

EDCO4B

ESTRUTURAS DE DADOS 2

Aula 06 - Conceitos sobre
Estruturas de Arquivos

Prof. Rafael G. Mantovani

Licença

Este trabalho está licenciado com uma Licença CC BY-NC-ND 4.0:



maiores informações:

https://creativecommons.org/licenses/by-nc-nd/4.0/deed.pt_BR

Roteiro

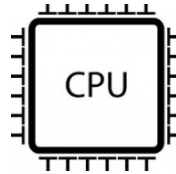


- 1** Introdução
- 2** Estruturas de Campos (*Fields*)
- 3** Estruturas de Registros (*Records*)
- 4** Exercício
- 5** Revisão
- 6** Referências

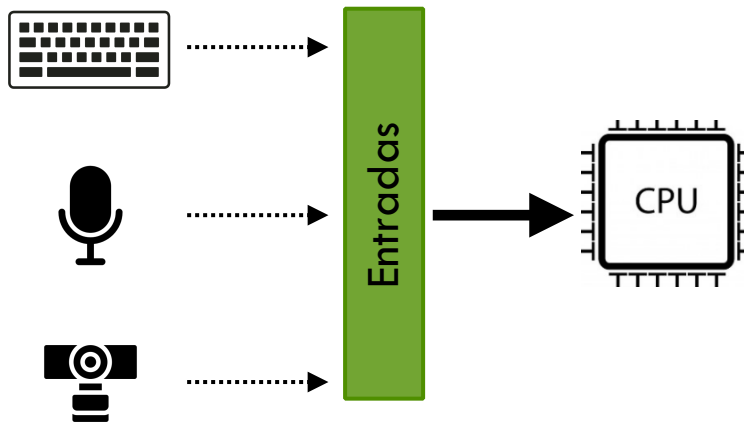
Roteiro

- 1** Introdução
- 2** Estruturas de Campos (*Fields*)
- 3** Estruturas de Registros (*Records*)
- 4** Exercício
- 5** Revisão
- 6** Referências

