Prof. L. Margara **Scritto del 19/6/2023**

Algoritmi e Strutture Dati

Esercizio 1.[11 punti]
Calcolare in ordine di grandezza il costo computazionale della funzione FIVE. Dire quanto vale FIVE(5673256734). Giustificare sempre tutte le risposte.

```
FIVE(n)
if n ≤ 5
      return 5
else
      return FIVE(⌊⁵√n⌋)
```

Traccia Soluzione esercizio 1
$T(n) = T(n^{\frac{1}{5}}) + k =$
$T(n^{\frac{1}{25}}) + k + k =$
$\vdots$
$T(n^{\frac{1}{5^i}}) + k + k + \cdots + k =$

Calcoliamo quanto deve essere grande i in modo tale che $n^{\frac{1}{5^i}} \leq 5$?
$n^{\frac{1}{5^i}} \leq 5$
$\log_5(n^{\frac{1}{5^i}}) \leq \log_5(5) = 1$
$\frac{1}{5^i}\log_5(n) \leq 1$
$\log_5(n) \leq 5^i$
$\log_5(\log_5(n)) \leq \log_5(5^i)$
$\log_5(\log_5(n)) \leq i$

E quindi $T(n) = \Theta(log(log(n))$

FIVE(n) restituisce sempre 5 e quindi FIVE$(5673256734) = 5$

Esercizio 2.[11 punti]
Sia A un albero binario i cui nodi contengono due puntatori (uno al figlio sinistro e uno al figlio destro). Scrivere una FUNZIONE RICORSIVA che prenda in input A e restituisca in output il numero di nodi con esattamente 3 nipoti. La funzione deve essere ricorsiva e scritta in pseudo codice. Analizzare il costo computazionale della funzione proposta.

Traccia Soluzione esercizio 2.

```
TRE-NIPOTI(p)
if p == NIL
    return FALSE
if p.left == NIL
    return FALSE
if p.right == NIL
    return FALSE
nipoti = 0
if p.left.left ≠ NIL
    nipoti = nipoti + 1
if p.left.right ≠ NIL
    nipoti = nipoti + 1
if p.right.left ≠ NIL
    nipoti = nipoti + 1
if p.right.right ≠ NIL
    nipoti = nipoti + 1
if nipoti == 3
    return TRUE
else
    return FALSE
```

```
CONTA-TRE-NIPOTI(p)
if p == NIL
    return 0
l = CONTA-TRE-NIPOTI(p.left)
r = CONTA-TRE-NIPOTI(p.right)
if TRE-NIPOTI(p)
    return l + r + 1
else
    return l + r
```

Costo $\Theta(n)$.

Esercizio 3.[11 punti]
Fornire un algoritmo greedy polinomiale (descrivendo prima l'idea a parole e poi fornendo lo pseudocodice) per risolvere "Coloring" (dato un grafo non pesato e non orientato colorare i suoi nodi con il numero minimo di colori, evitando di assegnare lo stesso colore a coppie di nodi collegate da un arco). Dimostrare che l'algoritmo fornito non è ottimo. Determinare il suo costo computazionale.

Traccia della soluzione dell'esercizio 3.
L'esercizio in questione può essere risolto in molti modi diversi. Qui di seguito riportiamo una possibile soluzione.

$Coloring(G)$

```
while esistono nodi non ancora colorati
      scegli un nodo u non ancora colorato
      se possibile, assegna a u un colore già assegnato
      altrimenti assegna a u un colore nuovo
```

Il costo computazionele di $Coloring$ è evidentemente polinomiale...
Controesempio: sia $G = (\{1, 2, 3, 4, 5\}, \{(1, 2), (1, 3), (2, 3), (2, 4), (2, 5), (3, 5), (4, 5)\})$

colorazione prodotta dall'algoritmo non ottima:
nodo 1 $\rightarrow$ *colore* A
nodo 2 $\rightarrow$ *colore* B
nodo 3 $\rightarrow$ *colore* C fino a qui la scelta dei colori è obbligata
nodo 4 $\rightarrow$ *colore* A scelgo un colore ammissibile già assegnato (scelta greedy sbagliata)
nodo 5 $\rightarrow$ *colore* D serve un quarto colore !

colorazione ottima:
nodo 1 $\rightarrow$ *colore* A
nodo 2 $\rightarrow$ *colore* B
nodo 3 $\rightarrow$ *colore* C fino a qui la scelta dei colori è obbligata
nodo 4 $\rightarrow$ *colore* C scelgo un colore ammissibile già assegnato
nodo 5 $\rightarrow$ *colore* A 3 colori invece di 4 !