Algoritmo e Estrutura de Dados II
COM-112

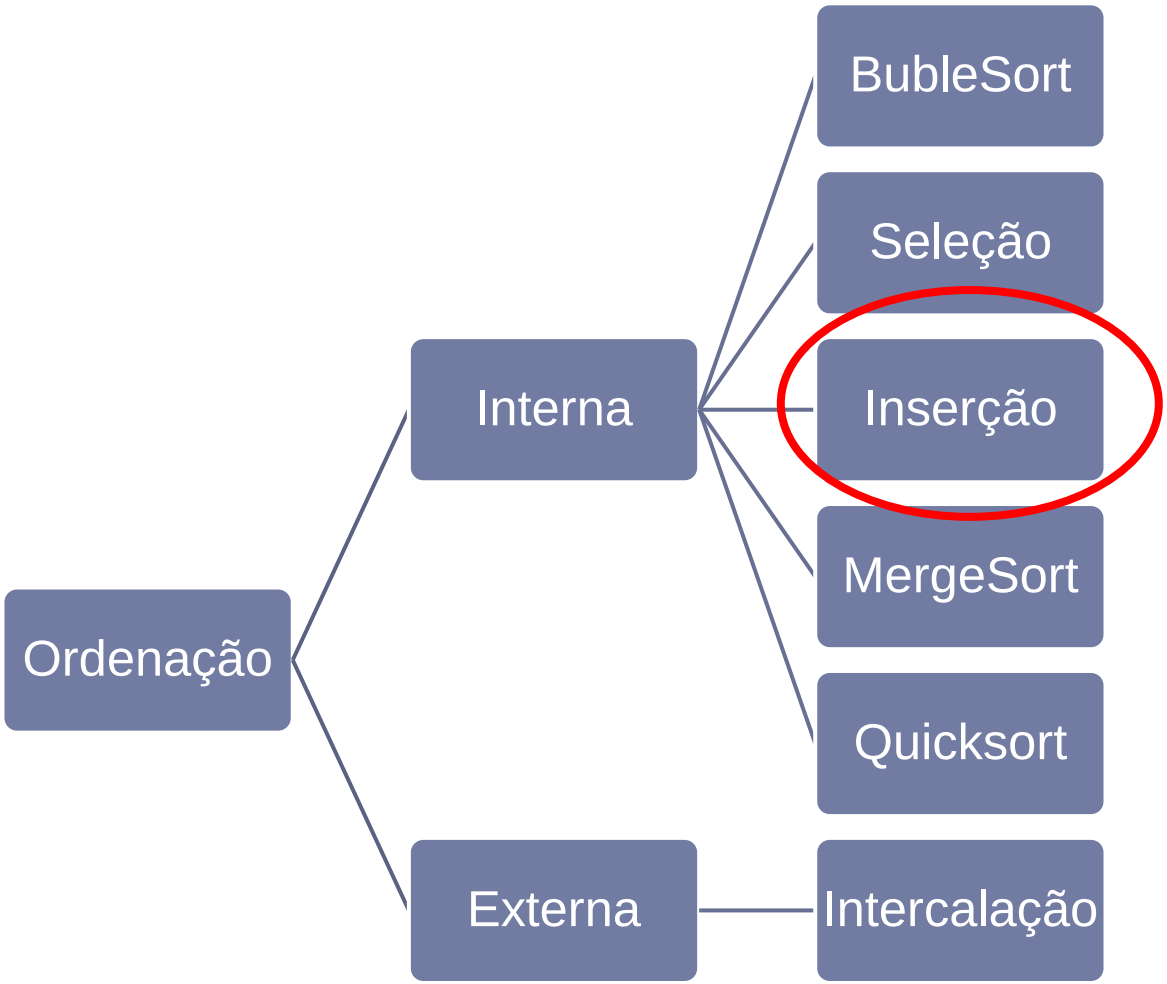
Inserção

Vanessa Souza

Universidade Federal de Itajubá – Instituto de Matemática e Computação

Ordenação

Classificação dos Métodos de Ordenação

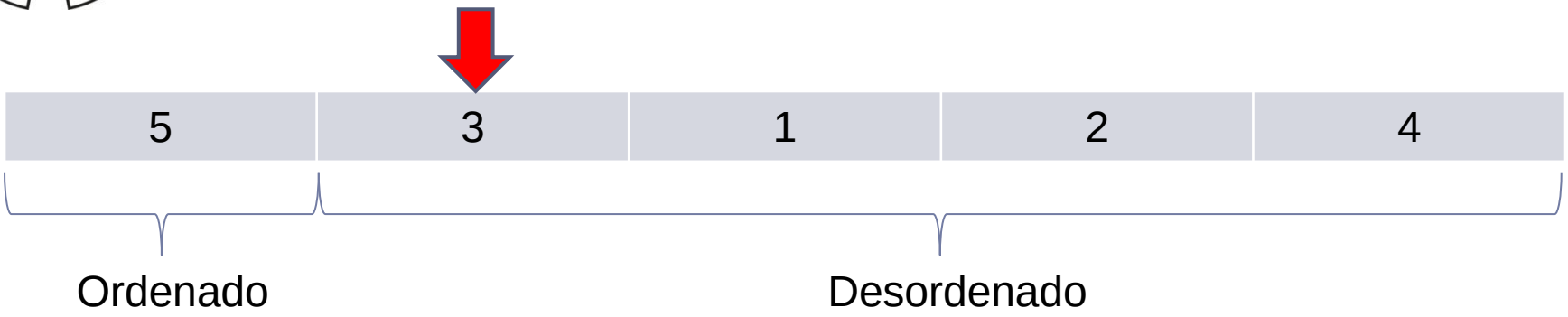


Ordenação por Inserção

InsertionSort

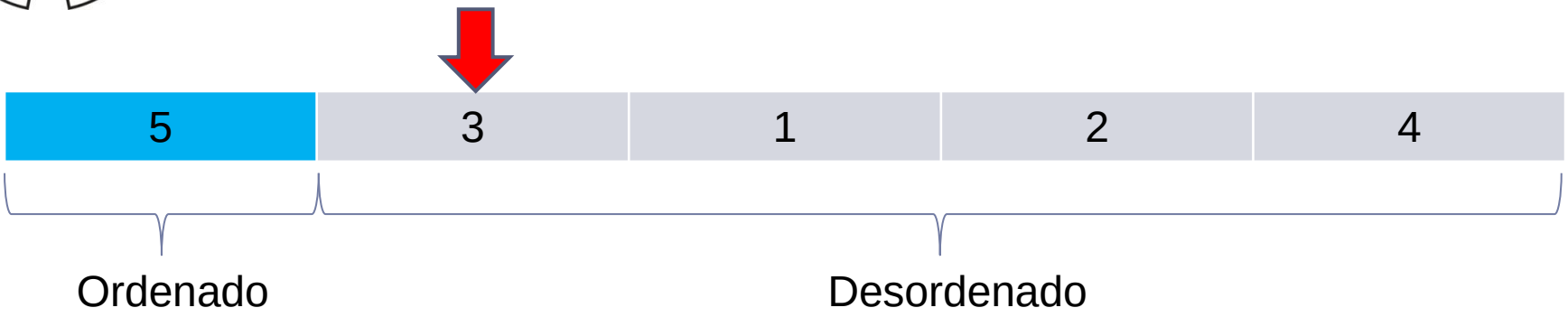
- Ideia: Também usa marcador mas, inicialmente, marcador = 1. Seja x o primeiro elemento da parte desordenada. Troca-se x de posição