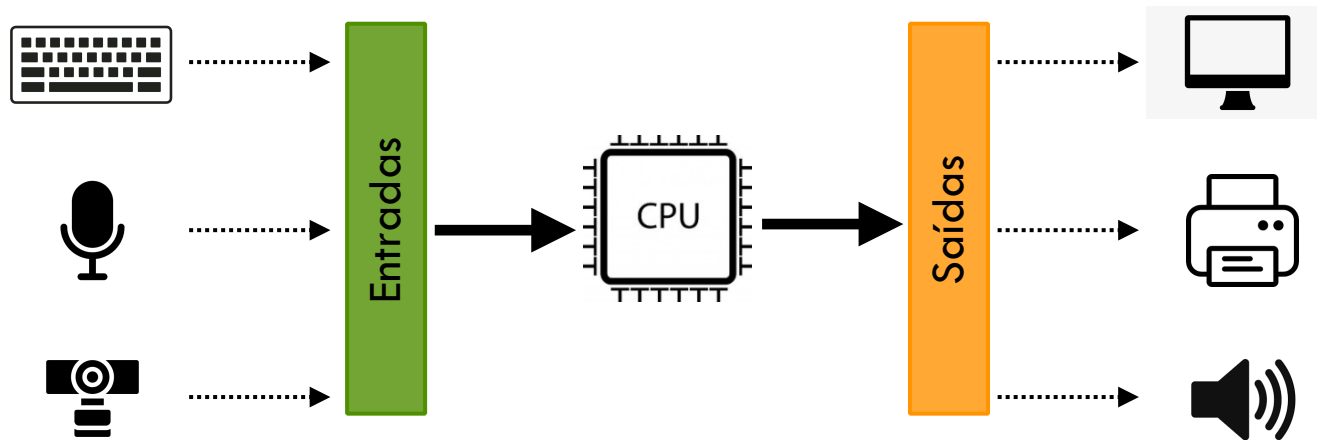
Introdução



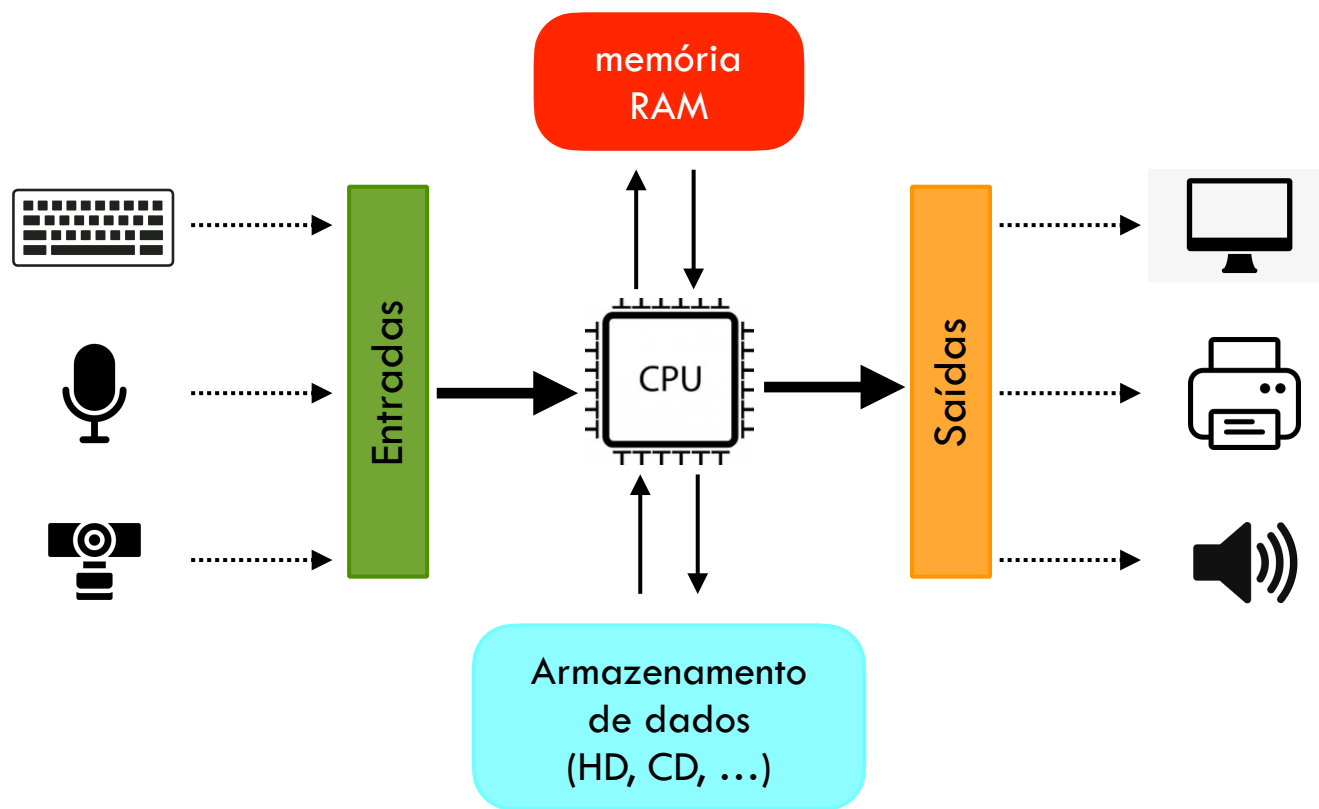
Introdução



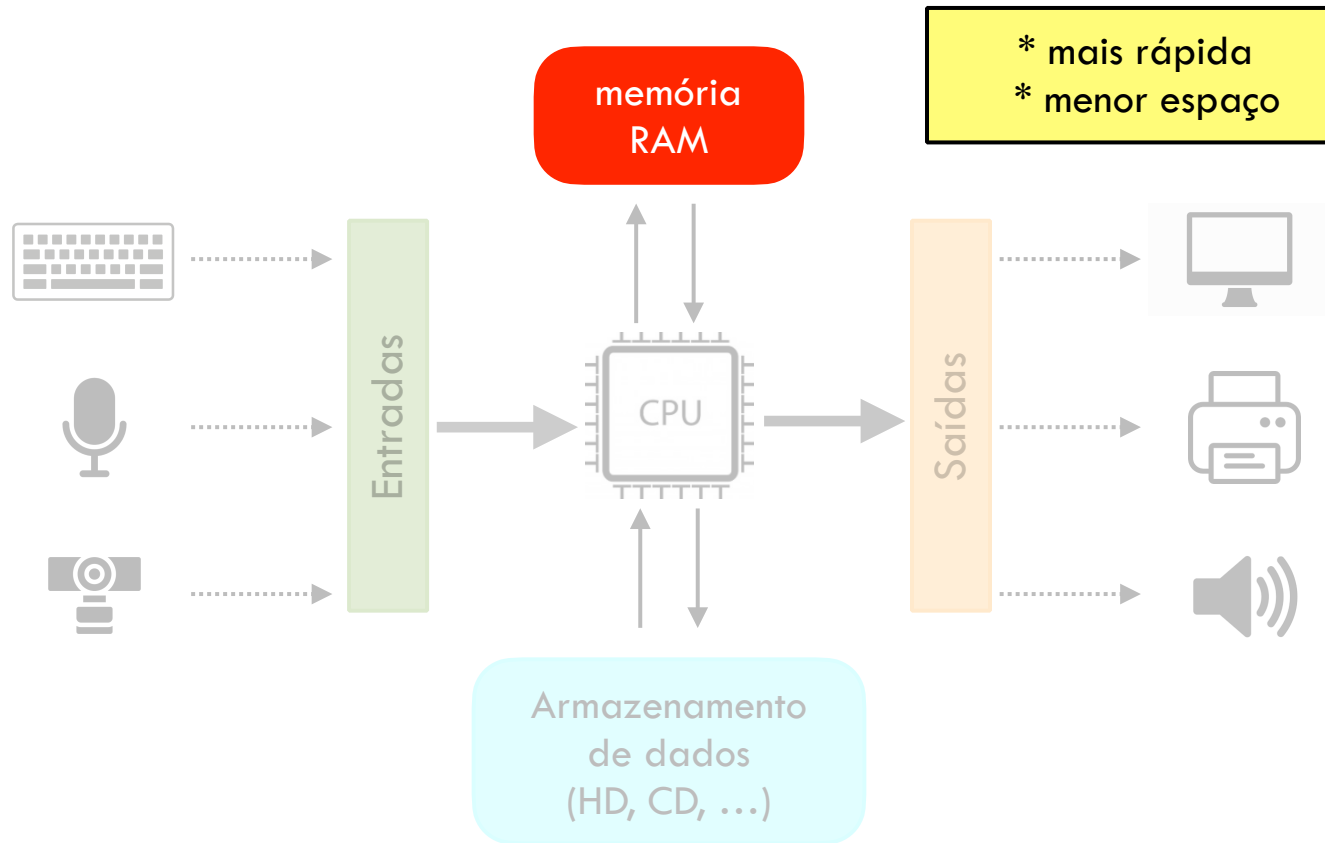
Introdução



