Prof. L. Margara **Scritto 13/06/2014**

Algoritmi e Strutture Dati

Scrivere su ogni foglio consegnato il numero di matricola trascurando gli zeri iniziali.

Esercizio 1.[10 punti]
Scrivere una funzione RICORSIVA che prenda in input un albero binario A (con nodi colorati di rosso o di verde) e restituisca in output il massimo tra l'altezza rossa e l'altezza verde dell'albero. La funzione deve essere ricorsiva e scritta in pseudo codice. Analizzare il costo computazionale della procedura proposta e argomentarne la correttezza.

Esercizio 2.[10 punti]
Trovare l'ordine di grandezza di $T(n)$ definita come segue:

$$T(n) = \begin{cases} 1 & \text{if } n \leq 2, \\ T(n-1) + \sqrt{n} & \text{if } n > 2. \end{cases}$$ con k intero costante maggiore di uno.

Esercizio 3.[10 punti]
Dato un vettore $A[1..n]$ composto da n numeri reali arbitrari diversi da zero. Sia $B[1..n]$ il vettore ottenuto da $A[1..n]$ in modo tale che $B[i] = A[1] * A[2] * ... * A[n]/A[i]$. Scrivere un algoritmo di costo $O(n)$ per calcolare tutti i valori del vettore $B[1..n]$, senza effettuare divisioni.

Traccia Soluzione esercizio 1
ALGORITMO(RedGreenHigh[p])

IF p=nil THEN Return(0,0,0)
< RedLeft,GreenLeft,MaxLeft>=RedGreenHigh[p.left]
< RedRight,GreenRight,MaxRight>=RedGreenHigh[p.right]

IF p.color = green THEN
Red=Max(RedLeft,RedRight); Green=Max(GreenLeft,GreenRight) +1

IF p.color = red THEN
Red=Max(RedLeft,RedRight) +1 ; Green=Max(GreenLeft,GreenRight)

High = Max(Red,Green)
Return(Red,Green,High)

Costo $\Theta(n)$

Traccia Soluzione esercizio 2 Notiamo innanzitutto che

$$T(n) = \sqrt{n} + \sqrt{n-1} + \cdots + T(2) + T(1)$$

1. Ovviamente $T(n)$ risulta essere minore del numero degli elementi della sommatoria per l'elemento più grande della sommatoria stessa, ovvero $T(n) \leq n\sqrt{n}$

2. $T(n) > \sqrt{n} + \sqrt{n-1} + \cdots + \sqrt{\frac{n}{2}} > \frac{n}{2}\sqrt{\frac{n}{2}}$
3. Da 1 e 2 si deduce che $T(n) = \Theta(n\sqrt{(n)})$

Traccia Soluzione esercizio 3 Notiamo innanzitutto che $B[i]$ corrisponde al prodotto di tutti gli elementi di A tranne $A[i]$. Definiamo due array $P[1..n]$ e $S[1..n]$. $P[i]$ contiene il prodotto dei valori del sottovettore $A[1..i-1]$; $P[1]$ vale 1. $S[i]$ contiene il prodotto dei valori del sottovettore $A[i+1..n]$; $S[n]$ vale 1. Una volta che i valori di $P[1..n]$ e $S[1..n]$ sono stati calcolati, si ha che $B[i] = P[i] * S[i]$. Il calcolo dei valori $P[i]$ e $S[i]$ si può effettuare in tempo complessivo $O(n)$.

```
ALGORITMO(A[1..n])

P[1] = 1
For i = 2 to n do P[i] = P[i-1] * A[i-1]

S[n] = 1
For i = n-1 downto 1 do S[i] = S[i+1] * A[i+1]

For i = 1 to n do B[i] = P[i] * S[i]

Return B
```