ORDENAÇÃO POR INSERÇÃO E SELEÇÃO

Engenharia de Software
Discente: Júlia de Souza Nascimento

INF
INSTITUTO DE INFORMÁTICA

UFG
UNIVERSIDADE FEDERAL DE GOIÁS

O QUE SÃO ALGORITMOS DE ORDENAÇÃO?

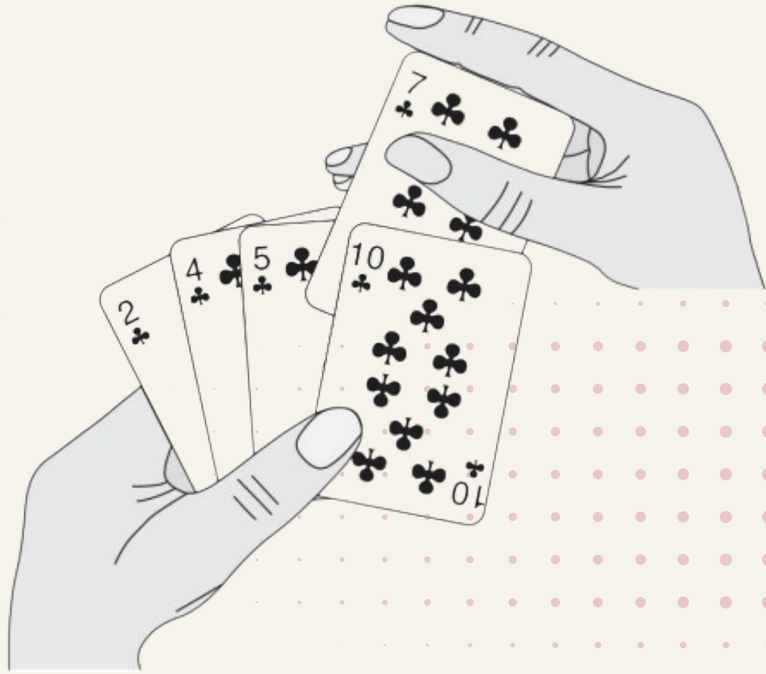
- Algoritmos que organizam elementos de uma lista em uma ordem específica (ex: crescente).

- Importantes para otimização e eficiência em computação.

ORDENAÇÃO POR INSERÇÃO (INSERTION SORT)

Organiza uma lista à medida que percorre os elementos, inserindo cada elemento na posição correta em relação aos anteriores.

Como se você estivesse organizando cartas de baralho na sua mão. Você pega uma carta de cada vez e a coloca na posição correta em relação às cartas que já estão na sua mão.



COMO FUNCIONA A ORDENAÇÃO POR INSERÇÃO

1 - Inicialização

Começamos com a primeira carta na mão (a primeira posição da lista está ordenada por definição).

2 - Iteração

Pegamos a próxima carta (ou elemento) e a comparamos com as cartas já ordenadas na mão, da direita para a esquerda.

3 - Inserção

Movemos as cartas maiores para a direita até encontrar a posição correta para a carta atual, então a inserimos ali.

4 - Repetição

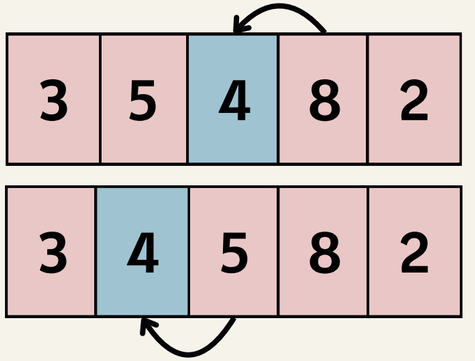
Repetimos os passos 2 e 3 até que todas as cartas (elementos) estejam ordenadas.

EXEMPLO PRÁTICO:

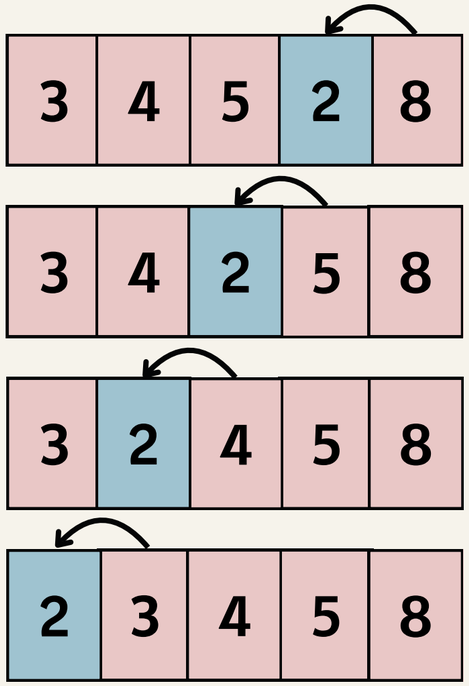
Considere a lista de números: 5 | 3 | 8 | 4 | 2 .