com os elementos que aparecem à esquerda até que x esteja em sua posição correta, e avança-se o marcador.
  - O processo se repete até que a parte desordenada do vetor esteja vazia.
  - Insere vet[marcador] na posição correta da parte ordenada do vetor

InsertionSort



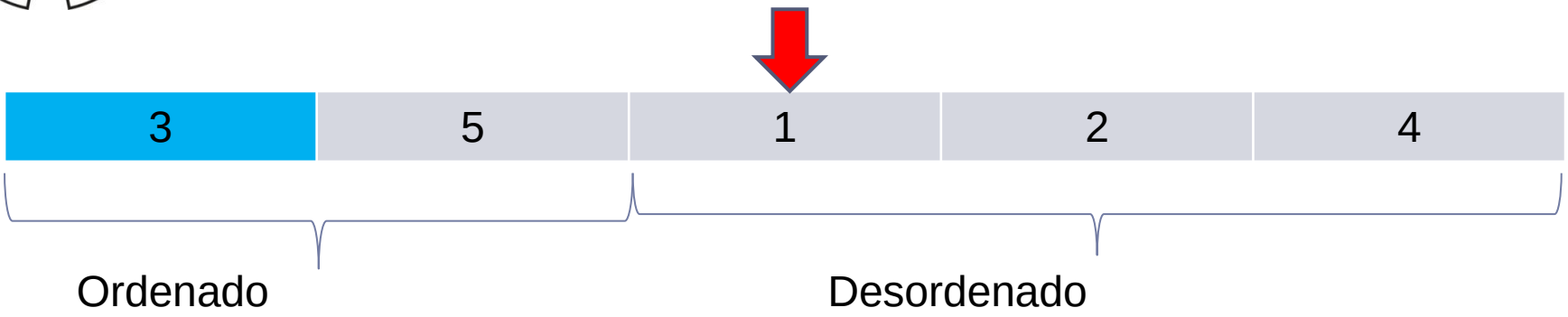
Procura na parte ordenada do vetor, o local correto de 3

InsertionSort



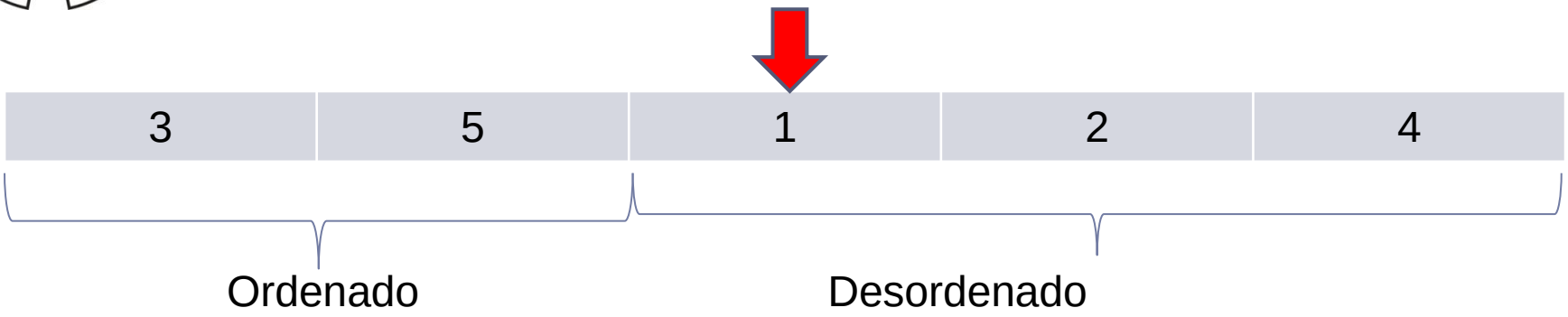
Procura na parte ordenada do vetor, o local correto de 3