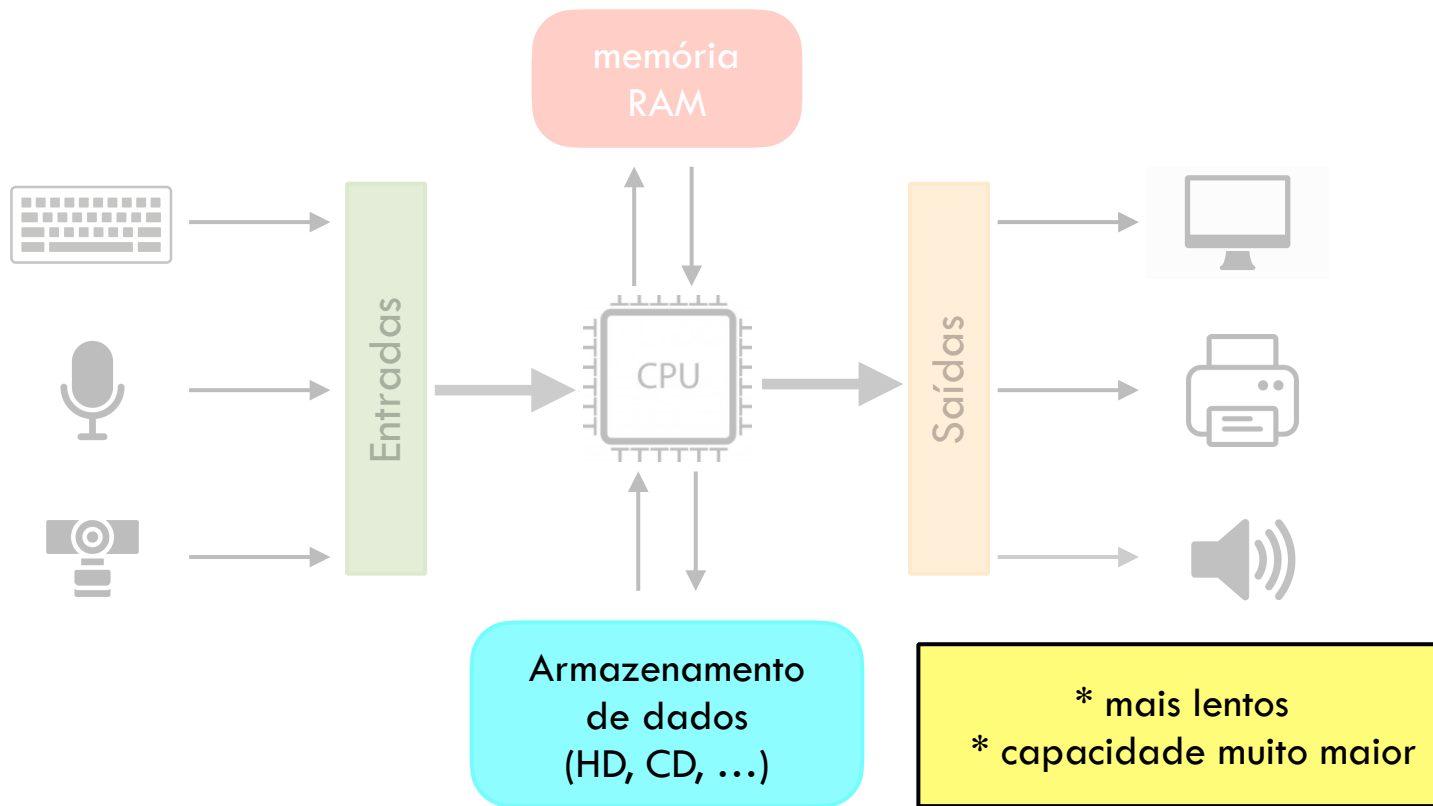
Introdução



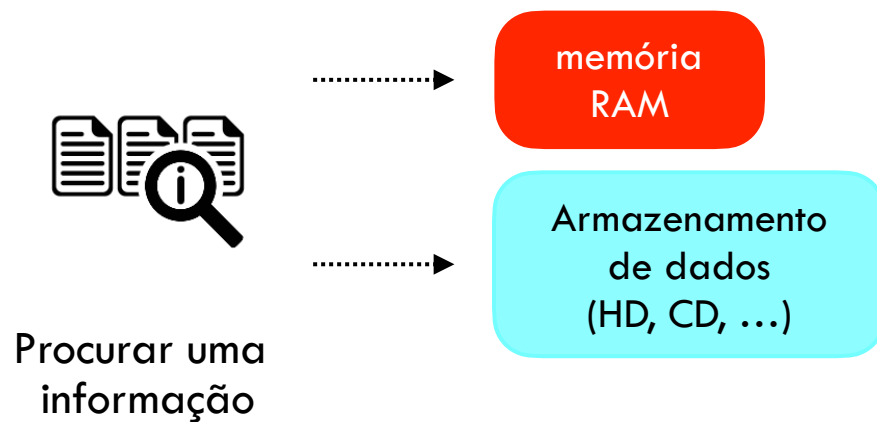
Introdução



Introdução



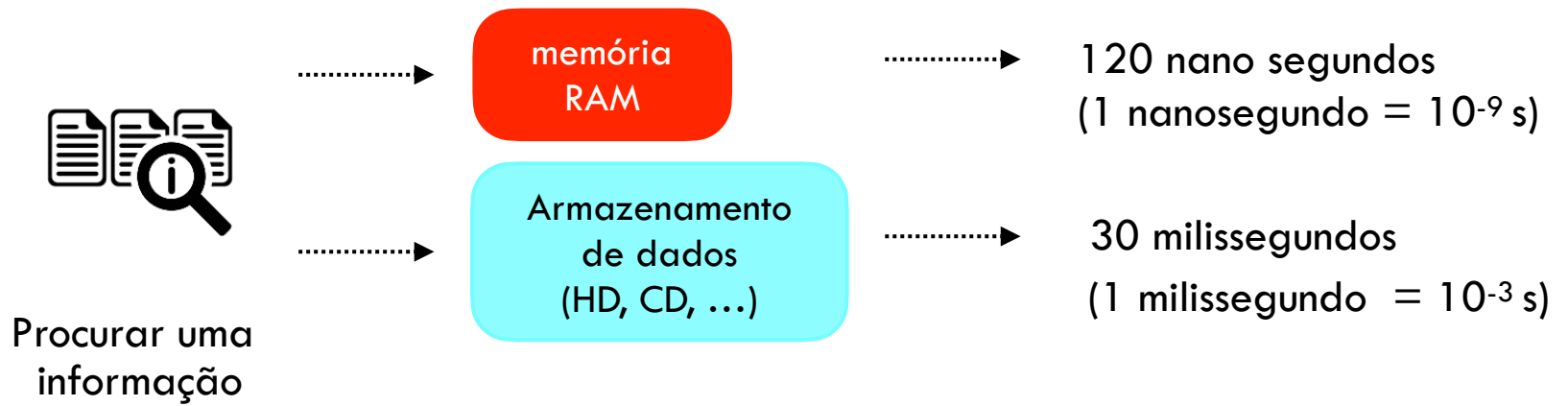
