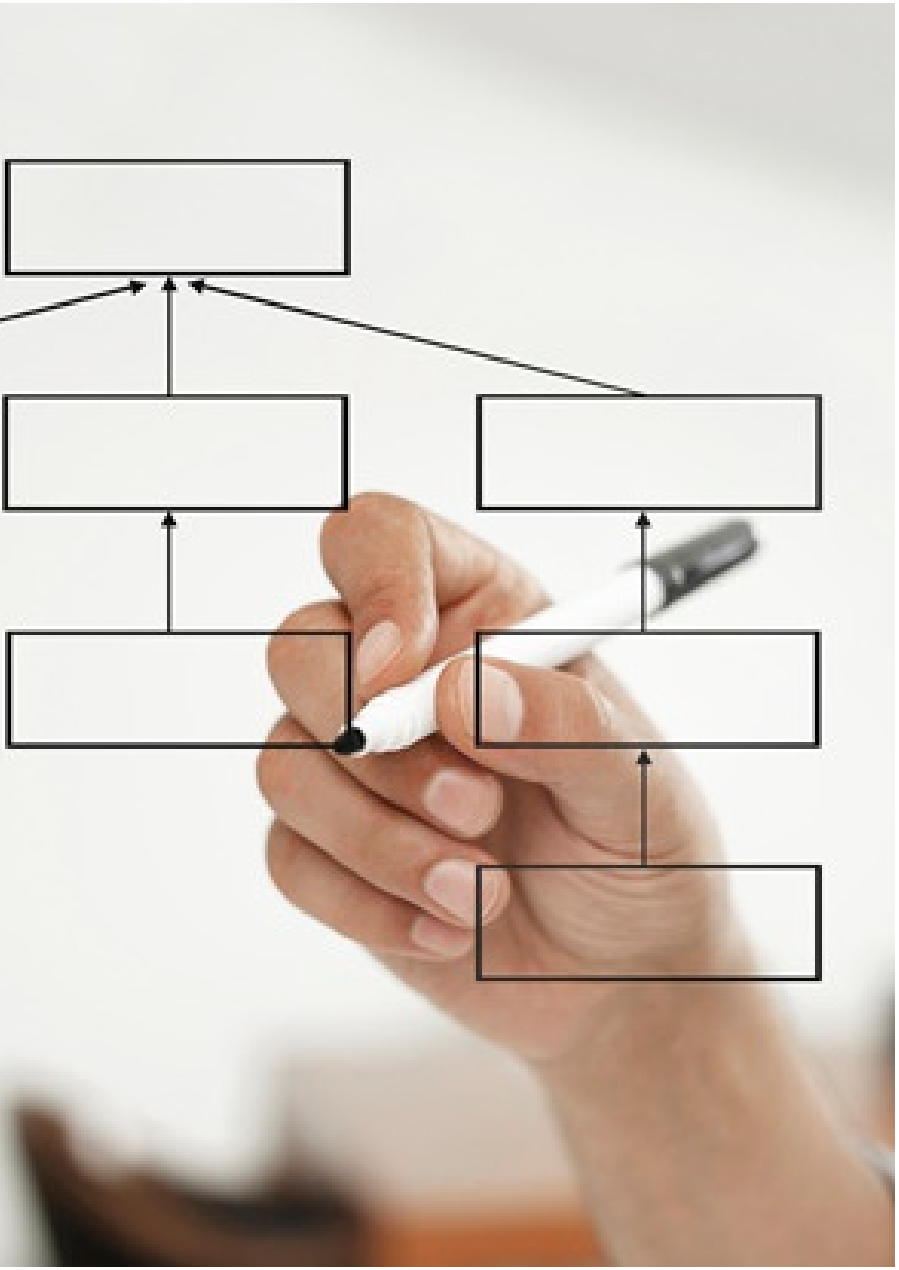
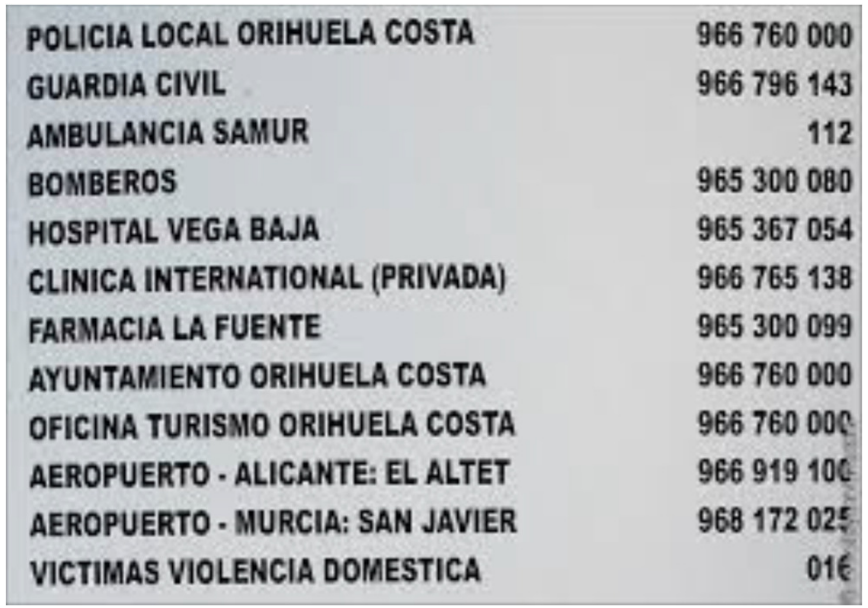