- Começamos com o primeiro elemento: 5 | 3 | 8 | 4 | 2
- Pegamos o segundo elemento 3 e o comparamos com 5 :
  - 3 é menor que 5, então inserimos 3 antes de 5: 3 | 5 | 8 | 4 | 2
- Pegamos o próximo elemento 8 e o comparamos com 5 :
  - 8 é maior que 5, então está na posição correta: 3 | 5 | 8 | 4 | 2

- Pegamos o próximo elemento 4 e o comparamos com 8 e depois com 5 :
  - 4 é menor que 8, então movemos 8 para a direita.
  - 4 é menor que 5, então movemos 5 para a direita.
  - Inserimos 4 na posição correta: 3 4 5 8 2 .



- Pegamos o último elemento 2 e o comparamos com todos os anteriores: 3 4 5 8 2
  - 2 é menor que 8, movemos 8.
  - 2 é menor que 5, movemos 5.
  - 2 é menor que 4, movemos 4.
  - 2 é menor que 3, movemos 3.
  - Inserimos 2 na posição correta.



ALGORITMO (INSERTION SORT)

Pseudocódigo

```
algoritmo "OrdenacaoPorInsercao"
var
   i, j, chave : inteiro
   vetor : vetor[1..5] de inteiro
inicio

   vetor[] := (5, 3, 8, 4, 2)
   n := 5

   // Algoritmo de Inserção
   para i de 2 ate n faca
      chave := vetor[i]
      j := i - 1

      enquanto (j > Ø) e (vetor[j] > chave) faca
         vetor[j + 1] := vetor[j]
         j := j - 1
      fimenquanto

      vetor[j + 1] := chave
   fimpara
fimalgoritmo
```

GO

```
package main

import "fmt"

func insertionSort(arr []int) {
    for i := 1; i < len(arr); i++ {
        key := arr[i]
        j := i - 1
        for j >= Ø && arr[j] > key {
            arr[j+1] = arr[j]
            j--
        }
        arr[j+1] = key
    }
}

func main() {
    arr := []int{5, 3, 8, 4, 2}
    insertionSort(arr)
    fmt.Println("Vetor ordenado:", arr)
}
```

ORDENAÇÃO POR SELEÇÃO (SELECTION SORT)

Organiza uma lista à medida que seleciona repetidamente o menor elemento e coloca-o na posição correta.

Agora, vamos pensar no algoritmo de ordenação por seleção como se estivéssemos organizando livros em uma estante, escolhendo o menor livro de cada vez e colocando-o na posição correta.

COMO FUNCIONA A ORDENAÇÃO POR INSERÇÃO

1 - Inicialização

Começamos com a primeira posição da estante (lista).

2 - Encontrar o menor

Percorremos toda a estante (lista) para encontrar o menor elemento.

3 - Troca

Trocamos o menor elemento encontrado com o elemento na posição inicial.

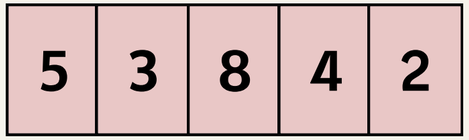
3 - Iteração

Repetimos os passos 2 e 3 para a próxima posição, ignorando a já ordenada.

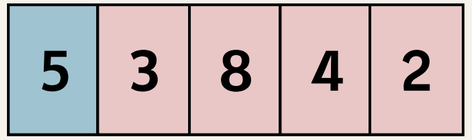
4 - Repetição

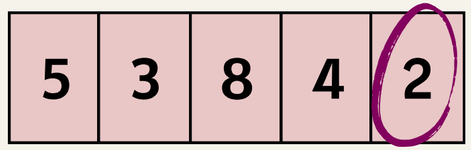
Continuamos até que toda a estante/lista esteja ordenada.

EXEMPLO PRÁTICO:

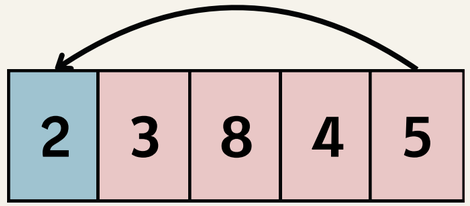
Considere a mesma lista de números: 

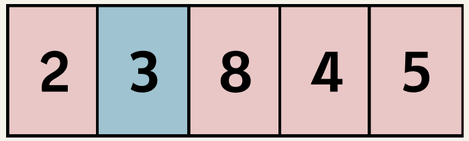
.

