INSTITUTO METRÓPOLE DIGITAL

UFRN UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE

Estrutura de Dados Básicas I

Aula 8 – Algoritmos de ordenação II

Prof. Eiji Adachi M. Barbosa

Objetivos

- Apresentar algoritmo de ordenação por inserção

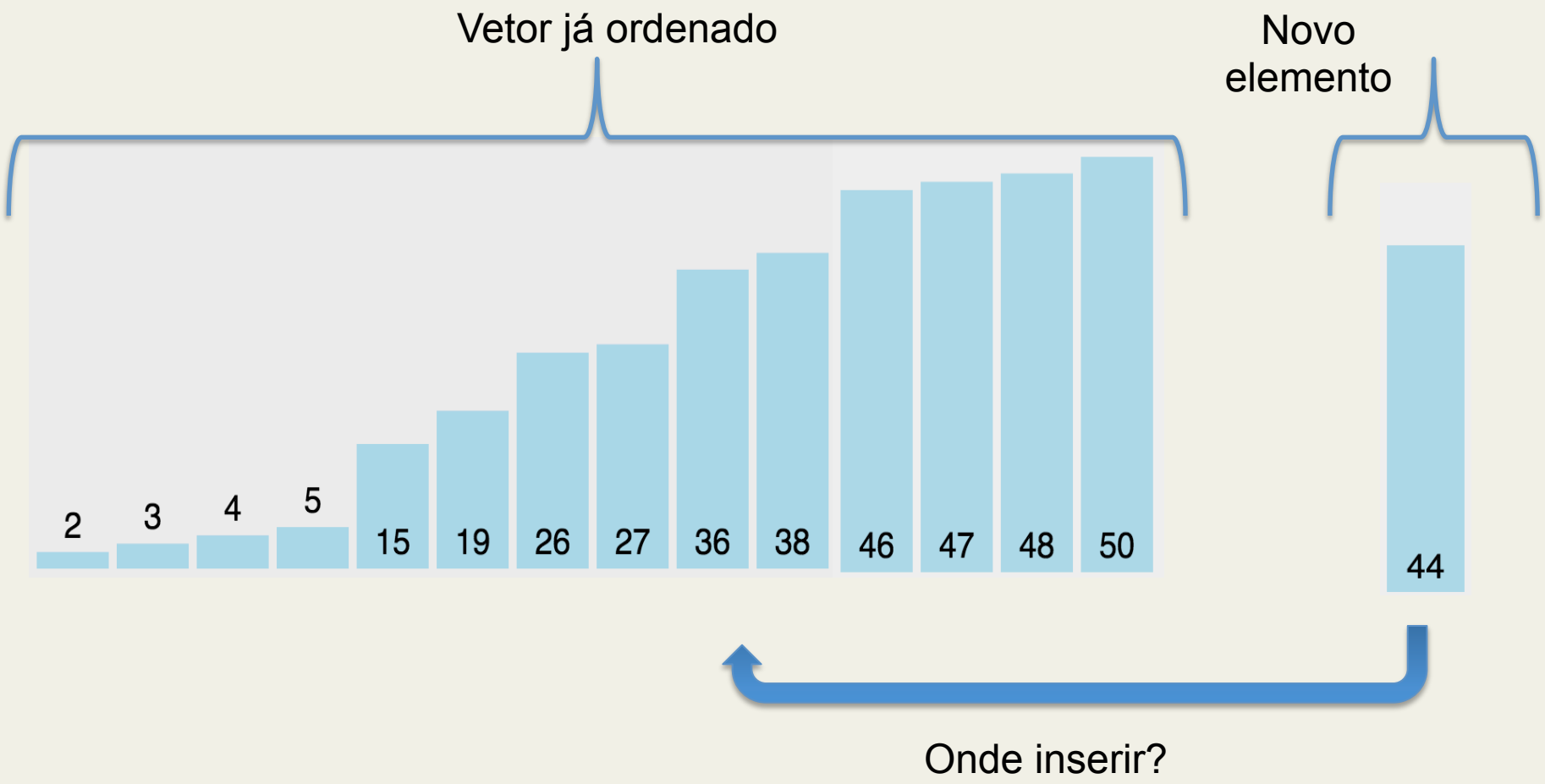
- Implementar algoritmo de ordenação por inserção