Estrutura de Dados I

CÁSSIO CAPUCHO PEÇANHA – 06

Ordenação

- Ordenar: processo de rearranjar um conjunto de objetos em uma ordem ascendente ou descendente.
- A ordenação visa facilitar a recuperação posterior de itens do conjunto ordenado.
  - Exemplo: Dificuldade de se utilizar um catálogo telefônico se os nomes das pessoas não estivessem listados em ordem alfabética.



POLICIA LOCAL ORIHUELA COSTA	966 760 000
GUARDIA CIVIL	966 796 143
AMBULANCIA SAMUR	112
BOMBEROS	965 300 080
HOSPITAL VEGA BAJA	965 367 054
CLINICA INTERNATIONAL (PRIVADA)	966 765 138
FARMACIA LA FUENTE	965 300 099
AYUNTAMIENTO ORIHUELA COSTA	966 760 000
OFICINA TURISMO ORIHUELA COSTA	966 760 000
AEROPUERTO - ALICANTE: EL ALTET	966 919 100
AEROPUERTO - MURCIA: SAN JAVIER	968 172 025
VICTIMAS VIOLENCIA DOMESTICA	016

Ordenação

- Notação utilizada nos algoritmos:
  - Os algoritmos trabalham sobre os registros de um arquivo.
  - Cada registro possui uma chave utilizada para controlar a ordenação.
  - Podem existir outros componentes em um registro.
- Qualquer tipo de chave sobre o qual exista uma regra de ordenação bem-definida pode ser utilizado.

Ordenação

- Do ponto da memória do computador, os algoritmos de ordenação podem ser classificados em:
  - **Ordenação Interna** (quando os dados a serem ordenados estão na memória principal).
  - **Ordenação Externa** (quando os dados a serem ordenados necessitam de armazenamento em memória auxiliar como por exemplo o disco HD).
- Na escolha de um algoritmo de **ordenação interna** deve ser considerado principalmente:
  - O tempo gasto pela ordenação;
  - O uso econômico da memória disponível;

Ordenação

- A maioria dos métodos de ordenação é baseado em comparação de chaves;
  - Exemplos: insertionsort, selectionsort, etc;
- Existem métodos de ordenação que utilizam o princípio da distribuição;
  - Exemplos: radixsort, bucketsort, etc;

Ordenação

- Exemplo de ordenação por distribuição: considere o problema de ordenar um baralho com 52 cartas na ordem:

$$A < 2 < 3 < \cdots < 10 < J < Q < K$$

e

$$\clubsuit < \diamondsuit < \heartsuit < \spadesuit.$$

- Algoritmo:
  - 1. Distribuir as cartas em treze montes: ases, dois, três, . . ., reis.
  - 2. Colete os montes na ordem especificada.
  - 3. Distribua novamente as cartas em quatro montes: paus, ouros, copas e espadas.
  - 4. Colete os montes na ordem especificada.

