Algoritmos e Complexidade
Ficha 2
Resolução

Eduardo Freitas Fernandes

2025

1 Contagem

Exercício 1

Função `bubbleSort()`:

Comparações entre elementos do array: O número de comparações entre elementos do array é igual em todos os casos, logo não existem situações em que o número de comparações mude.

Trocas Efetuadas:

- Melhor caso: array ordenado por ordem crescente (condição do `if statement` é sempre falsa)
- Pior caso: array ordenado por ordem decrescente (condição do `if statement` é sempre verdadeira)

Número de comparações:

$$\sum_{i=1}^{N-1} \sum_{j=0}^{i-1} 1 = \sum_{i=1}^{N-1} i = (N-1) \times N = \Theta(N^2)$$

Função `iSort()`:

Comparações entre elementos do array:

- Melhor caso: array ordenado por ordem crescente (apenas uma comparação no loop interno, de cada iteração do loop externo)
- Pior caso: array ordenado por ordem decrescente

Para as trocas de elementos, o melhor e pior caso são os mesmos, dado que o `swap` depende da condição de paragem do `for loop`.

Melhor caso:

$$\sum_{i=1}^{N-1} 1 = N - 1 = \Omega(N)$$

Pior caso:

$$\sum_{i=1}^{N-1} \sum_{j=1}^{i} 1 = \sum_{i=1}^{N-1} i = (N-1) \times N = O(N^2)$$

Exercício 2
Exercício 3

$$T(N) = ...$$

```
int maxSoma (int v[], int N) {

}
```

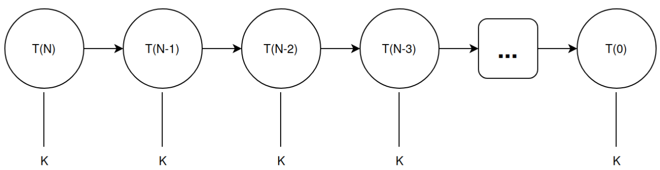
$$T(N) = ...$$

Exercício 4

2 Definições Recursivas

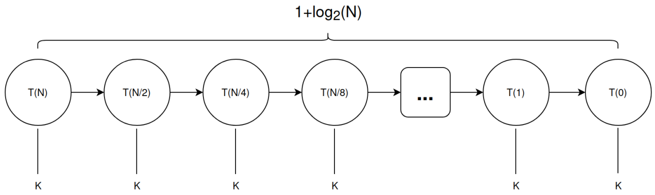
Exercício 1

a)



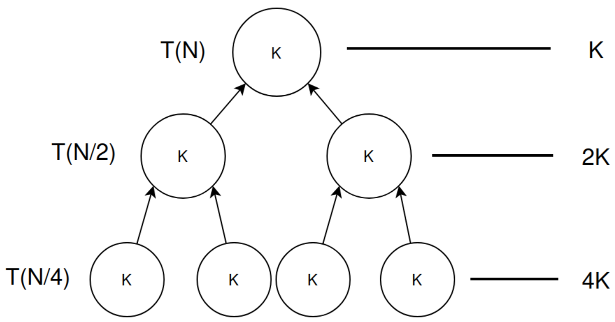
$$T(N) = \sum_{i=0}^{N} K = (N+1) \times K = \Theta(N)$$

b)



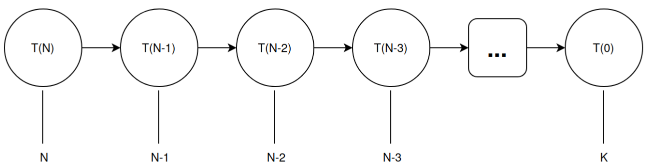
$$T(N) = \sum_{i=0}^{1+\log_2(N)} K = (2 + \log_2(N)) \times K = \Theta(\log_2(N))$$

c)



$$T(N) = \sum_{i=0}^{1+\log_2(N)} 2^i \times K = K \times (2^{\log_2(N)+2} - 1) = K \times (4 \times N - 1) = \Theta(N)$$

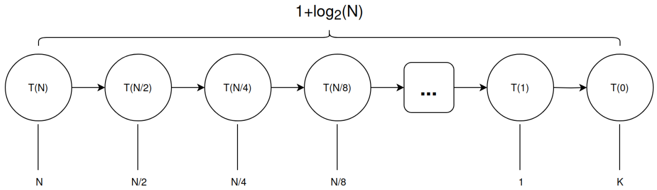
d)