InsertionSort



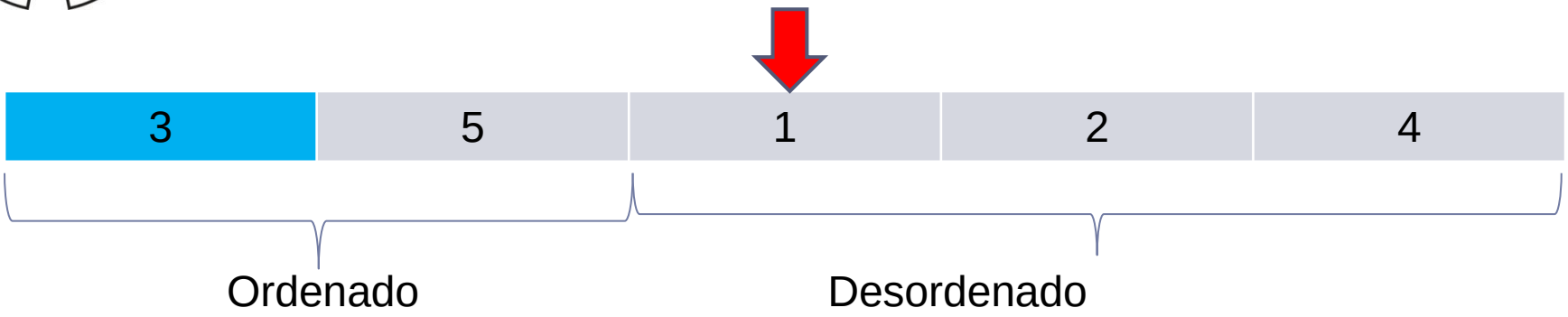
Insere o 3 no lugar do 5
Anda com o marcador

InsertionSort



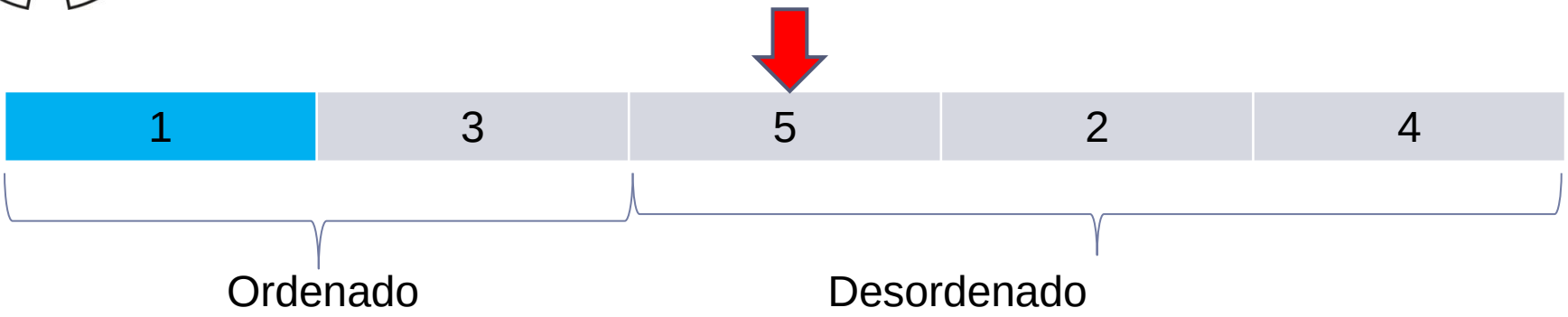
Procura na parte ordenada do vetor, o local correto de 1