Introdução



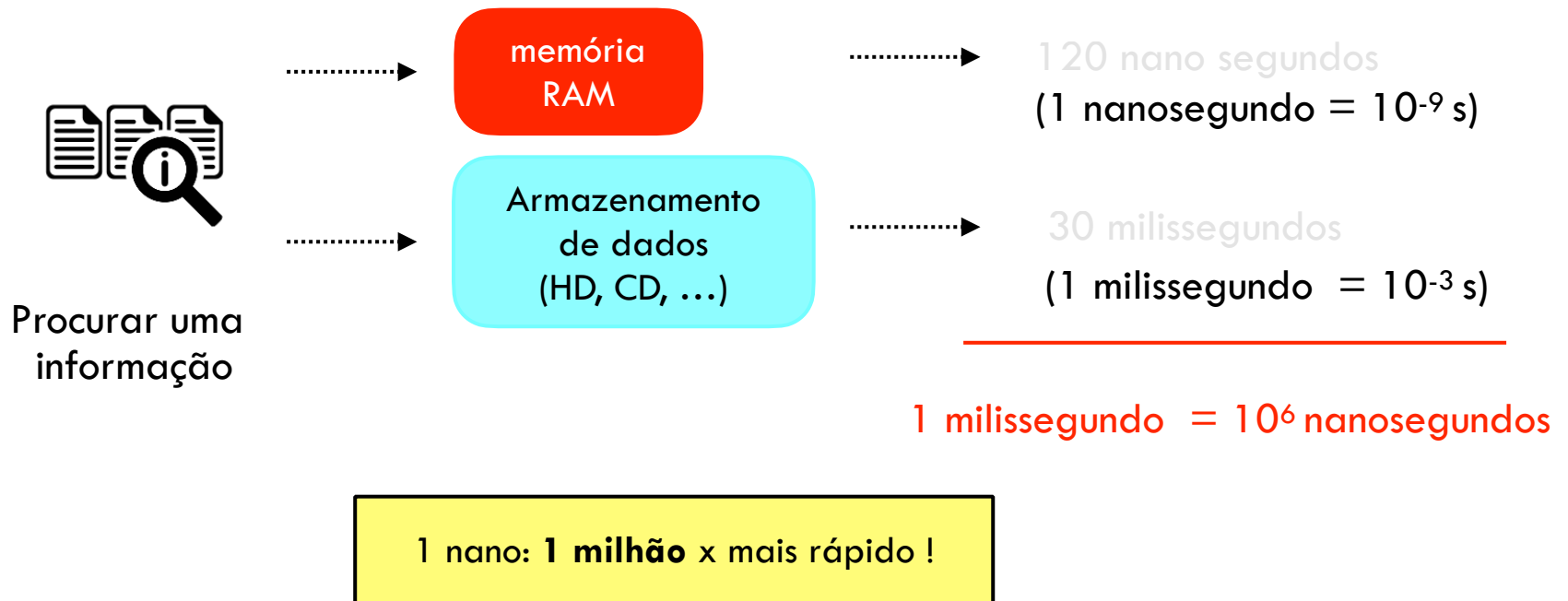
Introdução



Introdução

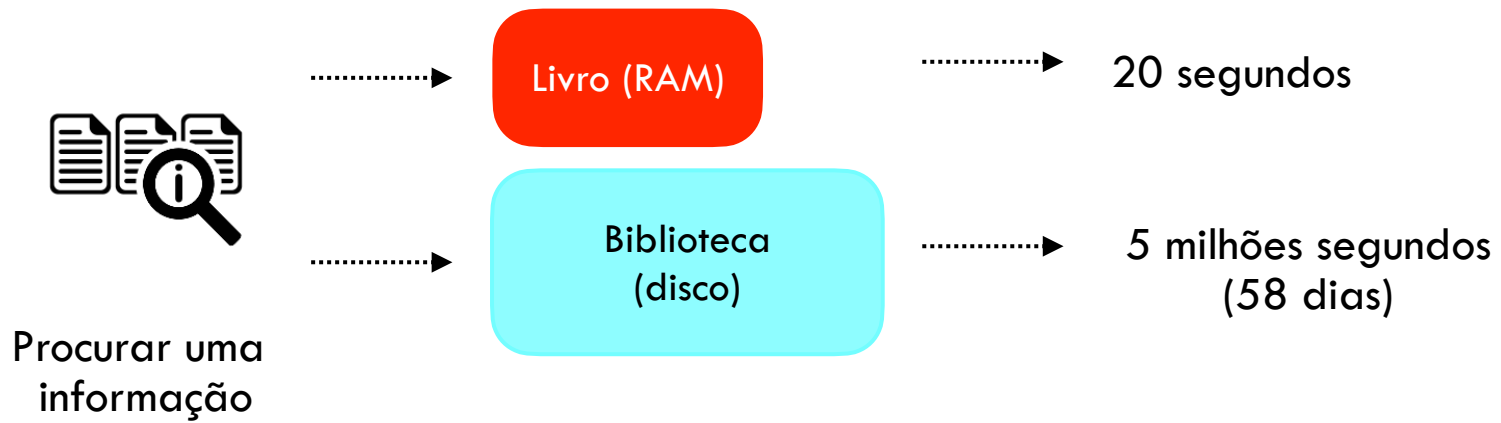


Introdução



