

EDCO4B

ESTRUTURAS DE DADOS 2

Aula 03 - Selection Sort

Prof. Rafael G. Mantovani

Roteiro



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

Roteiro

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

Introdução



**Algoritmos de
Ordenação**

Introdução

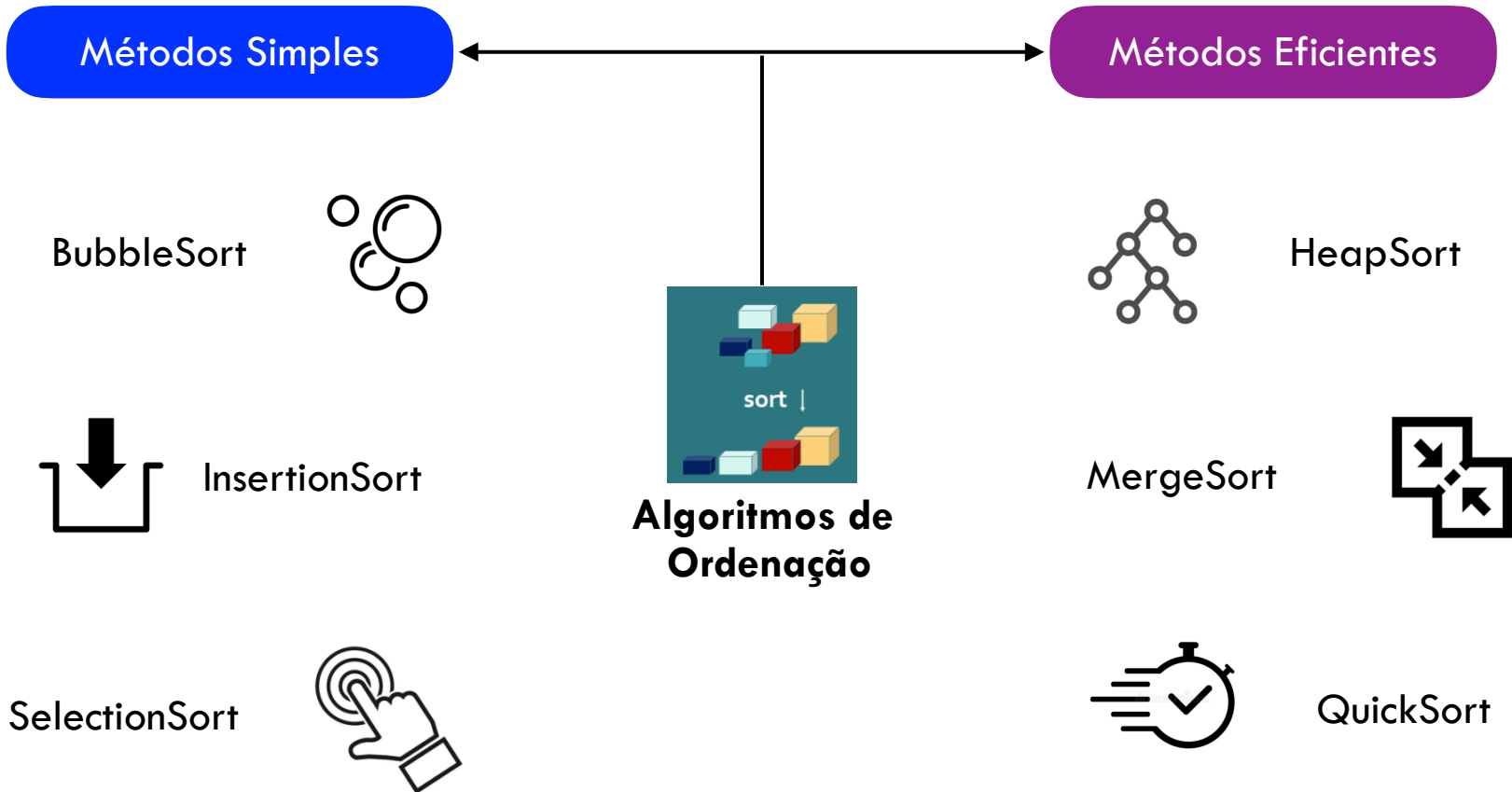
Métodos Simples

Métodos Eficientes

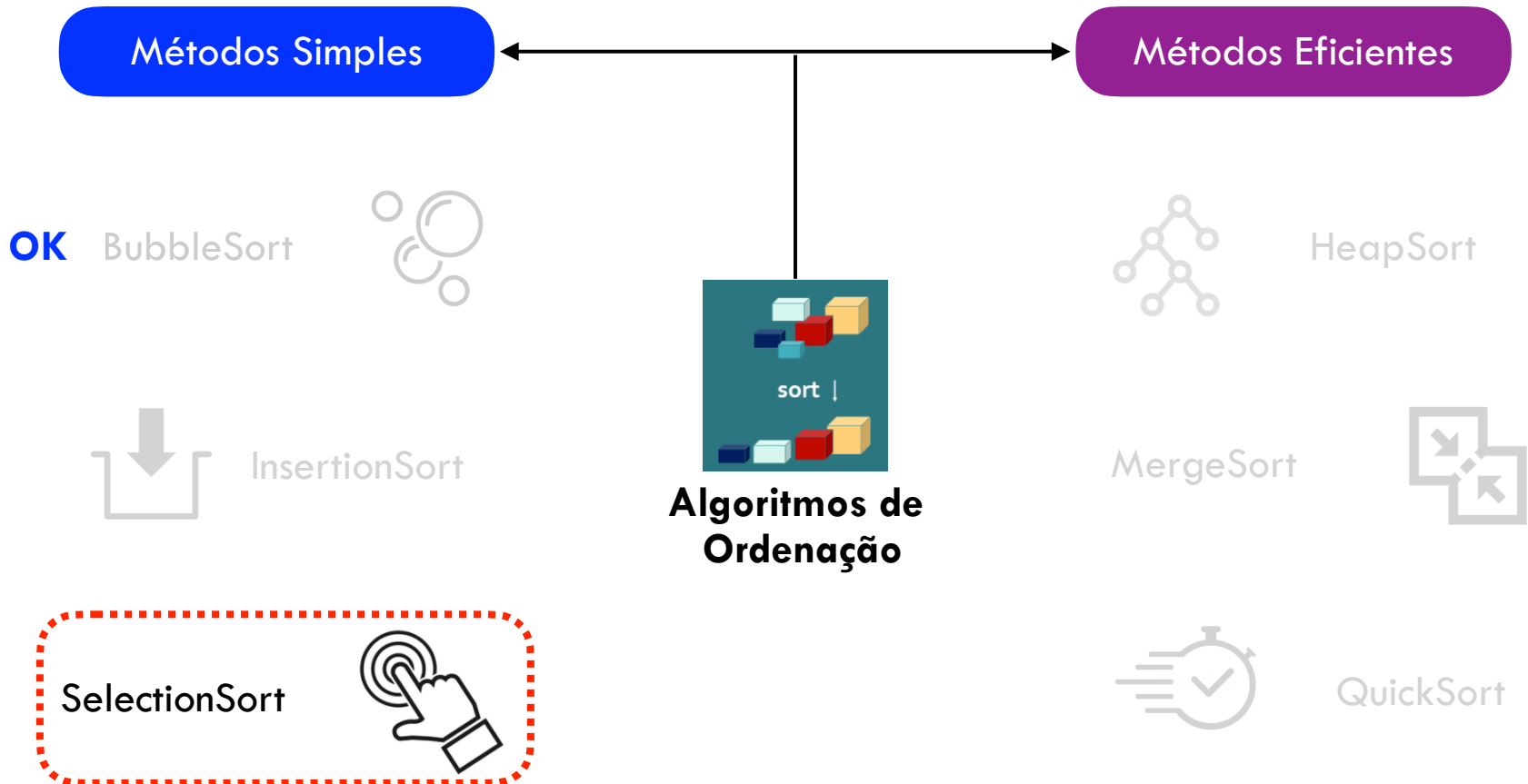


**Algoritmos de
Ordenação**

Introdução



Introdução



Roteiro



- 1 Introdução
- 2 Selection Sort
- 3 Exemplo
- 4 Exercício
- 5 Referências

Selection Sort

□ Ordenação por Seleção



Selection Sort

□ Ordenação por Seleção

Iteração i

Menor



Selection Sort

□ Ordenação por Seleção

Iteração i



BEAST-X
ULTRAZORD MEGAZORD



ASTRO
MEGAZORD



SAMURAI
GIGAZORD



DINO
MEGAZORD



RESCUE
MEGAZORD



SUPERTRAIN
MEGAZORD



RPM
ULTRAZORD

Menor

Selection Sort

□ Ordenação por Seleção

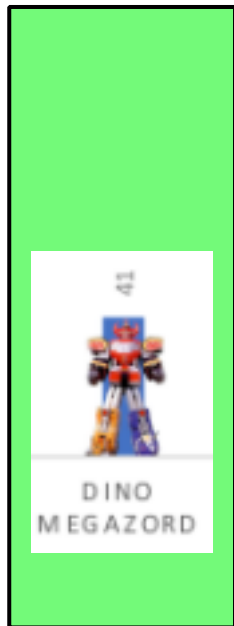


Selection Sort

□ Ordenação por Seleção

Menor

Iteração i



Selection Sort

□ Ordenação por Seleção



Selection Sort

□ Ordenação por Seleção

- * um dos algoritmos mais simples que existem
- * remete à ideia de sempre selecionar o menor elemento da iteração e colocá-lo na posição correta

Selection Sort

□ Funcionamento

- * **a cada passo**, procura o **menor elemento** do array e o **coloca na primeira posição não ordenada**
- * descarta-se a primeira posição do array e repete-se para a segunda em diante
- * repete-se o processo até que **todas as posições** do array estejam **ordenadas**

Selection Sort

□ Desempenho

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

Selection Sort

□ Desempenho

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

Obs: Ineficiente para grandes conjuntos de dados!

Selection Sort

□ Pseudocódigo

1. **SelectionSort** (V, TAM)
2. **Para** cada posição N entre 0 e TAM-1, faça:
3. * encontre o índice i com o menor elemento entre N e TAM-1
4. * **Se** menor valor é diferente do valor da posição N, então:
5. troque V[menor] com V[N]

Roteiro



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

Exemplo

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

vetor não ordenado

Exemplo

Iteração 0:

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

Exemplo

Iteração 0:

It = 0

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

i			menor			
23	4	67	-8	90	54	21

Exemplo

Iteração 0:

It = 0

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

i menor

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

trocar

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

Exemplo

Iteração 1:

It = 1

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

Exemplo

Iteração 1:

It = 1

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

i, menor

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

OK

Exemplo

Iteração 1:

It = 1

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

i, menor

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

OK

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

Exemplo

Iteração 2:

It = 2	-8	4	67	23	90	54	21
---------------	----	---	----	----	----	----	----

Exemplo

Iteração 2:

It = 2

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

		i					menor
-8	4	67	23	90	54	21	trocar

Exemplo

Iteração 2:

It = 2

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

		i					menor
-8	4	67	23	90	54	21	trocar

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

Exemplo

Iteração 3:

It = 3

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

Exemplo

Iteração 3:

It = 3

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

i, menor

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

OK

Exemplo

Iteração 3:

It = 3

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

i, menor

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

OK

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

Exemplo

Iteração 4:

It = 4

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

Exemplo

Iteração 4:

It = 4

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

i menor

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

trocar

Exemplo

Iteração 4:

It = 4

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

i menor

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

trocar

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

Exemplo

Iteração 5:

It = 5

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

Exemplo

Iteração 5:

It = 5

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

i menor

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

trocar

Exemplo

Iteração 5:


It = 5

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

i menor

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

trocar



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

Exemplo

Final:

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

Vetor Ordenado

Selection Sort

Vantagens

- * simples e de fácil entendimento e implementação
- * não altera a ordem dos dados (estável)
- * melhor que bubble sort, menor número de comparações

Selection Sort

Vantagens

- * simples e de fácil entendimento e implementação
- * não altera a ordem dos dados (estável)
- * melhor que bubble sort, menor número de comparações

Desvantagens

- * sua eficiência diminui de acordo com o número de elementos
- * não é recomendado para aplicações com grandes quantidades de dados ou que precisem de velocidade

Roteiro



- 1** Introdução
- 2** Selection 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 **Selection Sort** para a sua sequência de números aleatórios, definida na planilha de grupos da disciplina.

