

Lista de exercícios

1 Introdução

Exercício 1.1

Assista ao vídeo “*What’s an algorithm?*” (David J. Malan) em <https://youtu.be/6hf0vs8pY1k>.

Exercício 1.2

Assista ao vídeo “*Top 7 Data Structures for Interviews*” em <https://youtu.be/cQWr9DFE1ww>.

2 Complexidade de algoritmos

Exercício 2.1

Desenhe o gráfico das funções $8n$, $4n \log n$, $2n^2$, n^3 e 2^n usando uma escala logarítmica para os eixos x e y , isto é, se o valor da função $f(x)$ é y , desenhe esse ponto com a coordenada x em $\log x$ e a coordenada y em $\log y$.

Exercício 2.2

O número de operações executadas por dois algoritmos A e B é $8n \log n$ e $2n^2$, respectivamente. Determine n_0 tal que A seja melhor que B para $n \geq n_0$.

Exercício 2.3

O número de operações executadas por dois algoritmos A e B é $40n^2$ e $2n^3$, respectivamente. Determine n_0 tal que A seja melhor que B para $n \geq n_0$.

Exercício 2.4

Ordene as funções a seguir por sua taxa assintótica de crescimento.

- $4n \log n + 2n$
- 2^{10}
- $3n + 100 \log n$
- $4n$
- 2^n
- $n^2 + 10n$
- n^3
- $n \log n$

Exercício 2.5

Qual a complexidade assintótica no pior caso (em termos de \mathcal{O}) do algoritmo abaixo?

```
1  /** Returns the sum of the integers in given array. */
2  public static int alg1(int[] arr) {
3      int n = arr.length, total = 0;
4      for (int j=0; j < n; j++)
5          total += arr[j];
6      return total;
7  }
```

Exercício 2.6

Qual a complexidade assintótica no pior caso (em termos de \mathcal{O}) do algoritmo abaixo?

```
1  /** Returns the sum of the integers with even index in given array. */
2  public static int alg2(int[] arr) {
3      int n = arr.length, total = 0;
4      for (int j=0; j < n; j += 2)
5          total += arr[j];
6      return total;
7  }
```

Exercício 2.7

Qual a complexidade assintótica no pior caso (em termos de \mathcal{O}) do algoritmo abaixo?

```
1  /** Returns the sum of the prefix sums of given array. */
2  public static int alg3(int[] arr) {
3      int n = arr.length, total = 0;
4      for (int j=0; j < n; j++)
5          for (int k=0; k <= j; k++)
6              total += arr[k];
7      return total;
8  }
```

Exercício 2.8

Qual a complexidade assintótica no pior caso (em termos de \mathcal{O}) do algoritmo abaixo?

```
1  /** Returns the sum of the prefix sums of given array. */
2  public static int alg4(int[] arr) {
3      int n = arr.length, prefix = 0, total = 0;
4      for (int j=0; j < n; j++) {
5          prefix += arr[j];
6          total += prefix;
7      }
8      return total;
9  }
```

Exercício 2.9

Qual a complexidade assintótica no pior caso (em termos de \mathcal{O}) do algoritmo abaixo?

```
1  /** Returns the number of times second array stores sum of prefix sums from first. */
2  public static int alg5(int[] first, int[] second) {
3      int n = first.length, count = 0;
4      for (int i=0; i < n; i++) {
5          int total = 0;
6          for (int j=0; j < n; j++)
7              for (int k=0; k <= j; k++)
8                  total += first[k];
9          if (second[i] == total) count++;
10     }
11     return count;
12 }
```

Exercício 2.10

O algoritmo **A** executa uma computação em tempo $\mathcal{O}(\log n)$ para cada entrada de um arranjo de n elementos. Qual o pior caso em relação ao tempo de execução de **A**?

Exercício 2.11

Dado um arranjo **X** de n elementos, o algoritmo **B** escolhe $\log n$ elementos de **X**, aleatoriamente, e executa um cálculo em tempo $\mathcal{O}(n)$ para cada um. Qual o pior caso em relação ao tempo de execução de **B**?

Exercício 2.12

Dado um arranjo **X** de n elementos inteiros, o algoritmo **C** executa uma computação em tempo $\mathcal{O}(n)$ para cada número par de **X** e uma computação em tempo $\mathcal{O}(\log n)$ para cada elemento ímpar de **X**. Qual o melhor caso e o pior caso em relação ao tempo de execução de **C**?

Exercício 2.13

Dado um arranjo **X** de n elementos, o algoritmo **D** chama o algoritmo **E** para cada elemento **X**[*i*]. O algoritmo **E** executa em tempo $\mathcal{O}(i)$ quando é chamado sobre um elemento **X**[*i*]. Qual o pior caso em relação ao tempo de execução do algoritmo **D**?

Exercício 2.14

Implemente os algoritmos `disjoint1` e `disjoint2` (apresentados nos materiais de aula), e execute uma análise experimental dos seus tempos de execução. Visualize seus tempos de execução como uma função do tamanho da entrada usando um gráfico *di-log*.

Exercício 2.15

Execute uma análise experimental para testar a hipótese de que o método da biblioteca Java, `java.util.Arrays.sort` executa em um tempo médio $\mathcal{O}(n \log n)$.

Exercício 2.16

Execute uma análise experimental para determinar o maior valor de n para os algoritmos `unique1` e `unique2` (apresentados nos materiais de aula), de modo que o algoritmo execute em um minuto ou menos.