

EDCO4B

ESTRUTURAS DE DADOS 2

Aula 02 - Bubble Sort

Prof. Rafael G. Mantovani

Licença

Este trabalho está licenciado com uma Licença CC BY-NC-ND 4.0:



maiores informações:

https://creativecommons.org/licenses/by-nc-nd/4.0/deed.pt_BR

Roteiro



- 1** Introdução
- 2** Bubble Sort
- 3** Exemplo
- 4** Exercício
- 5** Referências

Roteiro

- 1** Introdução
- 2** Bubble Sort
- 3** Exemplo
- 4** Exercício
- 5** Referências

Introdução



**Algoritmos de
Ordenação**

Introdução

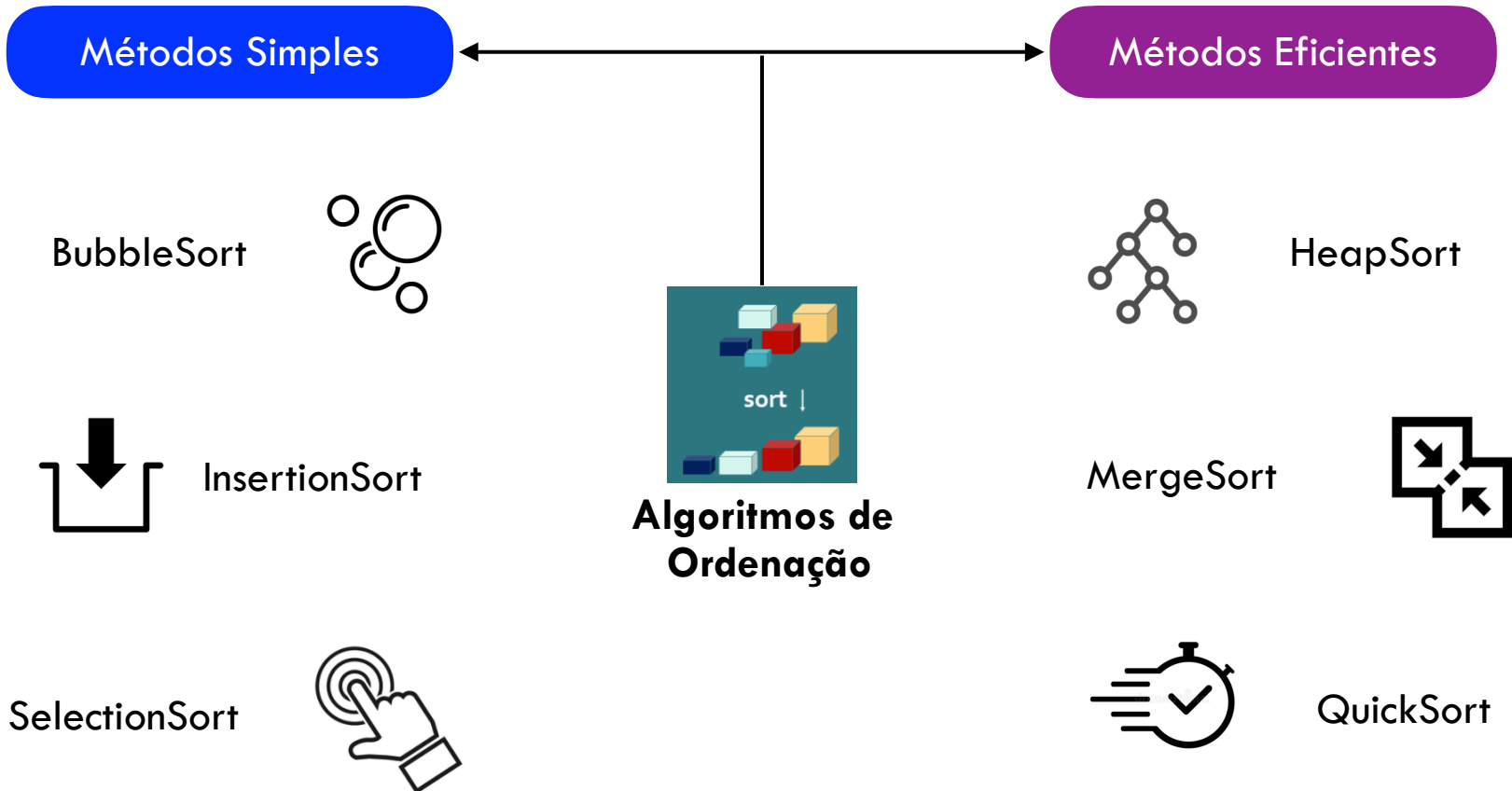
Métodos Simples

Métodos Eficientes

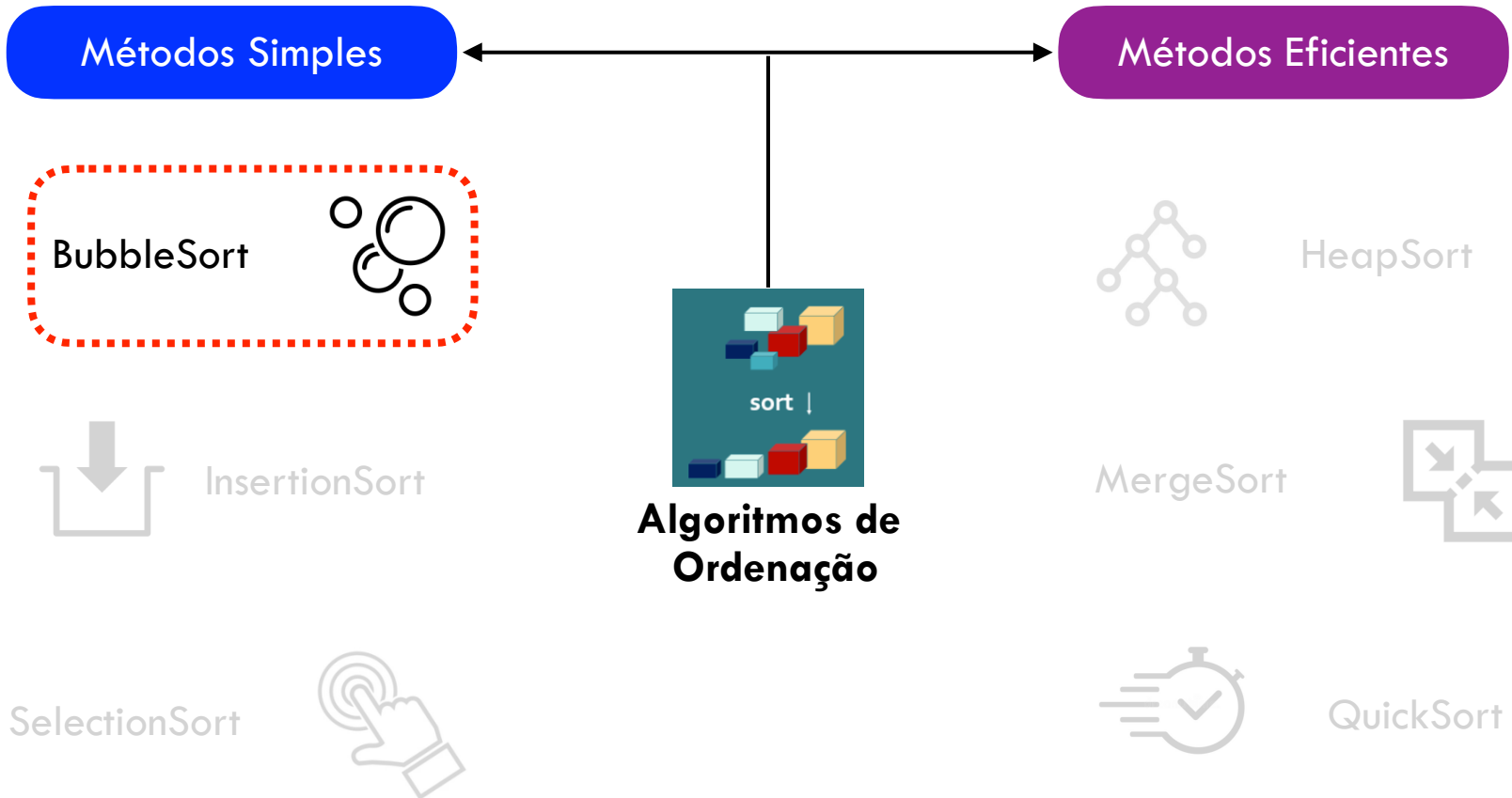


**Algoritmos de
Ordenação**

Introdução



Introdução



Roteiro



- 1 Introdução
- 2 **Bubble Sort**
- 3 Exemplo
- 4 Exercício
- 5 Referências

Bubble Sort

- Ordenação por Bolha / Borbulhamento



Bubble Sort