Referência extra para esta aula

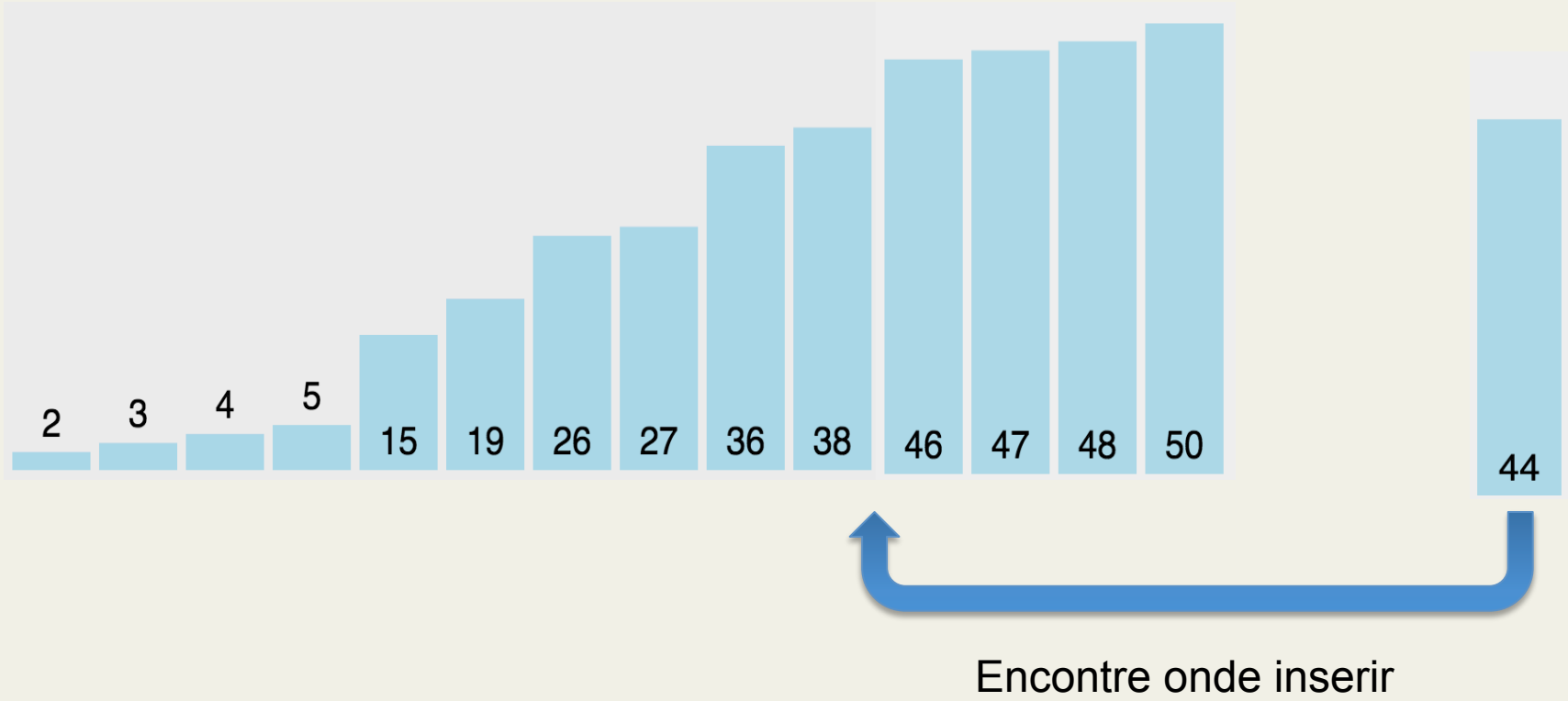
- Notas de aula “Análise de algoritmos”, prof. Paulo Feofiloff, IME-USP:
  - http://www.ime.usp.br/~pf/algoritmos/aulas/ordena.html