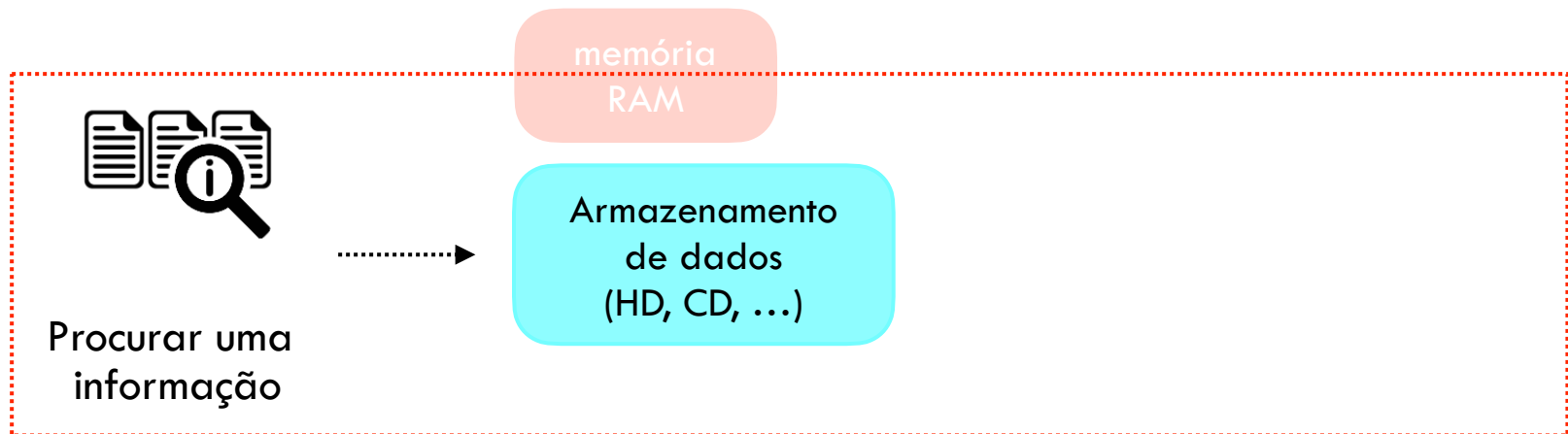
Introdução

mantendo as devidas proporções de tempo “**real**” ...



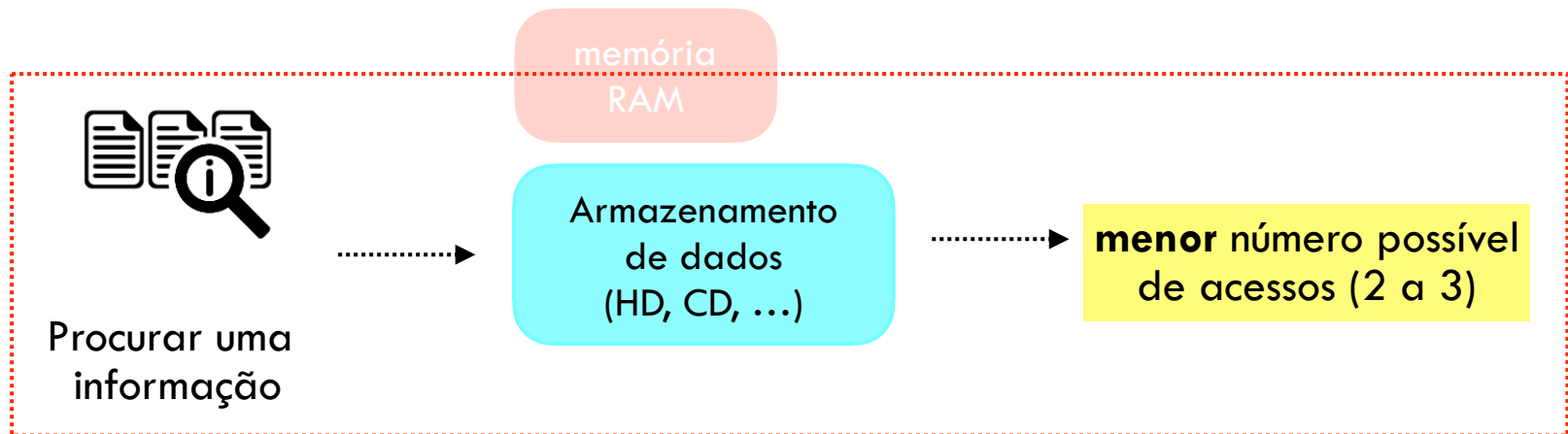
Introdução

Idealmente:



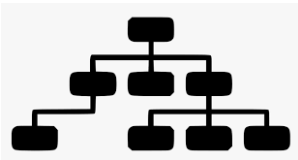
Introdução

Idealmente:



Introdução

Precisamos de ... :

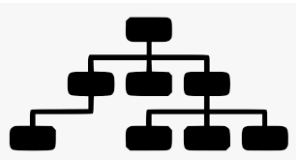


Estruturas de
Arquivos

combinação de uma representação de dados em arquivos,
e um conjunto de operações para acessar esses dados

Introdução

Precisamos de ... :



Estruturas de
Arquivos

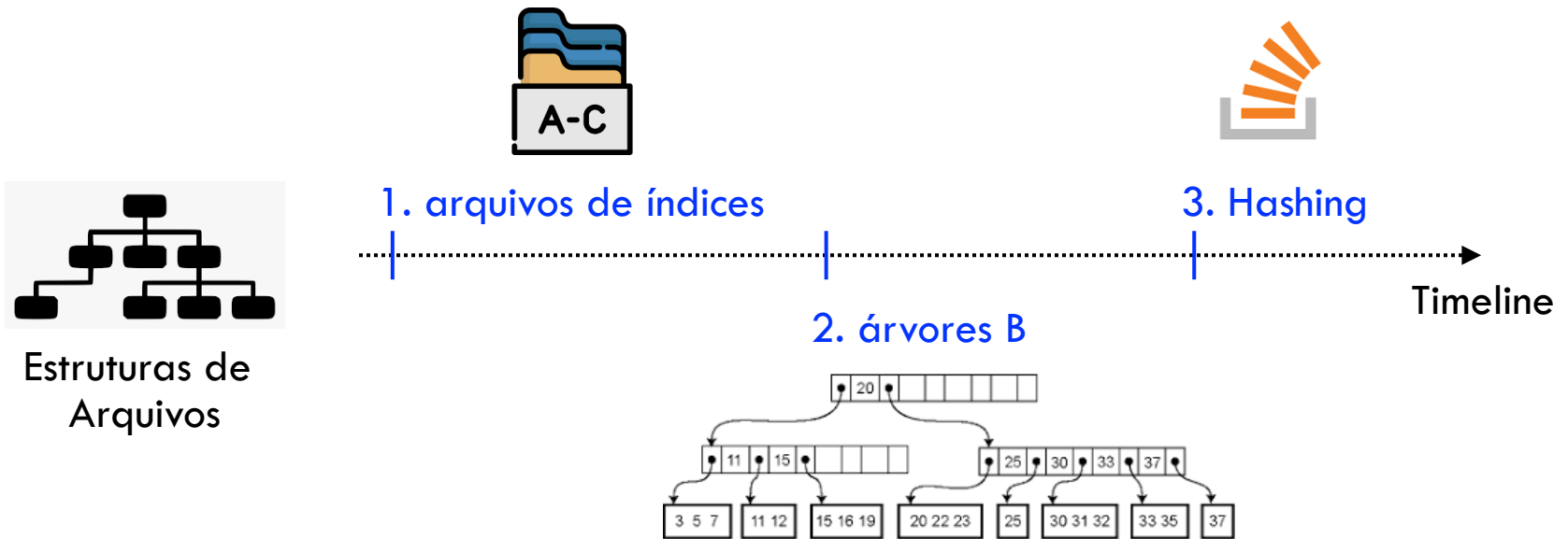
combinação de uma representação de dados em arquivos,
e um conjunto de operações para acessar esses dados

ler
escrever
modificar

recuperar
infos

design
efetivo

Introdução

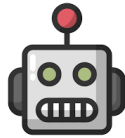


Roteiro

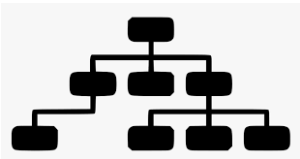
- 1 Introdução
- 2 Estruturas de Campos (*Fields*)
- 3 Estruturas de Registros (*Records*)
- 4 Exercício
- 5 Revisão
- 6 Referências