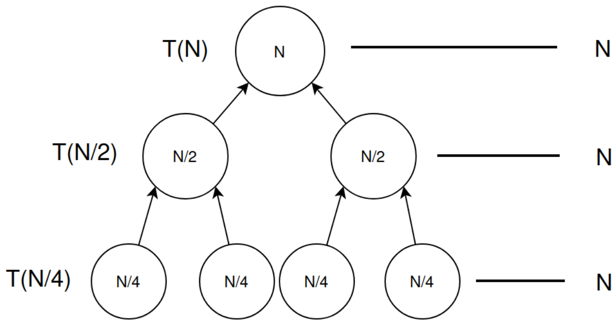
$$T(N) = K + \sum_{i=1}^{N} i = K + \frac{N \times (N+1)}{2} = \Theta(N^2)$$

e)

$$T(N) = k + \sum_{i=1}^{1+\log_2(N)} \frac{N}{2^i} = K + 2^{\log_2(N)+1} - 1 = K + 2 \times N - 1 = \Theta(N)$$



f)



$$T(N) = k + \sum_{i=1}^{1+\log_2(N)} N = K + N \times (1 + \log_2(N)) = \Theta(N \times \log_2(N))$$

Exercício 2

$$T(N) = \begin{cases} 0 & N \leq 0 \\ 1 + N - 1 + T(N-1) & N > 0 \end{cases} = \begin{cases} 0 & N \leq 0 \\ N + T(N-1) & N > 0 \end{cases}$$

$$= \sum_{i=1}^{N} i = \frac{N \times (N+1)}{2} = \Theta(N^2)$$

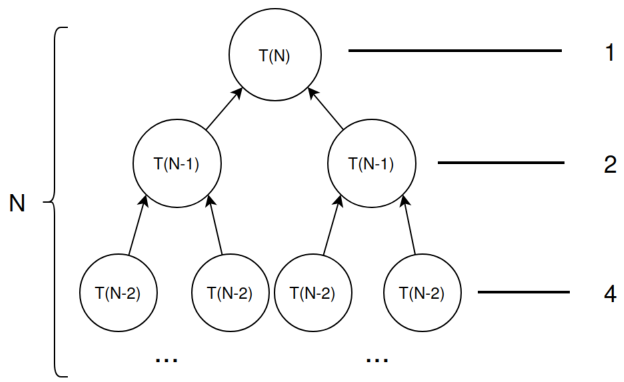
Exercício 3

$$T(N) = \begin{cases} 0 & N \leq 0 \\ 1 + 2 \times T(N-1) & N > 0 \end{cases}$$

$$T(N) = \sum_{i=0}^{N-1} 2^i = 2^N - 1 = \Theta(2^N)$$

Exercício 4

$$T(N) = \begin{cases} 1 & N \leq 1 \\ T_{mergeH}(N) + 2 \times T(N/2) & N > 1 \end{cases} = \begin{cases} 1 & N \geq 1 \\ 2 \times N + 2 \times T(N/2) & N > 1 \end{cases}$$



$$= \sum_{i=1}^{1+\log_2(N)} 2 \times N = 2 \times N \times (1 + \log_2(N)) = \Theta(N \times \log_2(N))$$

Exercício 5

Árvores Equilibradas

$$T(N) = \begin{cases} 0 & N \leq 0 \\ 1 + 2 \times T(\frac{N-1}{2}) & N > 0 \end{cases} = \begin{cases} 0 & N \leq 0 \\ 1 + 2 \times T(N/2) & N > 0 \end{cases}$$

$$= \sum_{i=0}^{1+\log_2(N)} 2^i = 4 \times N - 1 = \Theta(N)$$

Árvores "Lista"

$$T(N) = \begin{cases} 0 & N \leq 0 \\ 1 + T(N-1) & N > 0 \end{cases} \quad = \quad \sum_{i=1}^{N} 1 = N = \Theta(N)$$

3 Análise de Caso Médio

Exercício 1
Exercício 2
Exercício 3
Exercício 4
Exercício 5
Exercício 6

4 Análise Amortizada

Exercício 1
Exercício 2
Exercício 3