- Ordenação por Bolha / Borbulhamento



Topo

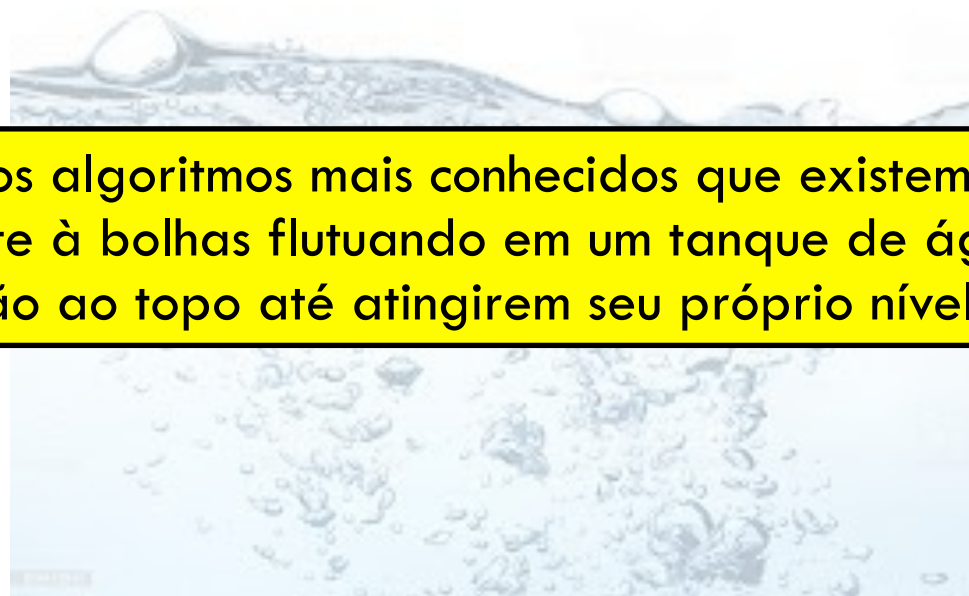


Fundo

Bubble Sort

□ Ordenação por Bolha / Borbulhamento

- * um dos algoritmos mais conhecidos que existem
- * remete à bolhas flutuando em um tanque de água: vão em direção ao topo até atingirem seu próprio nível



Topo

Fundo

Bubble Sort

□ Funcionamento

* **compara pares de elementos adjacentes** em um array e os troca de lugar se estiverem na ordem errada

Bubble Sort

□ Funcionamento

- * **compara pares de elementos adjacentes** em um array e os troca de lugar se estiverem na ordem errada
- * trabalha de forma a movimentar, uma posição por vez, **o maior valor existente na posição não ordenada** de um vetor para sua respectiva posição no vetor ordenado

Bubble Sort

□ Funcionamento

- * **compara pares de elementos adjacentes** em um array e os troca de lugar se estiverem na ordem errada
- * trabalha de forma a movimentar, uma posição por vez, **o maior valor** existente **na posição não ordenada** de um vetor para sua respectiva posição no vetor ordenado
- * processo se **repete** até que mais **nenhuma troca** seja necessária (elementos já estão ordenados)

Bubble Sort

□ Desempenho

- * **melhor caso:** $O(N)$, os elementos já estão ordenados
- * **pior caso:** $O(N^2)$, os elementos estão na ordem decrescente
- * **caso médio:** $O(N^2)$, demais situações

Bubble Sort

□ Desempenho

- * **melhor caso:** $O(N)$, os elementos já estão ordenados
- * **pior caso:** $O(N^2)$, os elementos estão na ordem decrescente
- * **caso médio:** $O(N^2)$, demais situações

Obs: **Não** recomendado para conjuntos de dados grandes!