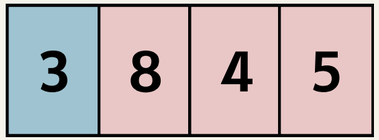
- Começamos com a primeira posição: 

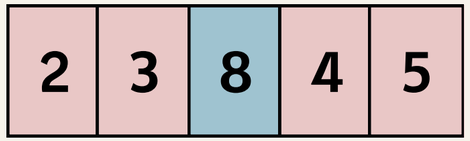
  - Encontramos o menor elemento da lista. 

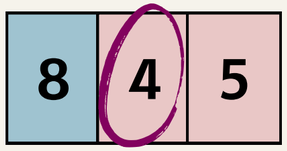
  - Trocamos 2 com o elemento da primeira posição:



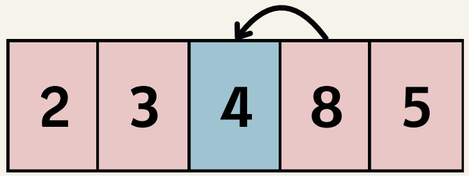
- Passamos para a segunda posição: 

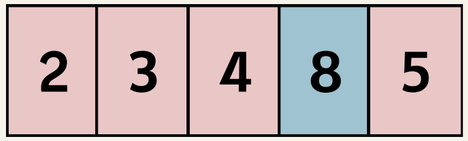
  - O menor elemento da sub-lista 

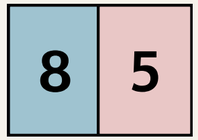
é 3, que já está na posição correta.
- Passamos para a terceira posição: 

  - Encontramos o menor elemento da sub-lista 

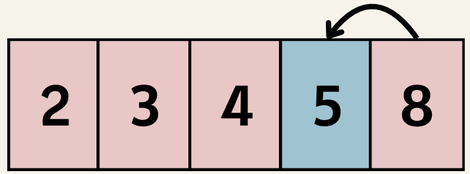
.
  - Trocamos 4 com o elemento da terceira posição (8):



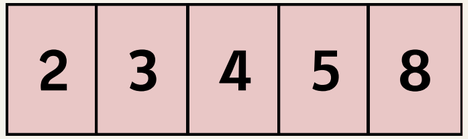
- Passamos para a quarta posição: 

    - O menor elemento da sub-lista 

é 5.
    - Trocamos 5 com o elemento da quarta posição (8):



- A última posição já está ordenada, então terminamos.



ALGORITMO (SELECTION SORT)

Pseudocódigo

```
algoritmo "OrdenacaoPorSelecao"
var
   i, j, min, temp : inteiro
   vetor : vetor[1..5] de inteiro
inicio
   vetor[] := (5, 3, 8, 4, 2)
   n := 5
   // Algoritmo de Seleção
   para i de 1 ate n-1 faca
      min := i

      para j de i+1 ate n faca
         se vetor[j] < vetor[min] entao
            min := j
         fimse
      fimpara

      temp := vetor[i]
      vetor[i] := vetor[min]
      vetor[min] := temp
   fimpara
fimalgoritmo
```

GO

```
package main

import "fmt"
func selectionSort(arr []int) {
    n := len(arr)
    for i := Ø; i < n-1; i++ {
        minIdx := i
        for j := i + 1; j < n; j++ {
            if arr[j] < arr[minIdx] {
                minIdx = j
            }
        }
        arr[i], arr[minIdx] =
arr[minIdx], arr[i]
    }
}
func main() {
    arr := []int{5, 3, 8, 4, 2}
    selectionSort(arr)
    fmt.Println("Vetor ordenado:", arr)
}
```

CONCLUSÃO

- **Ordenação por Inserção:**
  - **Bom para listas quase ordenadas.**
  - **Complexidade média: O(n²).**
  - **Inserção de elementos na posição correta.**

- **Ordenação por Seleção:**
  - **Simples e intuitivo.**
  - **Complexidade média: O(n²).**
  - **Escolha do menor elemento para cada posição.**

REFERÊNCIAS

- **Métodos de Ordenação: Selection, Insertion, Bubble, Merge (Sort)**
  https://ww2.inf.ufg.br/~hebert/disc/aedI/AEDI_04_ordenacaoI.pdf

- **Algoritmos e Estruturas de Dados II Ordenação**
  https://homepages.dcc.ufmg.br/~loureiro/alg/071/aeds2_ORDENACAO_1pp.pdf

THANK YOU

INF
INSTITUTO DE INFORMÁTICA

UFG
UNIVERSIDADE FEDERAL DE GOIÁS