Exercícios

2) Implemente o **selectionSort** em **Python** considerando a seguinte assinatura de função:

```
/* Ordena o vetor usando Selection Sort
```

```
Parâmetros:
```

```
array: vetor a ser ordenado
```

```
option: 1 - ordenação crescente, 2 - ordenação decrescente
```

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

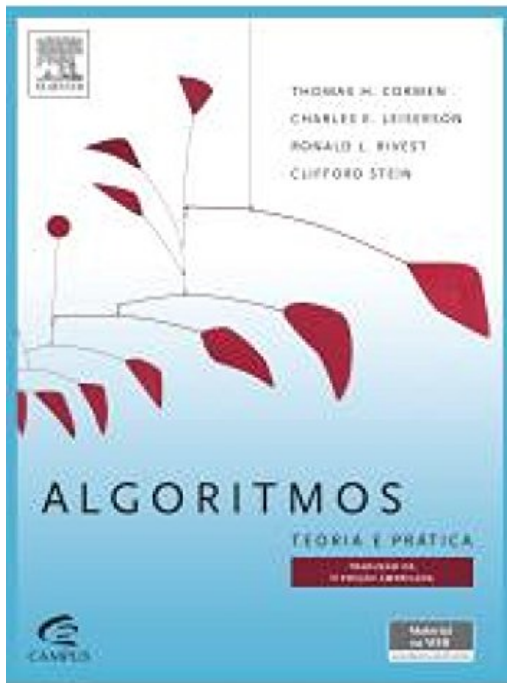
```
def selectionSort(array, option):
```

Roteiro



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

Referências sugeridas

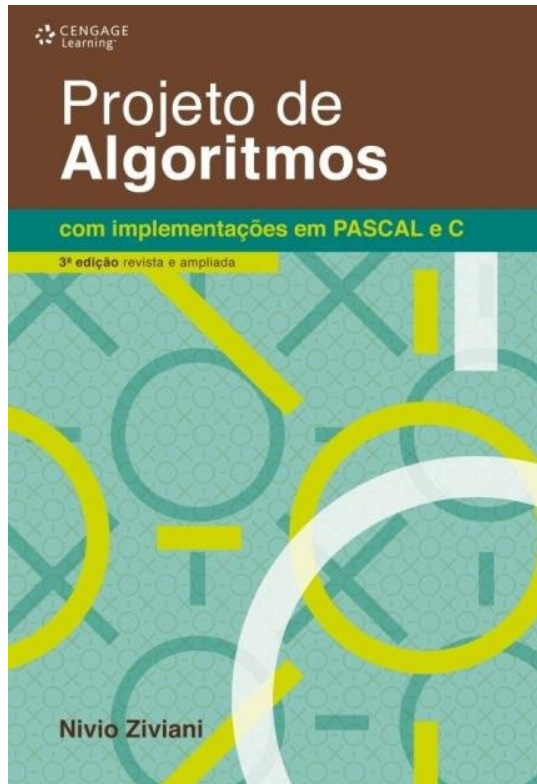


[Cormen et al, 2018]

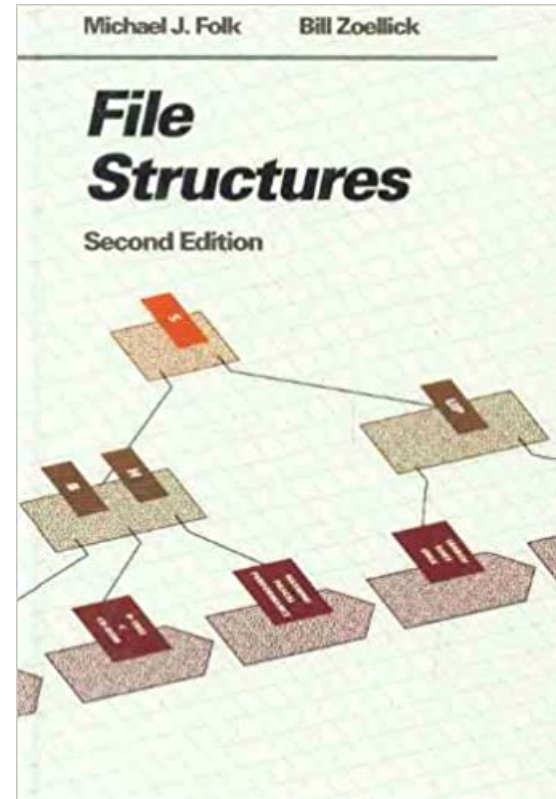


[Drozdek, 2017]

Referências sugeridas



[Ziviani, 2010]



[Folk & Zoellick, 1992]

Perguntas?

Prof. Rafael G. **Mantovani**

rafaelmantovani@utfpr.edu.br