Bubble Sort

□ Pseudocódigo

```
1. BubbleSort (V, N)
2.   trocou ← TRUE
3.   Enquanto (trocou for TRUE), faça:
4.     trocou ← FALSE
5.     para todo índice i de 0 a N-1
6.       se V[i] é maior que V[i+1] então
7.         troque V[i] com V[i+1]
8.         trocou ← TRUE
```

Roteiro



- 1 Introdução
- 2 Bubble Sort
- 3 Exemplo
- 4 Exercícios
- 5 Referências

Exemplo

23	4	67	-8	90	54	21
----	---	----	----	----	----	----

vetor não ordenado

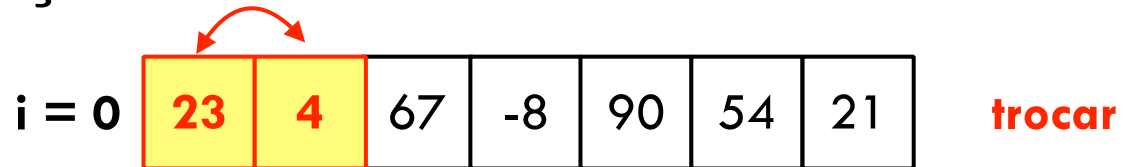
Exemplo

Iteração 01:

$i = 0$	23	4	67	-8	90	54	21	trocar
---------	----	---	----	----	----	----	----	---------------

Exemplo

Iteração 01:



Exemplo

Iteração 01:

i = 0	23	4	67	-8	90	54	21	trocar
i = 1	4	23	67	-8	90	54	21	

Exemplo

Iteração 01:

i = 0	23	4	67	-8	90	54	21	trocar
i = 1	4	23	67	-8	90	54	21	

Exemplo

Iteração 01:

$i = 0$	23	4	67	-8	90	54	21	trocar
$i = 1$	4	23	67	-8	90	54	21	OK
$i = 2$	4	23	67	-8	90	54	21	trocar
$i = 3$	4	23	-8	67	90	54	21	OK
$i = 4$	4	23	-8	67	90	54	21	trocar
$i = 5$	4	23	-8	67	54	90	21	trocar

4	23	-8	67	54	21	90
---	----	----	----	----	----	----

Exemplo

Iteração 01:

i = 0	23	4	67	-8	90	54	21	trocar
i = 1	4	23	67	-8	90	54	21	OK
i = 2	4	23	67	-8	90	54	21	trocar
Tivemos trocas, então seguimos para uma próxima iteração								
i = 4	4	23	-8	67	90	54	21	trocar
i = 5	4	23	-8	67	54	90	21	trocar

4	23	-8	67	54	21	90
---	----	----	----	----	----	----

Final da iteração 01

Exemplo

Iteração 02:

$i = 0$	4	23	-8	67	54	21	90
---------	---	----	----	----	----	----	----

Exemplo

Iteração 02:

$i = 0$	4	23	-8	67	54	21	90	OK
$i = 1$	4	23	-8	67	54	21	90	trocar
$i = 2$	4	-8	23	67	54	21	90	OK
$i = 3$	4	-8	23	67	54	21	90	trocar
$i = 4$	4	-8	23	54	67	21	90	trocar

4	-8	23	54	21	67	90
---	----	----	----	----	----	----

Final da iteração 02

Exemplo

Iteração 02:

i = 0	4	23	-8	67	54	21	90	OK
i = 1	4	23	-8	67	54	21	90	trocar
i = 2	4	-8	23	67	54	21	90	OK
i = 3	4	-8	23	54	67	21	90	trocar
i = 4	4	-8	23	54	67	21	90	trocar

Tivemos trocas, então seguimos para uma próxima iteração

4	-8	23	54	21	67	90
---	----	----	----	----	----	----

Final da iteração 02

Exemplo

Iteração 03:

i = 0	4	-8	23	54	21	67	90
--------------	---	----	----	----	----	----	----

Exemplo

Iteração 03:

$i = 0$	4	-8	23	54	21	67	90	trocar
$i = 1$	-8	4	23	54	21	67	90	OK
$i = 2$	-8	4	23	54	21	67	90	OK
$i = 3$	-8	4	23	54	21	67	90	trocar