ORDENAÇÃO POR INSERÇÃO

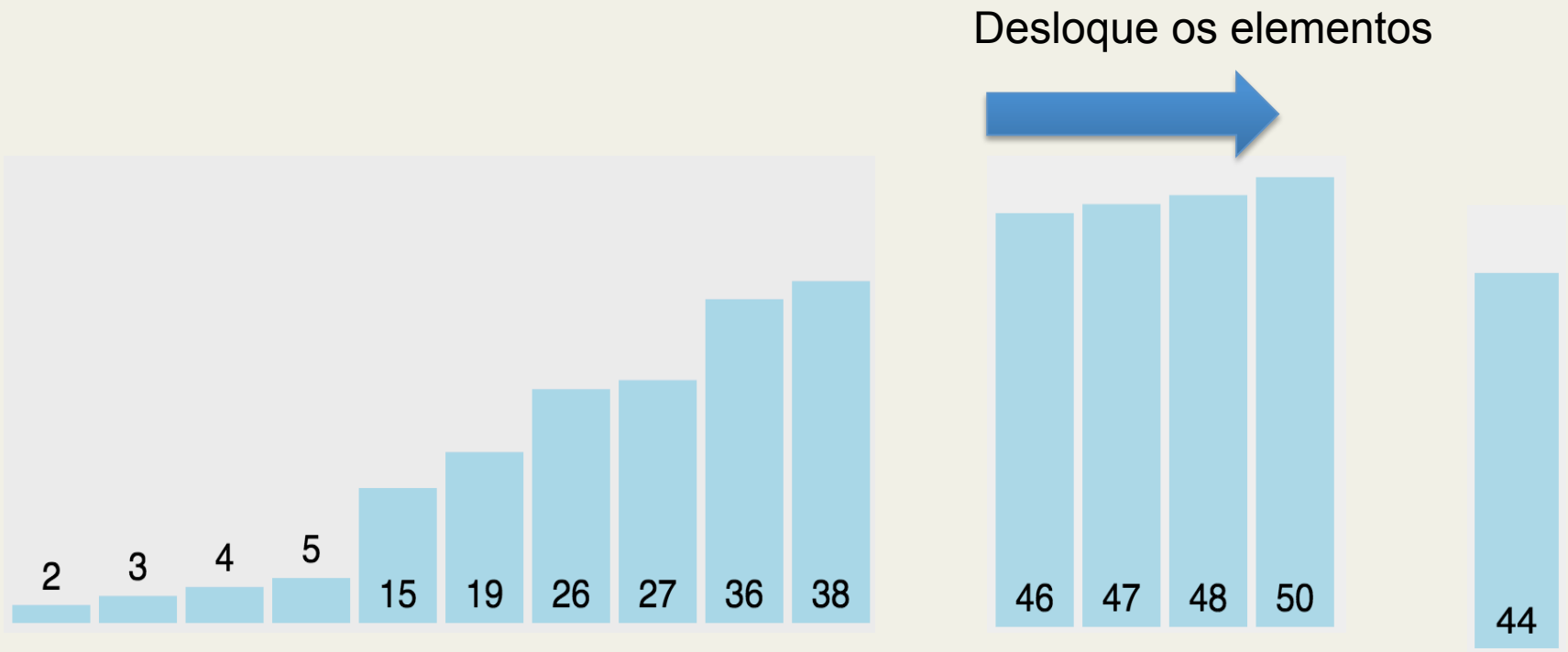
Ideia geral


Vetor já ordenado
Novo elemento
2
3
4
5
15
19
26
27
36
38
46
47
48
50
44
Onde inserir?