InsertionSort



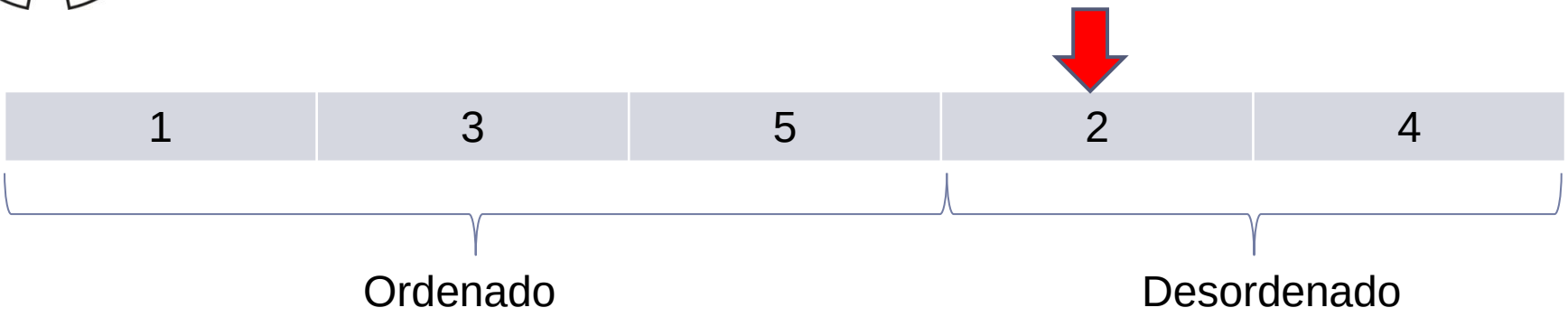
Local correto de 1 : posição 0

InsertionSort



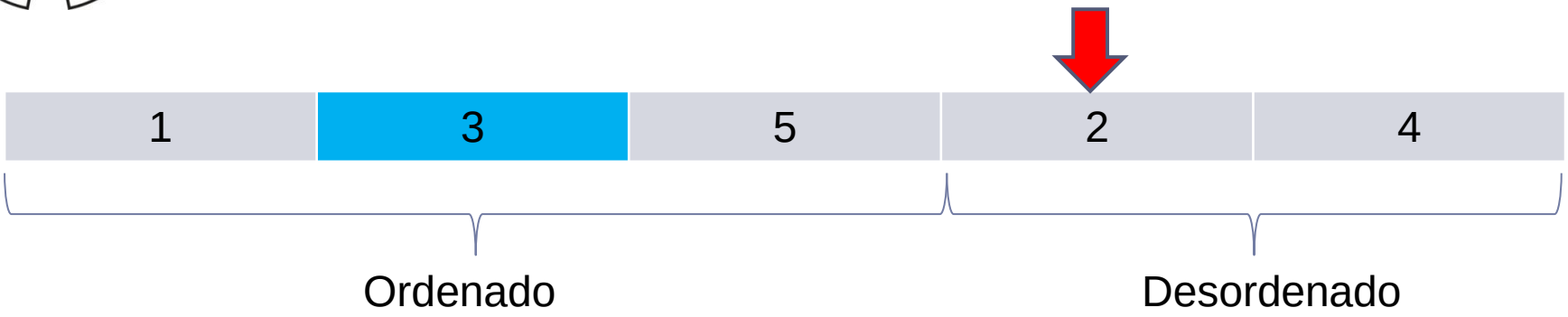
Insere o 1 no seu local correto, movimentando as demais posições