-8	4	23	21	54	67	90
----	---	----	----	----	----	----

Final da iteração 03

Exemplo

Iteração 03:

i = 0	4	-8	23	54	21	67	90	trocar
i = 1	-8	4	23	54	21	67	90	OK
i = 2	-8	4	23	54	21	67	90	OK

Tivemos trocas, então seguimos para uma próxima iteração

trocar

-8	4	23	21	54	67	90
----	---	----	----	----	----	----

Final da iteração 03

Exemplo

Iteração 04:

i = 0	-8	4	23	21	54	67	90
--------------	----	---	----	----	----	----	----

Exemplo

Iteração 04:

$i = 0$	-8	4	23	21	54	67	90	OK
$i = 1$	-8	4	23	21	54	67	90	OK
$i = 2$	-8	4	23	21	54	67	90	trocar

-8	4	21	23	54	67	90
----	---	----	----	----	----	----

Final da iteração 04

Exemplo

Iteração 04:

i = 0	-8	4	23	21	54	67	90	OK
i = 1	-8	4	23	21	54	67	90	OK
i = 2	-8	4	23	21	54	67	90	

trocar

Tivemos trocas, então seguimos para uma próxima iteração

-8	4	21	23	54	67	90
----	---	----	----	----	----	----

Final da iteração 04

Exemplo

Iteração 05:

$i = 0$	-8	4	21	23	54	67	90
---------	----	---	----	----	----	----	----

Exemplo

Iteração 05:

i = 0	-8	4	21	23	54	67	90	OK
i = 1	-8	4	21	23	54	67	90	OK

Exemplo

Iteração 05:

i = 0	-8	4	21	23	54	67	90	OK
i = 1	-8	4	21	23	54	67	90	OK

Não tivemos trocas, fim!

-8	4	21	23	54	67	90
----	---	----	----	----	----	----

Vetor Ordenado

Bubble Sort

Vantagens

- * simples e de fácil entendimento e implementação
- * está entre os métodos mais difundidos

Bubble Sort

Vantagens

- * **simples** e de fácil entendimento e implementação
- * está entre os métodos mais difundidos

Desvantagens

- * **não** é um algoritmo **eficiente**
- * Sua eficiência diminui de acordo com o número de elementos
- * é estudado apenas para fins didáticos

Roteiro



- 1 Introdução
 - 2 Bubble Sort
 - 3 Exemplo
 - 4 Exercícios
 - 5 Referências
- 

Exercícios



HANDS ON :)))

Exercícios

1) Reuna-se com seu grupo e execute o teste de mesa (simulação) do algoritmo para as sequências de números apresentadas

Link planilha grupos/sequências de teste:

<https://docs.google.com/spreadsheets/d/1X9IGtcZeAt7j0lIR1W3JPupl2wCtPQgybH0KZm8j-iE/edit?usp=sharing>

Exercícios

2) Implemente o **bubbleSort** em C considerando a seguinte assinatura de função:

/ Ordena o vetor usando BubbleSort*

Parâmetros:

v: vetor a ser ordenado

n: número de elemento do vetor, tamanho do vetor

*Esse algoritmo tem um comportamento assintótico $O(N^2)$ */*

```
void bubbleSort(int *v, int n);
```

Exercícios

Possíveis funções auxiliares:

```
/* Troca os valores de duas variáveis(posições do vetor)  
Parâmetros:  
*a, *b: duas variáveis inteiras, acessíveis por referência. */
```

```
void swap(int *a, int* b);
```

```
/* Gera um vetor com N elementos
```

```
Parâmetros:
```

```
v: vetor a ser ordenado
```

```
n: número de elemento do vetor, tamanho do vetor
```

```
opcao: 0 - crescente, 1 - decrescente, 2 - aleatorio */
```

