Prof. L. Margara **Scritto 4/06/2012**

Algoritmi e Strutture Dati

Esercizio 1.[11 punti]
Risolvere in ordine di grandezza la seguente equazione ricorsiva.

$$T(n) = \begin{cases} 1 & \text{if } n \leq c, \\ 23T(n/600) + 77T(n/601) + 1 & \text{if } n > c. \end{cases}$$

Traccia Soluzione.

$T(n) = 23T(n/600) + 77T(n/601) + 1$
caso: 1.
$T(n) \leq 23T(n/600) + 77T(n/600) + 1 = 100T(n/600) + 1$
caso: 2.
$T(n) \geq 23T(n/601) + 77T(n/601) + 1 = 100T(n/601) + 1$
quindi:
$100T(n/601) + 1 \leq T(n) \leq 100T(n/600) + 1$

visto che:
$100T(n/600) + 1$ è $\Theta(n^{log_{600}(100)})$ e che
$100T(n/601) + 1$ è $\Theta(n^{log_{601}(100)})$

concludiamo:
$T(n) = \Omega(n^{log_{601}(100)})) = \Omega(n^{0.719716})$
$T(n) = O(n^{log_{600}(100)})) = O(n^{0.719903})$

Esercizio 2.[11 punti]
Scrivere una FUNZIONE RICORSIVA che prenda in input il puntatore alla radice di un albero binario e restituisca l'intero positivo n definito come segue.
n è la lunghezza della più lunga sequenza *verticale* e monocromatica (stesso colore) di nodi dell'albero. Una sequenza di nodi $u_1, u_2, \ldots, u_n$ è verticale se u_1 é la radice dell'albero e u_i è il padre di u_{i+1} con $1 \leq i \leq n-1$.
Argomentare la sua correttezza e analizzare il suo costo computazionale. Vietato usare variabili globali !!!

Traccia Soluzione esercizio.


```
F(p):
    if p = nil then return 0
    if (p ha il figlio sin) and (colore di p = colore figlio sin di p)
        then s = F(figlio sin di p)
        else s = 0
    if (p ha il figlio des) and (colore di p = colore figlio des di p)
        then d = F(figlio des di p)
        else d = 0
    return Max(s, d) + 1
```

Costo computazionale: $T(n) = \Theta(n)$

Esercizio 3.[11 punti]
Sia $P = \{p_1, p_2, \ldots, p_n\}$ un insieme di n numeri interi positivi e T un numero intero positivo. Dare un algoritmo Greedy polinomiale per calcolare un sottoinsieme S di P di cardinalità massima tale per cui la somma dei numeri in S sia minore o uguale a T. Determinare se l'algoritmo proposto è ottimo oppure no. Calcolare il costo computazionale dell'algoritmo proposto.

Traccia Soluzione esercizio.

```
F(v,T):
    w = Sort(v)
    s = 0
    i = 1
    n = length(w)
    sol = insieme vuoto
    while (i ≤ n) and (s + w[i] ≤ T)
        s = s + w[i]
        i = i + 1
        sol = sol ⋃ w[i]
    return sol
```

Costo computazionale: $T(n) = \Theta(n\ log(n))$