2007

1. *Tempo disponibile 180 minuti. È ammesso ritirarsi entro 90 minuti.*
2. *Sono ammessi al più 3 scritti consegnati tra Giugno 2006 e Febbraio 2007.*
3. *Non è possibile consultare appunti, libri o persone, né uscire dall'aula.*
4. *Per raggiungere la sufficienza occorrono* almeno *3 esercizi risolti senza alcun errore.*
5. *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. Si valuti la complessità $T(n)$ della seguente funzione Pascal:

```
function PIPPO(n: integer): integer;
  var i, j: integer;
  begin
    if n > 13 then begin
      for j := n – 1 to n + 1 do i := 2;
      PIPPO := 3*PIPPO(n div 2)*PIPPO(n div i) + n*i
    end else if n > 17
      then PIPPO := 3*PIPPO(n + 2) + 3*i
      else PIPPO := 5
  end;
```

2. Dato un albero binario A implementato con puntatori, in cui ogni nodo contiene un intero, scrivere una funzione Pascal di *complessità ottima* che modifichi A aggiungendo un figlio sinistro (contenente 0) ad ogni foglia che contiene un elemento dispari.

3. Data una lista L di interi, si vuole togliere da L ogni elemento che compare solo una volta e inserirlo in una nuova lista M, mantenendo in entrambe le liste l'ordine originario degli elementi (p.e. se in input L = 2, 5, 1, 4, 5, 7, 4, allora in output L = 5, 4, 5, 4 e M = 2, 1, 7). Si scriva una procedura Pascal efficiente utilizzando gli operatori delle liste visti a lezione.

4. Si descriva la struttura di dati Merge-Find-Set (MFSET), con le operazioni tipicamente utilizzate, una sua realizzazione efficiente, e si illustri algoritmo dove l'utilizzo di tale struttura è vantaggiosa.

5. Dati un array A ordinato di n interi distinti e un intero z, scrivere una procedura Pascal efficiente per determinare il numero di coppie di elementi che danno per somma z. Ad esempio, se A = [1,2,3,4,5,6,8] e z = 9, il risultato è 3 (1+8, 3+6, 4+5).

6. Dati un insieme A, di n interi distinti, ed un intero k, si vuole decidere se esiste un sottoinsieme S di A tale che il prodotto degli elementi in S sia uguale a k volte la somma degli elementi in $A - S$. Scrivere (in pseudo-codice) un algoritmo non deterministico che richieda tempo polinomiale.