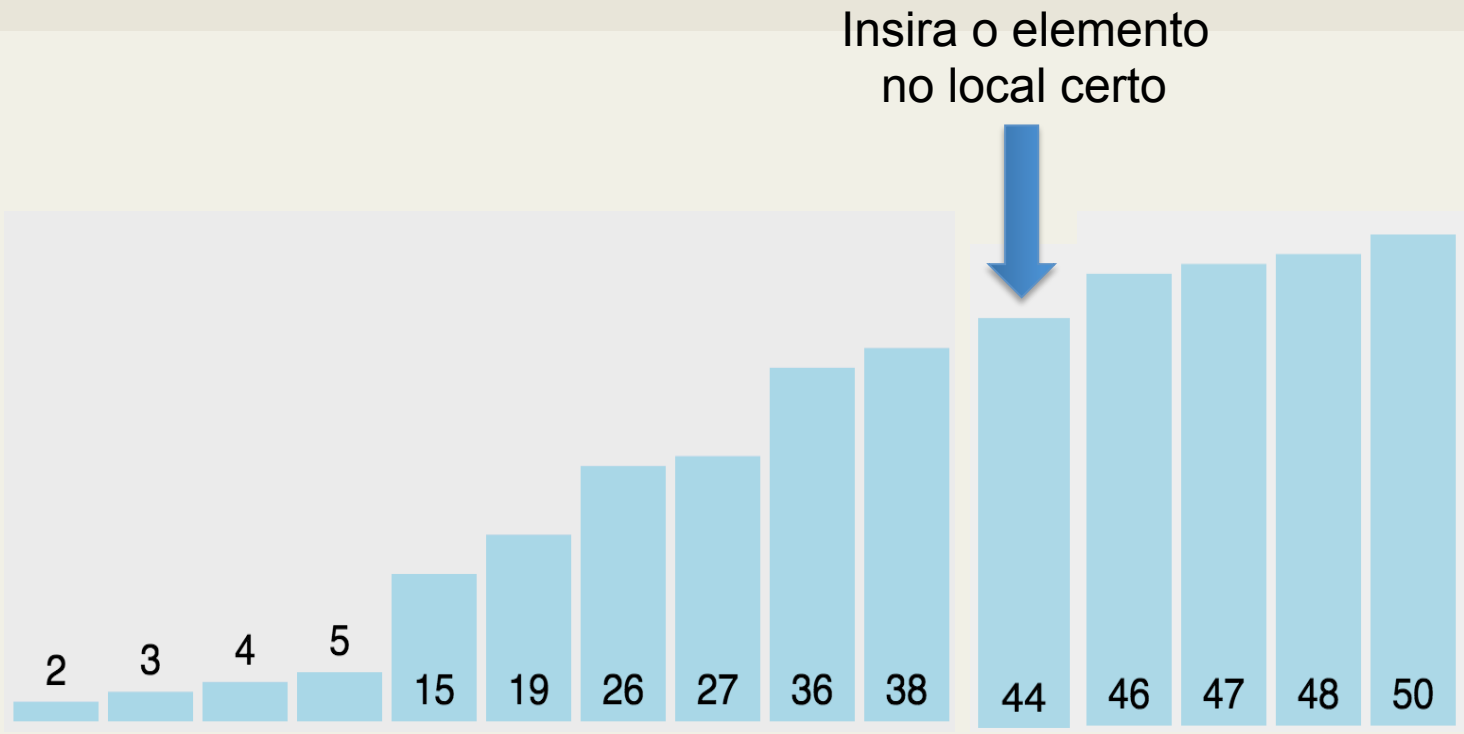
Ideia geral



Ideia geral

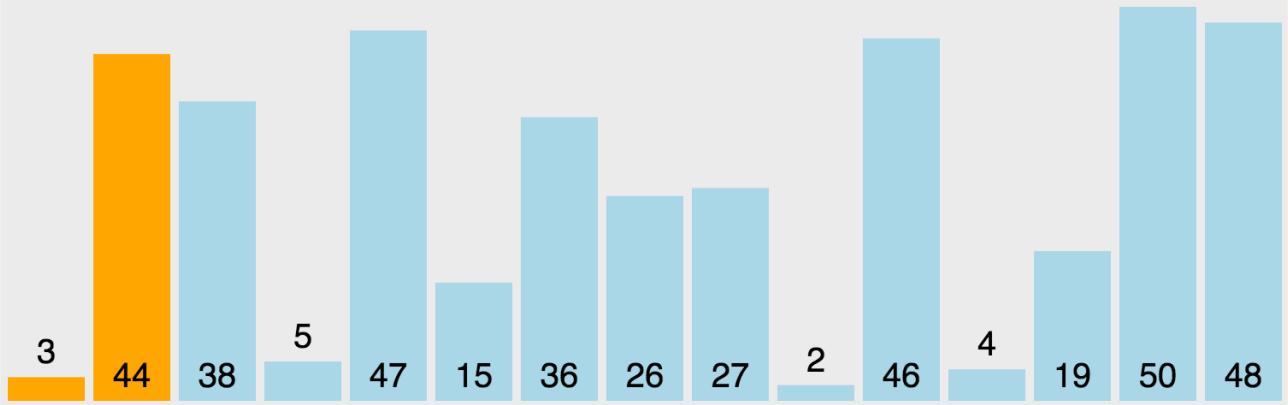


Ideia geral


Insira o elemento
no local certo
2
3
4
5
15
19
26
27
36
38
44
46
47
48