Ordenação

- Métodos como o ilustrado são também conhecidos como ordenação digital, radixsort ou bucketsort.
- O método não utiliza comparação entre chaves.
- Uma das dificuldades de implementar este método está relacionada com o problema de lidar com cada monte.
- Se para cada monte nós reservarmos uma área, então a demanda por memória extra pode tornar-se proibitiva.

Ordenação

- Algoritmos de ordenação podem ser aplicados a diversos tipos de estrutura, tais como:
  - Vetores;
  - Matrizes;
  - Estruturas dinâmicas.
- Na escolha de um algoritmo de ordenação interna deve ser considerado o tempo gasto pela ordenação.
- Sendo n o número registros no arquivo, as medidas de complexidade relevantes são:
  - Número de comparações C(n) entre chaves.
  - Número de movimentações M(n) de itens do arquivo.
- O uso econômico da memória disponível é um requisito primordial na ordenação interna.
- Métodos que utilizam listas encadeadas não são muito utilizados.
- Métodos que fazem cópias dos itens a serem ordenados possuem menor importância.

Ordenação

Classificação dos métodos de ordenação interna:

- Métodos simples:
  - Adequados para pequenos arquivos.
  - Requerem O(n2) comparações.
  - Produzem programas pequenos.

- Métodos eficientes:
  - Adequados para arquivos maiores.
  - Requerem O(n log n) comparações.
  - Usam menos comparações.
  - As comparações são mais complexas nos detalhes.
  - Métodos simples são mais eficientes para pequenos arquivos.

Ordenação por inserção

(InsertionSort)

25/08/2025

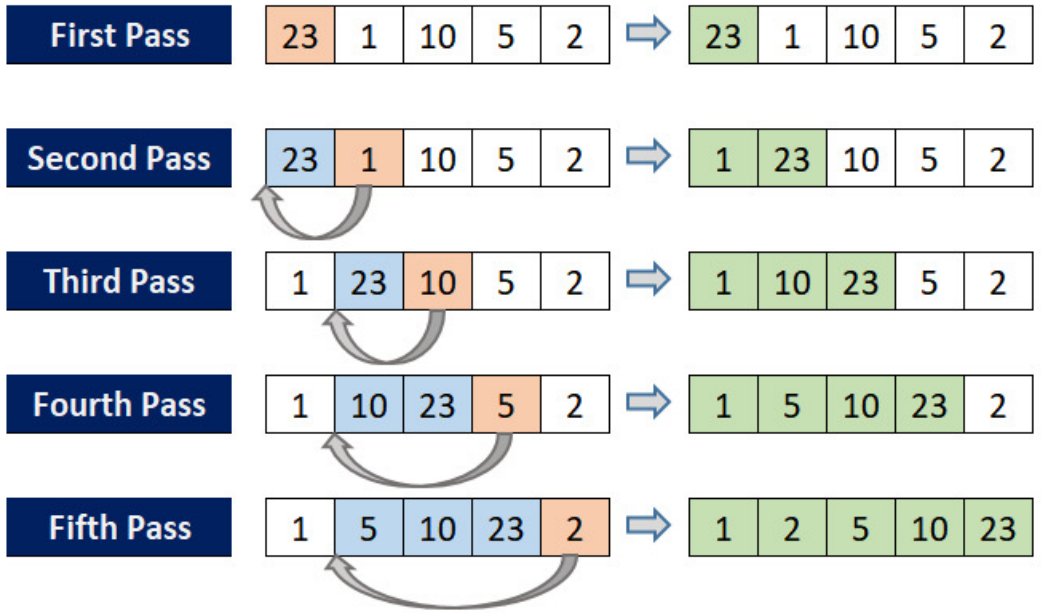
InsertionSort

Um dos algoritmos de implementação mais simples.

Método de ordenação semelhante ao que usamos para ordenar as cartas de um baralho.

Pega-se uma carta de cada vez e a coloca em seu devido lugar, sempre deixando as cartas da mão em ordem.

InsertionSort



InsertionSort - Exercício

1. Executar a função insertionSort para um vetor de tamanho 10 em ordem crescente e contar o número de vezes que a linha “V[j] = V[j - 1]; “ é executada;
2. Executar a função insertionSort para um vetor de tamanho 10 em ordem decrescente e contar o número de vezes que a linha “V[j] = V[j - 1]; “ é executada;
3. Qual é o pior caso e o melhor caso para este algoritmo?
4. A ordenação por inserção é estável?