Esercizi Complessità

1. Studiate la complessità in tempo del seguente algoritmo iterativo:

```
Fill( Array of int A):int
int n := length(A);
int z := 1;
int i := 0;
for i := 1; i++; n do
    A[i]:= 2^i
for i := 1; i++; n do
    j := A[i];
    while j ≥ 1 do
        z := z + 1; j := ⌊j/2⌋;
return: z
```

2. Studiate con l'albero della ricorsione la complessità in tempo del seguente algoritmo ricorsivo: Provate per induzione la stima fatta.

```
Tip( n :int):int
int z := 1;
for i := 1; i++; n do
    for j := 1; j++; n/i do
        z := z + 1
if (n > 2) then
    return: Tip(⌊n/2⌋) * Tip(⌊n/2⌋ + 2 * z
else
    return: z
```

Soluzioni

1. Per stimare la complessità in tempo del codice proposto è necessario studiare la complessità dei suoi cicli, i.e., in questo caso 3.

   Studiamo quindi il primo ciclo. Il primo FOR parte da 1 fino ad n con passo 1, lo possiamo dunque espriremere come:

   $$\sum_{i=1}^{n} \mathcal{O}(1) = \Theta(n)$$

   Per quanto riguarda i restanti due cicli, il primo FOR ancora una volta ci costa $\Theta(n)$, mentre per quanto riguarda il WHILE innestato al suo interno è neccesario ragionare sul valore assunto da j.

   Innanzitutto osserviamo che la condizione di arresto si raggiunge quando $j < 1$, e che il valore di j dimezza ad ogni passo. Quindi:

Step	1	2	3	...	k
j	$\frac{j}{2}$	$\frac{j}{4}$	$\frac{j}{8}$	...	$\frac{j}{2^k}$

   Impostiamo quindi la disequazione per studiare quanti passi dovremo eseguire prima di arrestarci:

   $$\frac{j}{2^k} < 1$$

   Osservando Riga.8 si può facilmente constatare che il valore di j dipende dal valore assegnato $A[i]$, che si può facilmente desumere da Riga.6, i.e., 2^i. Sostiuiamo quindi $j = 2^i$:

   $$\frac{2^i}{2^k} < 1 \quad \Rightarrow \quad \frac{1}{2^k} < \frac{1}{2^i} \quad \Rightarrow \quad k \log 2 \geq i \log 2 \quad \Rightarrow \quad k \geq i$$

   Da questa analisi abbiamo scoperto che il ciclo WHILE più interno esegue un numero di passi $\geq i$. Dunque il costo del nostro algoritmo sarà:

   $$\Theta(n) + \sum_{i=1}^{n} \sum_{j=1}^{i} \Theta(1) \quad = \quad \Theta(n) + \sum_{i=1}^{n} i \quad = \quad \Theta(n) + \frac{n(n+1)}{2} \quad \Rightarrow \quad \text{FILL} \in \Theta(n^2).$$

2. Al fine di descrivere l'equazioni di ricorrenza dobbiamo prima di tutto comprendere il costo della fase "combina"; in questo caso data dai due cicli FOR.

   Quindi per prima cosa studiamo la loro complessità. Il primo ciclo FOR va da 1 ad n con passo costante, mentre il secondo va da 1 a n/i. Considerato che il secondo FOR è innestato all'interno dell'altro, possiamo dunque scrivere:

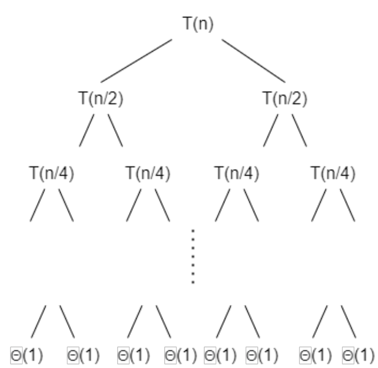
$$\sum_{i=1}^{n}\sum_{j=1}^{n/i}\Theta(1) \quad = \quad \sum_{i=1}^{n} n/i \quad = \quad n\sum_{i=1}^{n}\frac{1}{i} \quad < \quad n\sum_{i=1}^{\infty}\frac{1}{i} \quad = \mathcal{O}(n\log n)$$

   Possiamo quindi definire l'equazione di ricorrenza:

$$T(n) = \begin{cases} 2T(\lfloor n/2 \rfloor) + \mathcal{O}(n\log n) & \text{if} \quad n > 2 \\ \Theta(1) & \text{if} \quad n \leq 2 \end{cases}$$

   Nota bene: in questo caso il fattore di ramificazione è 2 ($a = 2$) perchè ci sono esattamente 2 chiamate di TIP.

   Costruiamo l'albero di ricorrenza:



   Nodo generico: $T(n/2^i)$

   Altezza: $\frac{n}{2^i} \leq 2 \Rightarrow i > \log\frac{n}{2}$

   Costo livello generico: $2^i(\frac{n}{2^i}\log\frac{n}{2^i}) = n\log\frac{n}{2^i}$

   Numero foglie: $2^{\log\frac{n}{2}} = (\frac{n}{2})^{\log 2} = \frac{n}{2}$

   Possiamo quindi stimare un costo per la nostra ricorrenza:

$$\sum_{i=1}^{\log\frac{n}{2}-1}(n\log\frac{n}{2^i}) + \frac{n}{2} \quad = \quad n\sum_{i=1}^{\log\frac{n}{2}-1}(\log n - i\log 2) + \frac{n}{2} \quad = \quad n(\sum_{i=1}^{\log\frac{n}{2}-1}\log n - \sum_{i=1}^{\log\frac{n}{2}-1} i) + \frac{n}{2} \leq$$

$$\leq n(\log^2 n - \frac{\log n(\log n - 1)}{2}) + \frac{n}{2} \quad = \quad \frac{n\log^2 n}{2} + \frac{n\log n}{2} + \frac{n}{2} \quad \Rightarrow \quad \in \mathcal{O}(n\log^2 n)$$

   Proviamo il risultato ottenuto per induzione:

   Ipotesi: $\exists c > 0, n_0 > 0 : \forall n \geq n_0 T(n/2) \leq 2(c(\frac{n}{2}\log^2\frac{n}{2}))$

$$T(n) \leq 2c\frac{n}{2}\log^2\frac{n}{2} + n\log n \quad \leq \quad cn(\log n - log2)^2 + n\log n \quad \leq \quad cn(\log^2 n + 1 - 2\log n) + n\log n \leq$$

$$\leq cn\log^2 n + cn - 2cn\log n + n\log n \quad \leq \quad cn\log^2 n + \underbrace{cn - (2c-1)n\log n}_{<0} \leq (*)$$

   In particolare

$$cn - (2c-1)n\log n < 0$$
$$c - (2c-1)\log n < 0$$
$$c < (2c-1)\log n$$

   Quindi è sufficiente prendere una $c = 2$ per verificare la disequazione, i.e., $2 < 3\log n$. Segue:

$$(*) \leq cn\log^2 n$$