50

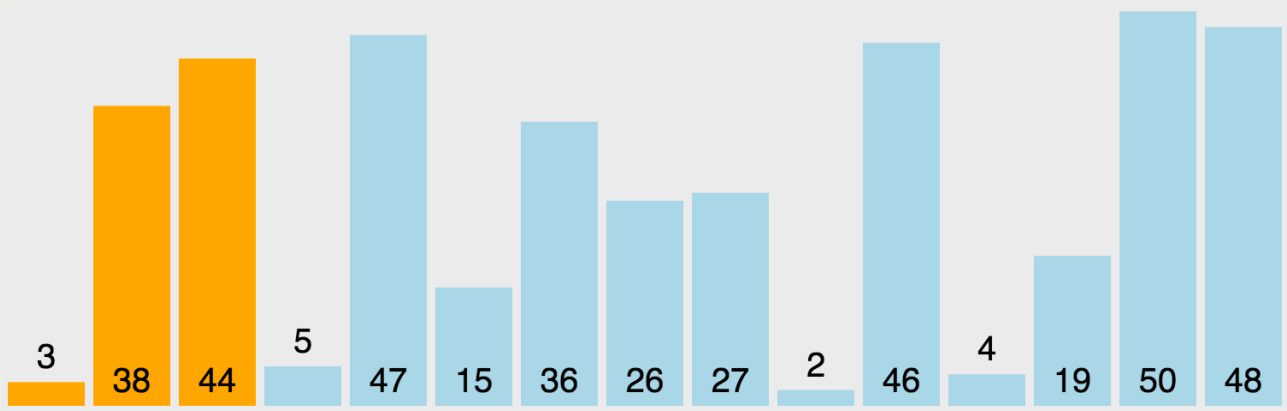
Ideia geral

- Dado um vetor v[N], ele terá uma parte ordenada e outro não ordenada
  - Insira primeiro elemento da parte não ordenada na parte ordenada



