# 1: Lista de Exercícios

Hugo Silveira Sousa      Matr.: 378998

1: D) $\frac{n}{1000}$ não é $O(1)$

$\frac{n}{1000} = \frac{1}{1000} n = O(n)$

$\frac{1}{1000} n \leq c \cdot n \qquad c \geq \frac{1}{1000}$

$\frac{n}{1000}$ não tem $O()$ constante.

I) $34 n \log n^2 + 13n$ é $\Omega(n)$ e $O(n^2)$

$\Omega(n) = ?$

$34n \cdot \log n^2 + 13n \geq c \cdot n$

$34 \log n^2 + 13 \geq c$

p/ um $n_0 = 1$, $c \leq 13$ é verdadeira

$O(n^2) = ?$

$34n \cdot \log n^2 + 13n \geq c \cdot n^2$

$34 \log n^2 + 13 \geq c \cdot n$

$\nexists c$ que satisfaça

$O(34n \cdot \log n^2 + 13n)$

$O(34 n \log n^2)$

$O(n \log n^2)$

(6) $\sum_{i=1}^{n} i^k$ é $\Theta(n^{k+1})$

$O(n^{k+1}) = ?$

$O(1^k + 2^k + \dots + n^k) = O(n^k) \leq O(n^{k+1})$

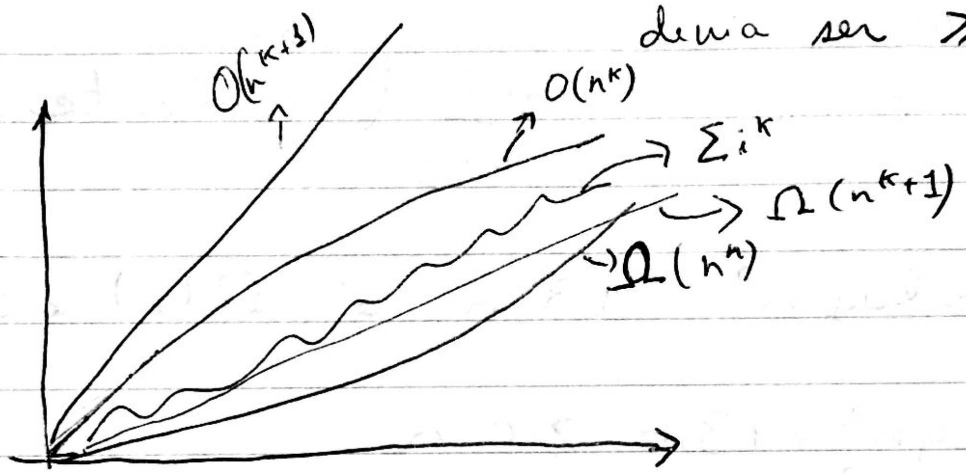
$O(n^{k+1})$ ✓

$\Omega(n^{k+1}) = ?$

$\Omega(1^k + 2^k + \dots + n^k) = \Omega(n^k) \leq \Omega(n^{k+1})$

deveria ser $\geq \Omega(n^{k+1})$

Ex:




tilibra

2: (c) MergeSort

```
mergesort (vetor, inicio, fim){
    se (inicio < fim) {
        meio = arredonda_baixo ((inicio + fim)/2)
        mergesort (vetor, inicio, meio)
        mergesort (vetor, meio+1, fim)
        conquista (vetor, inicio, meio, fim)
    }
}
```

* O método conquista vai unindo os vetores que estão separados de forma ordenada.

* O pior e o melhor caso tem a mesma complexidade pois no caso de um vetor já ordenado, ele vai continuar todo o algoritmo para conferir se esta ordenado, se não estiver ele apenas insere as posições de forma trocada para ordenar.

* Como o algoritmo divide o problema em problemas menores, temos um fator log n, cada divisão tem um fator n, então $O(n \cdot \log n)$ e $\Omega(n \cdot \log n)$

( / / )

8º i) Criar_lista_L'();

```
*ultimo_lista_L' =*inicio_lista_L -> prox
ultimo_lista_L' -> prox = NULL
aux_L = inicio_lista_L -> prox
aux_L' = inicio_lista_L'
para ( i = 1 ; i ≤(tamanho(L) - 1) ; i++){
    * aux_L' -> prox = * aux_L -> prox
    aux_L = aux_L -> prox
    aux_L' = aux_L' -> prox
} aux_L' -> prox = ultimo_lista_L'
```

(ii) Criar_lista_L'()

```
x = inicio
y = x -> prox
inicio -> prox = NULL
para ( i = 1 ; i ≤ (tamanto_lista) ; i++){
        z = y
        y->prox = x
        x = y
        y = z -> prox
}
    inicio = y
```

(iii) criar_lista_L'();
para (i = 1 ; i ≤ ((tamanho)/2) ; i++ ){
  Lista_x -> prox = L -> prox
  Lista_x -> prox = lista_x -> prox -> prox
  L -> prox = L -> prox -> prox
} para ( j = i ; j < tamanho(L) ; j++ ){
  Lista_y -> prox = L -> prox
  lista_y -> prox = lista_y -> prox -> prox
  L -> prox = L -> prox -> prox
} L' = soma (Lista_x + Lista_y)