InsertionSort



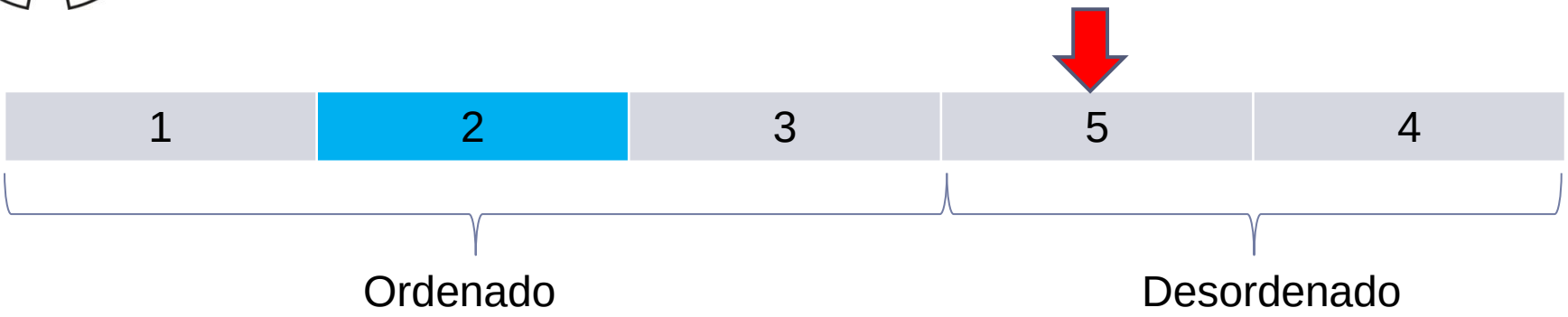
Repete o processo

InsertionSort



Repete o processo

InsertionSort



Repete o processo

InsertionSort

Repete o processo

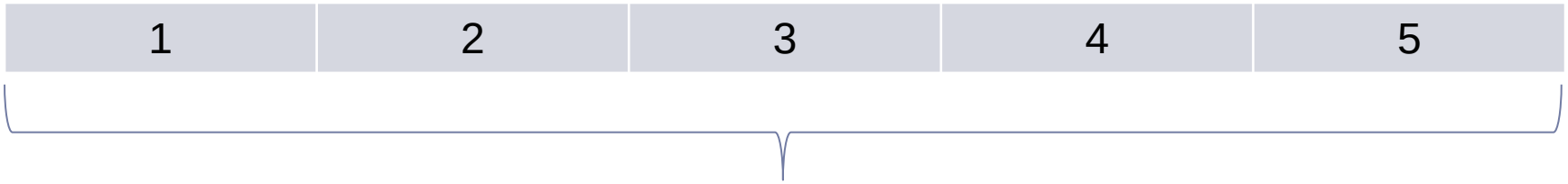
InsertionSort

Repete o processo

InsertionSort

Repete o processo

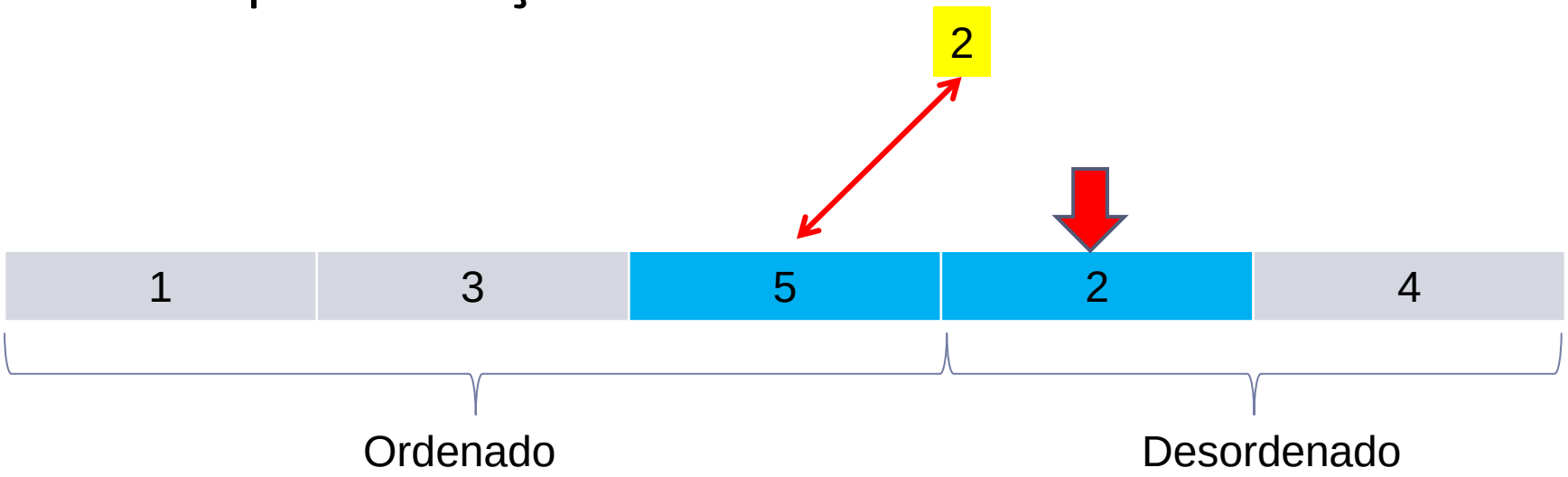
InsertionSort



Vetor ordenado

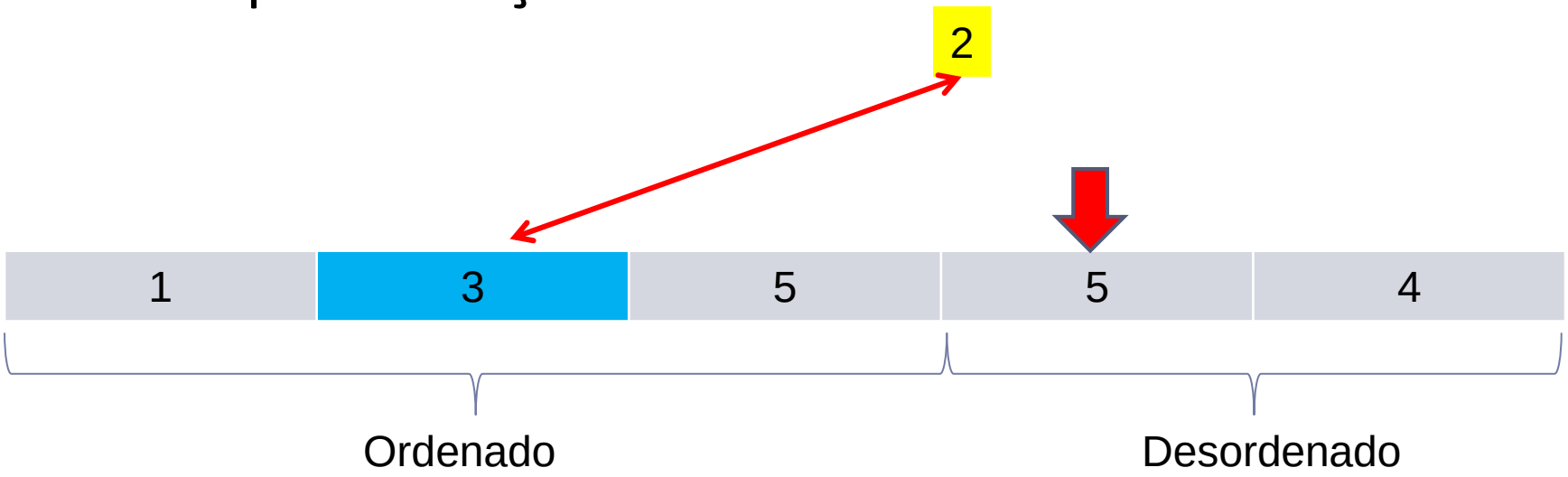
InsertionSort

- Na implementação...