Estruturas de Campos

1



programa 1: cria dados na memória e
salva em um arquivo

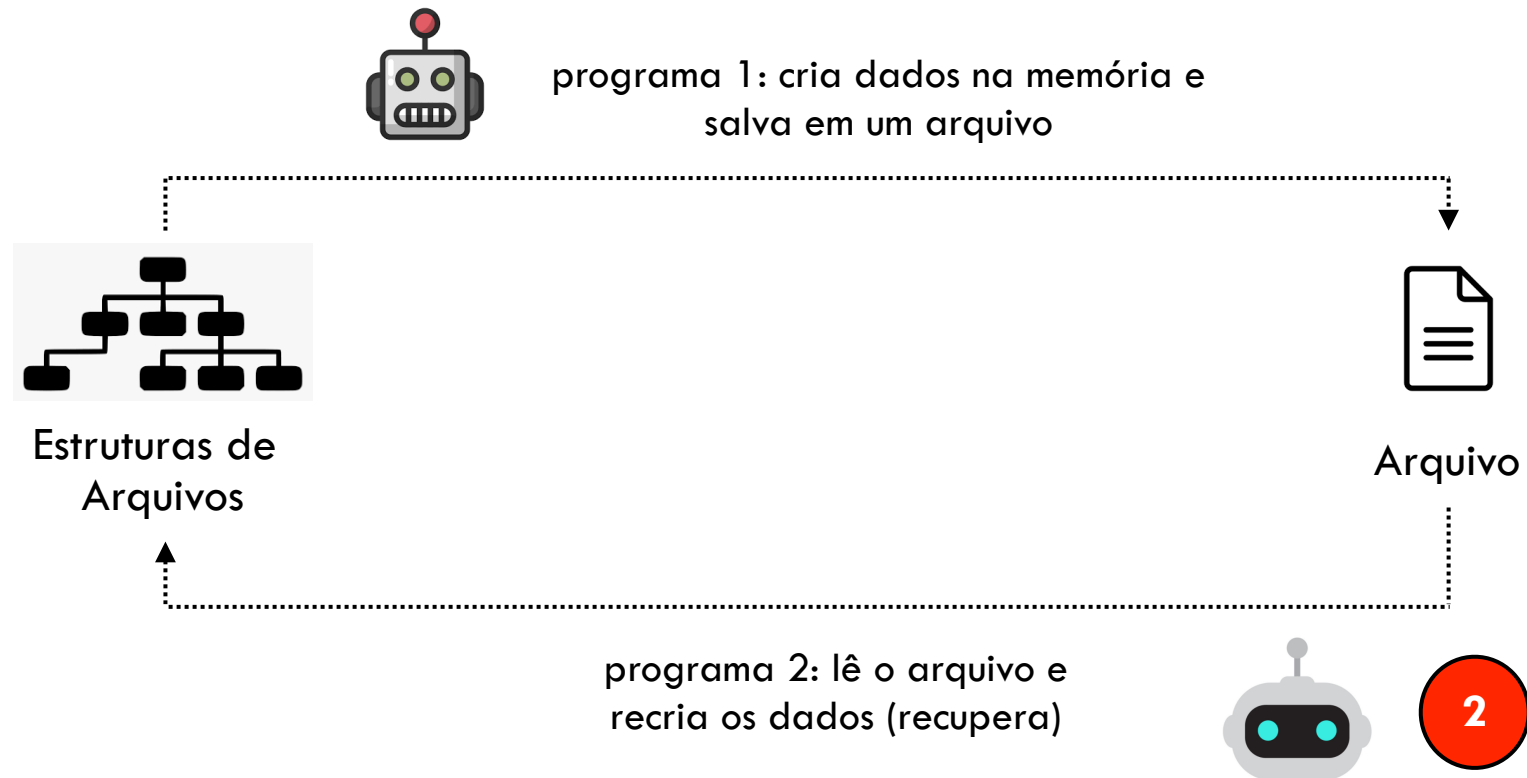


Estruturas de
Arquivos

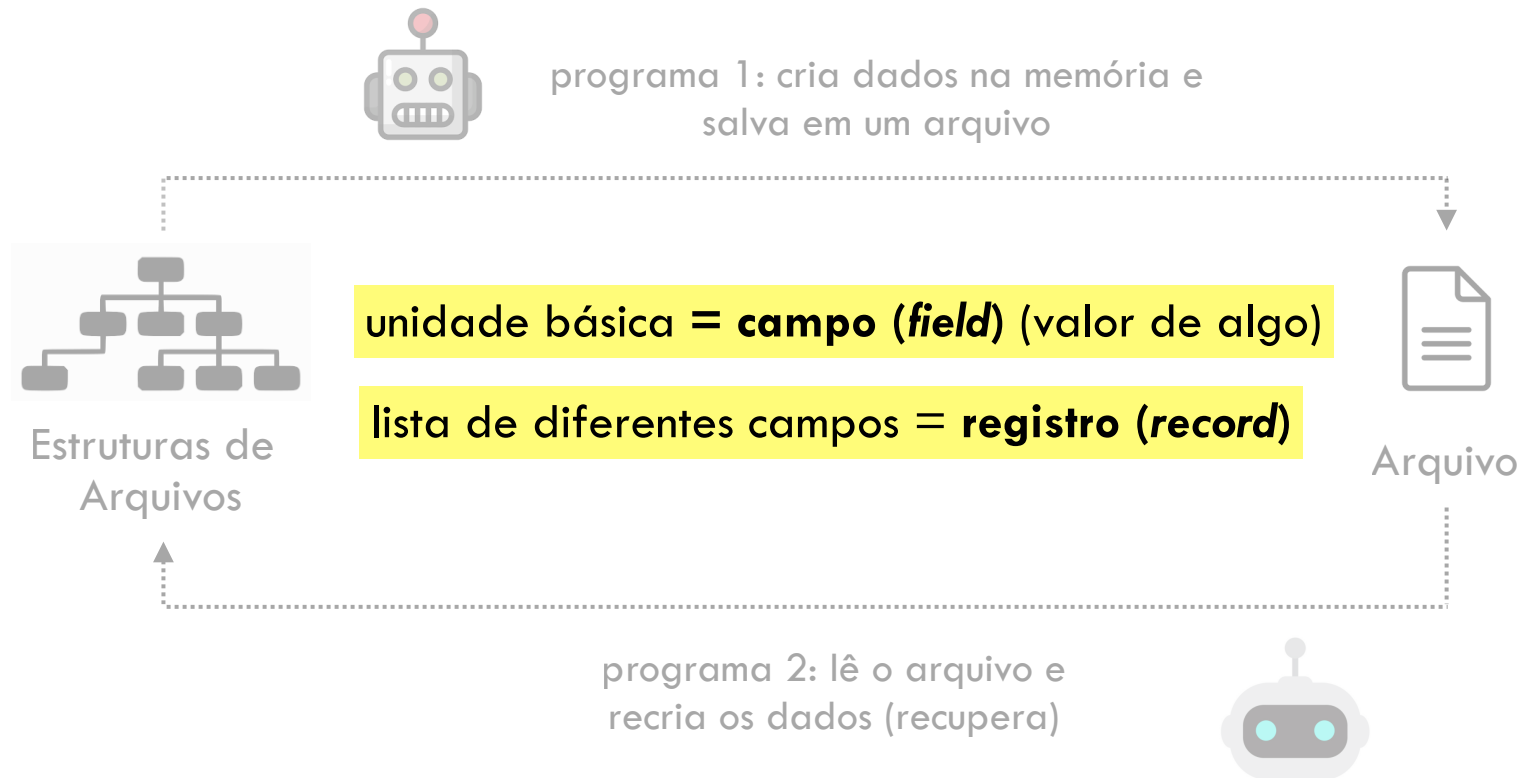


Arquivo