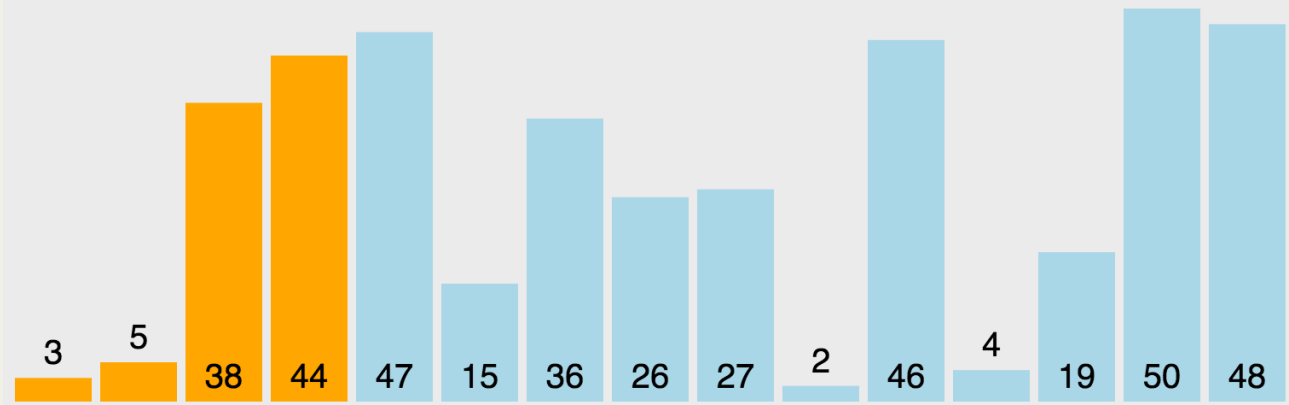
Ideia geral

- Dado um vetor v[N], ele terá uma parte ordenada e outro não ordenada
  - Insira primeiro elemento da parte não ordenada na parte ordenada



Ideia geral

- Dado um vetor v[N], ele terá uma parte ordenada e outro não ordenada
  - Insira primeiro elemento da parte não ordenada na parte ordenada



Ordenação por inserção

- Visualizar funcionamento em:
  - http://visualgo.net/sorting.html#

Ordenação por inserção

- Algoritmo:

```
Sort( v[N] ):
   FOR i = 1; i < N; i++:
        novo = v[i] // 'novo' é o primeiro não ordenado
        j = i-1 // 'j' é o fim da parte ordenada
        WHILE j ≥ 0 && novo < v[j] :
               v[j+1] = v[j] // desloca elementos
               j = j-1
        END_WHILE
        v[j+1] = novo // insere novo elemento no local
   END_FOR
END
```

Qual a complexidade (pior e melhor caso)?

Ordenação por inserção

- Algoritmo:

```
Sort( v[N] ):
   FOR i = 1; i < N; i++:
        novo = v[i] // 'novo' é o primeiro não ordenado
        j = i-1 // 'j' é o fim da parte ordenada
        WHILE j ≥ 0 && novo < v[j] :
               v[j+1] = v[j] // desloca elementos
               j = j-1
        END_WHILE
        v[j+1] = novo // insere novo elemento no local
   END_FOR
END
```

Pior caso:
novo < v[j] sempre é verdade

Ordenação por inserção

- Algoritmo:

Melhor caso:
novo < v[j] sempre é falso

```
Sort( v[N] ):
   FOR i = 1; i < N; i++:
        novo = v[i] // 'novo' é o primeiro não ordenado
        j = i-1 // 'j' é o fim da parte ordenada
        WHILE j ≥ 0 && novo < v[j] :
               v[j+1] = v[j] // desloca elementos
               j = j-1
        END_WHILE
        v[j+1] = novo // insere novo elemento no local
   END_FOR
END
```

Algoritmos de ordenação

- Ordenação por seleção
  - Pior e melhor caso: $\Theta(n^2)$
    - Qualquer vetor

- Ordenação por inserção
  - Pior caso: $\Theta(n^2)$
    - Vetor ordenado em ordem inversa
  - Melhor caso: $\Theta(n)$
    - Vetor já ordenado