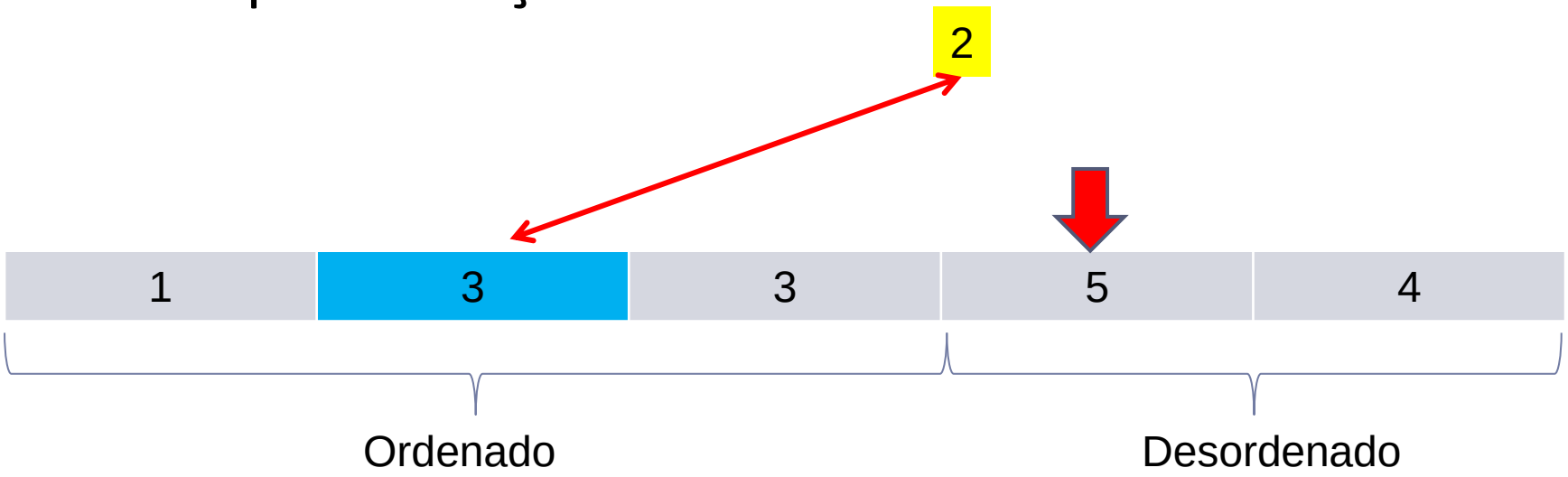
InsertionSort

- Na implementação...



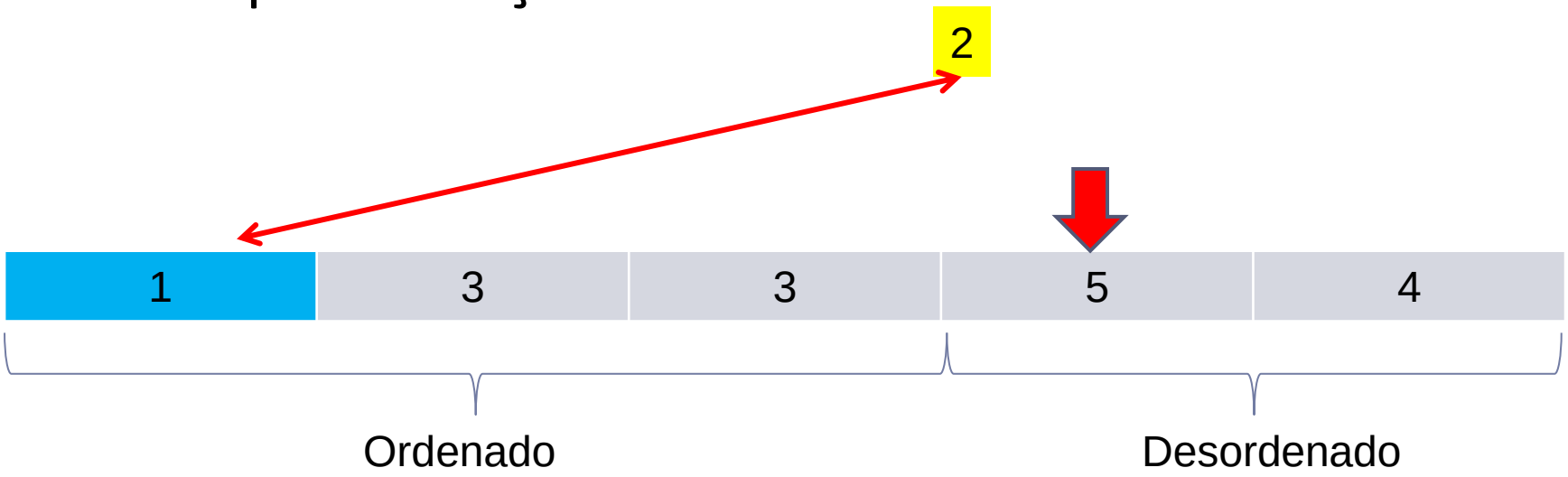
InsertionSort

- Na implementação...