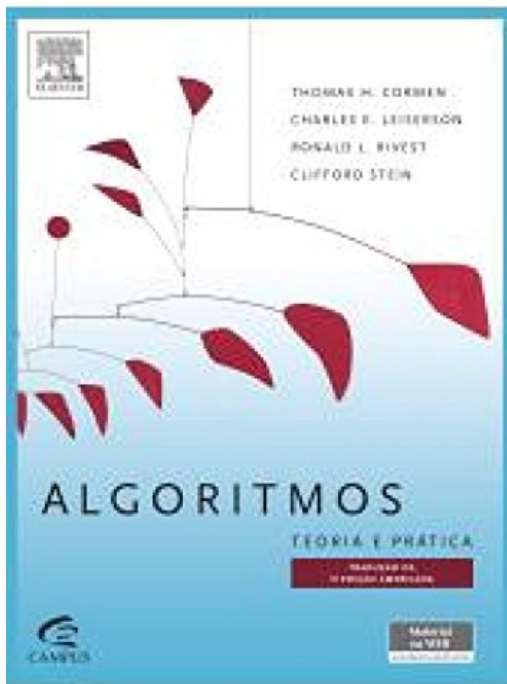
```
void geraVetor(int *v, int n, int opcao);
```

Roteiro



- 1** Introdução
- 2** Bubble Sort
- 3** Exemplo
- 4** Exercícios
- 5** Referências

Referências sugeridas

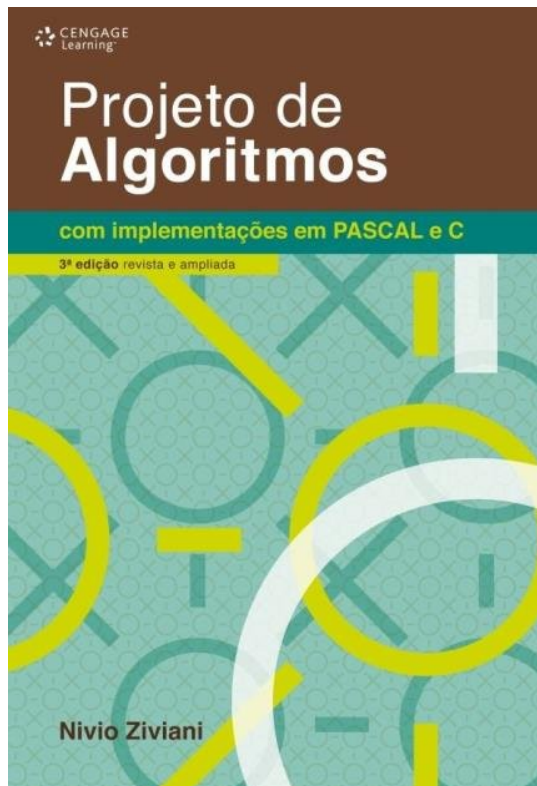


[Cormen et al, 2018]

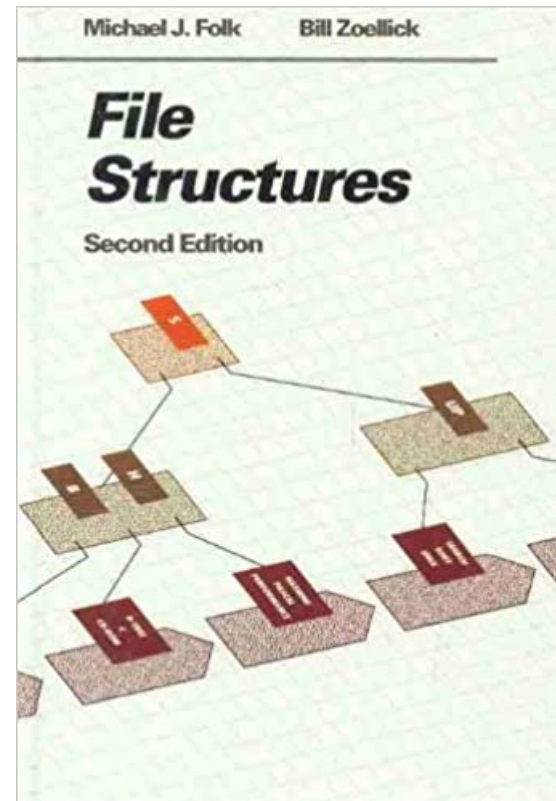


[Drozdek, 2017]

Referências sugeridas



[Ziviani, 2010]



[Folk & Zoellick, 1992]

Perguntas?

Prof. Rafael G. **Mantovani**

rafaelmantovani@utfpr.edu.br