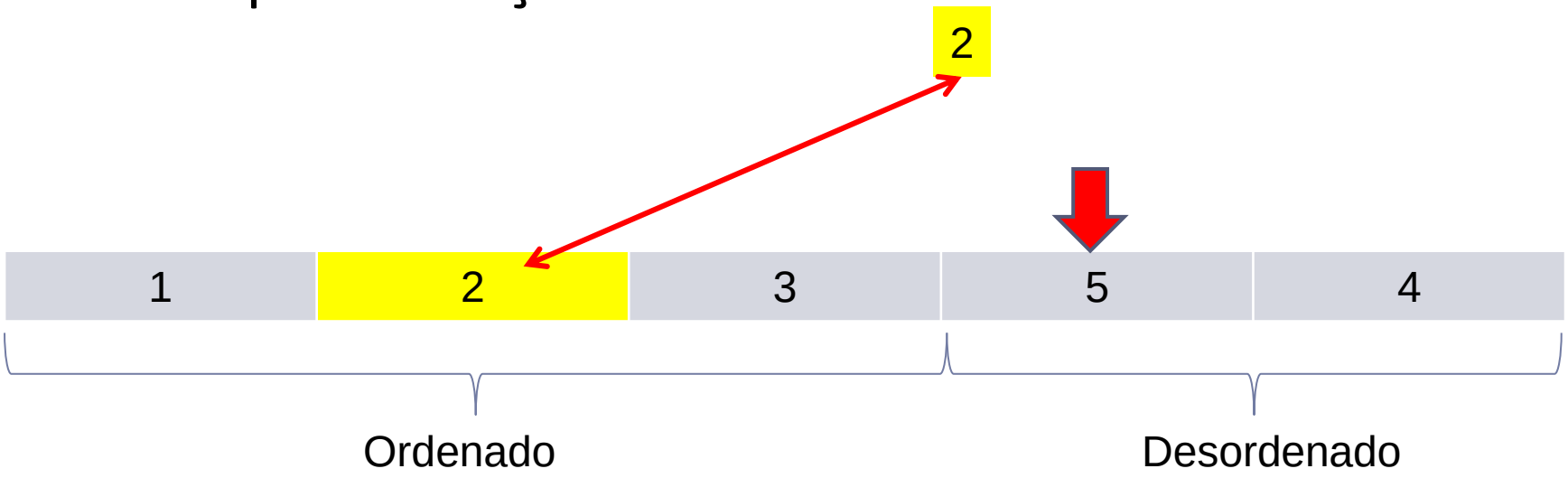
InsertionSort

- Na implementação...



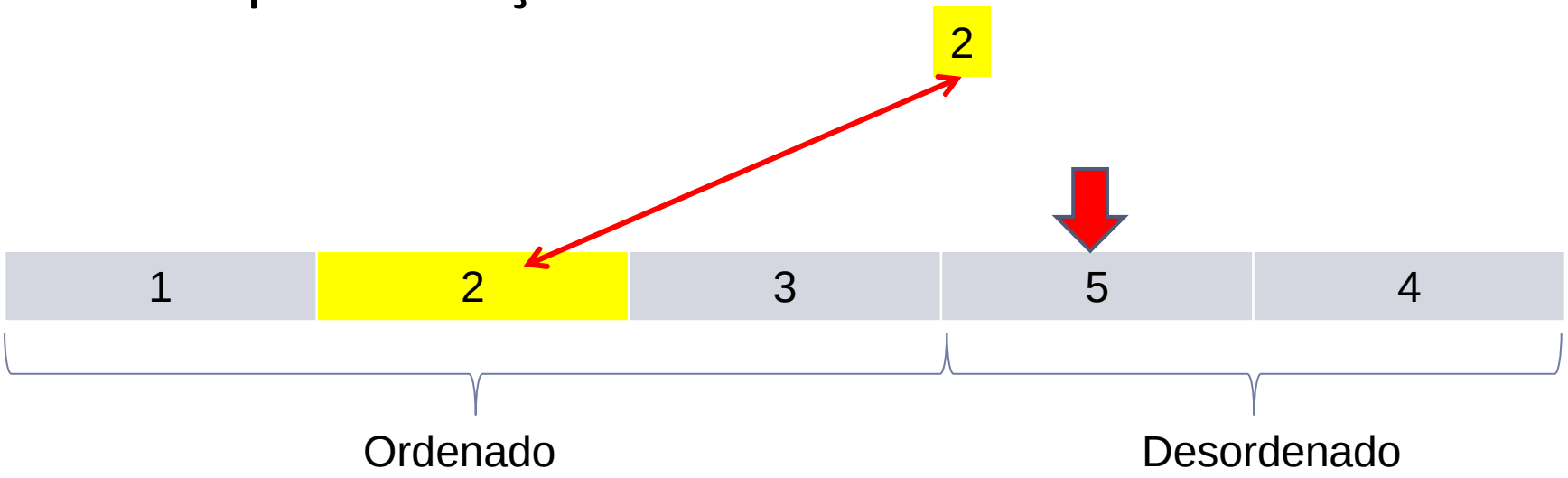
InsertionSort

- Na implementação...



InsertionSort

- Na implementação...



Condição de parada : enquanto pos >= 0 e enquanto aux < vet[pos]
pos : variável inicializada em marcador – 1;
aux : variável que guarda vet[marcador]

Exercício

- Usando o algoritmo de ordenação por **inserção**, ordene o vetor abaixo:
- 12, 43, 1, 6, 56, 23, 52, 9

InsertionSort

- Complexidade assintótica do algoritmo de Inserção
  - Qual o pior caso?

InsertionSort

- Vamos programar?

Algorithm 1 Ordenação por Inserção

```
procedure INSERCAO(V, tamVet)                    ▷ V é um vetor numérico
    for (marcador = 1; marcador < tamVet) do
        pos ← marcador − 1
        aux ← V[marcador]

        ...Inserindo V[marcador] na parte ordenada...
        while (pos ≥ 0)E(aux < V[pos]) do
            V[pos + 1] ← V[pos]
            pos ← pos − 1
        end while
        V[pos + 1] ← aux

    end for
end procedure
```

Comparação entre os métodos

- Os algoritmos vistos até agora são muito citados por sua simplicidade.
- Todos possuem complexidade assintótica do pior caso de $O(n^2)$.
- Costumam ser bons para arquivos pequenos e já quase ordenados.

Comparação entre os métodos

Algoritmo	Melhor Caso*	Caso Médio	Pior Caso	Estratégia
Bolha	$O(n^2)$	$O(n^2)$	$O(n^2)$	Maior elemento na última posição
Bolha Inteligente	$O(n)$	$O(n^2)$	$O(n^2)$	
Seleção	$O(n^2)$	$O(n^2)$	$O(n^2)$	Menor elemento na primeira posição
Inserção	$O(n)$	$O(n^2)$	$O(n^2)$	Cada elemento em sua posição correta na parte ordenada do vetor

* vetor ordenado

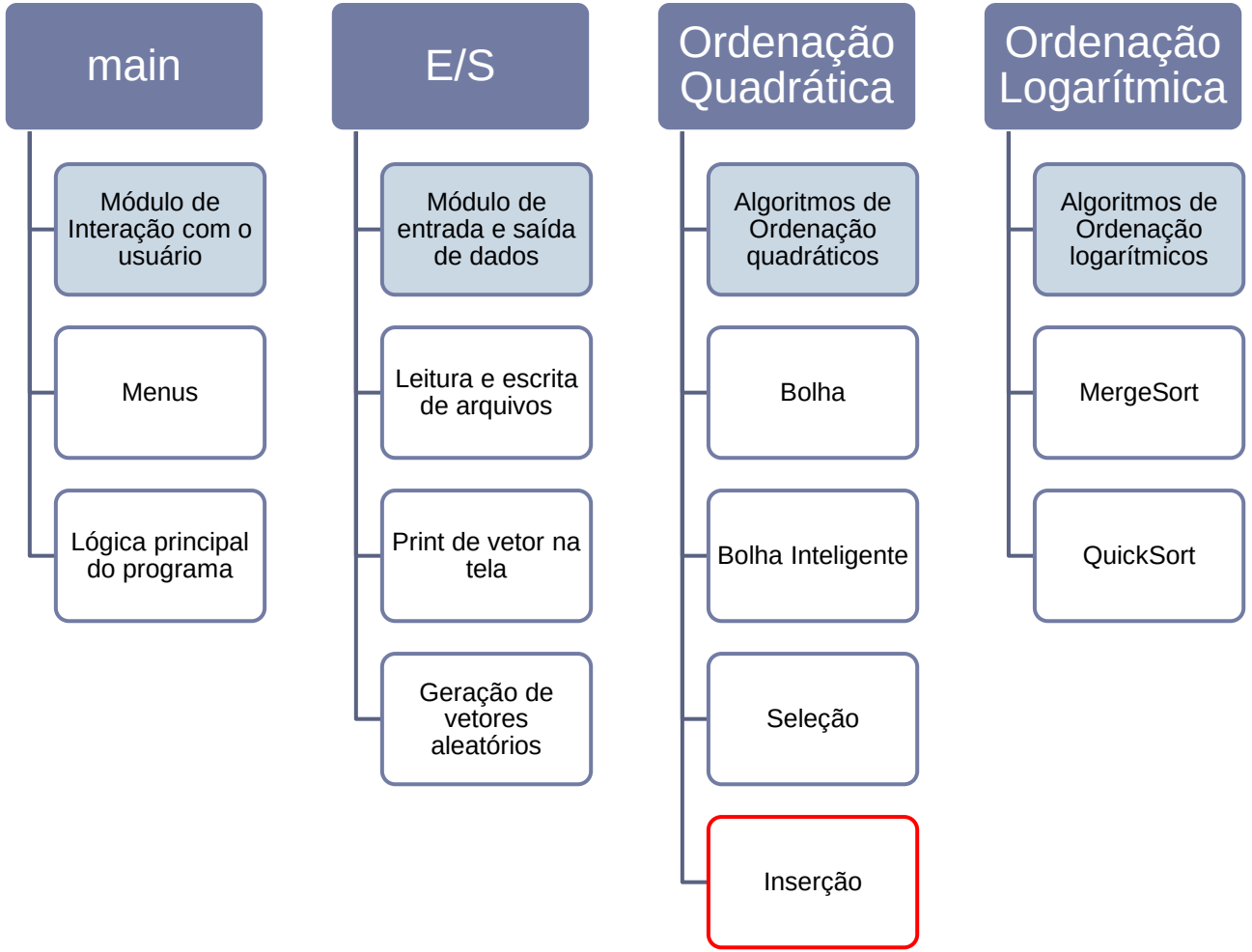
Comparação entre os métodos

- Existe uma outra gama de algoritmos de ordenação mais eficientes ($O(\log_2 n)$).
  - mergeSort
  - quickSort

- Esses algoritmos baseiam-se na estratégia de DIVIDIR PARA CONQUISTAR

- Implementações mais difíceis – estritamente recursivos

Modularização – Algoritmos de Ordenação



Comparação entre os métodos

- Acrescentar contagem para cada algoritmo
  - Movimentações
  - Comparações
- Medição de tempo