Estruturas de Campos

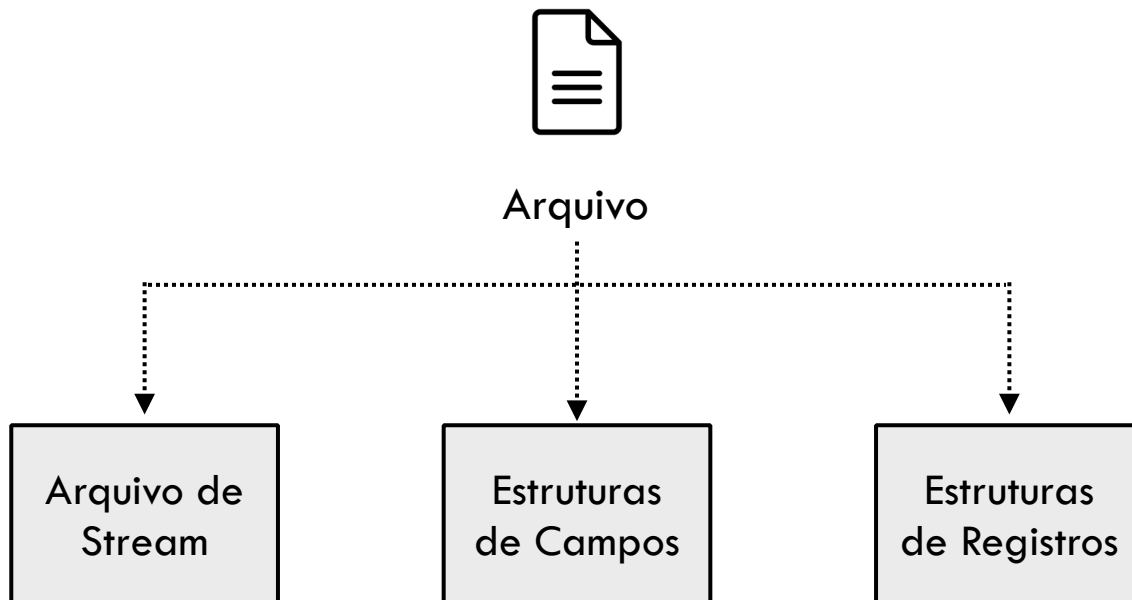


Estruturas de Campos



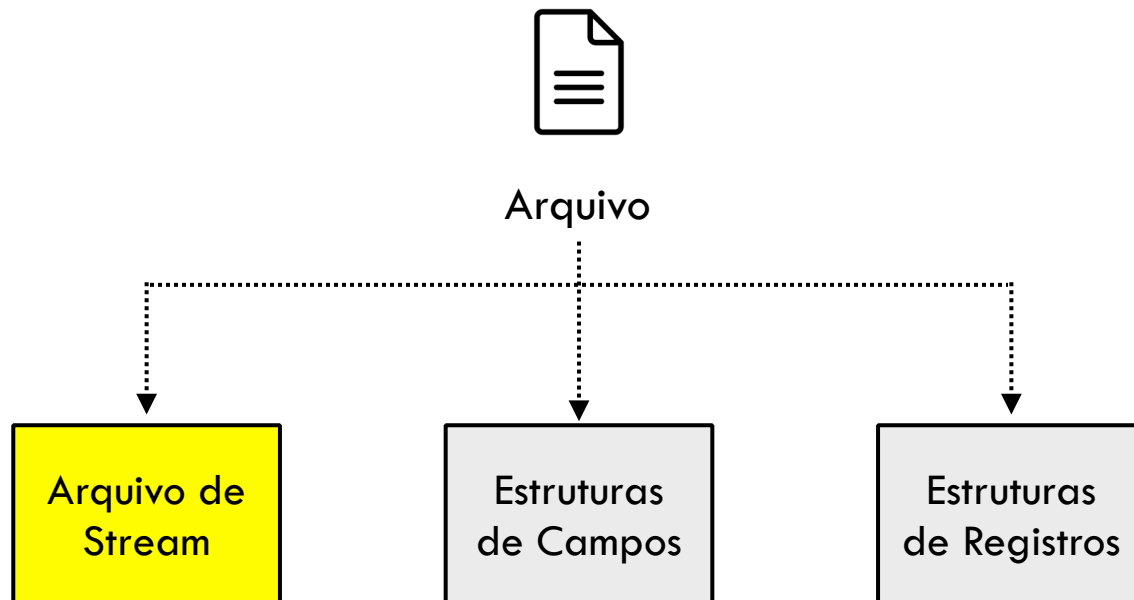
Estruturas de Campos

- Diferentes formas de representar a organização de arquivos



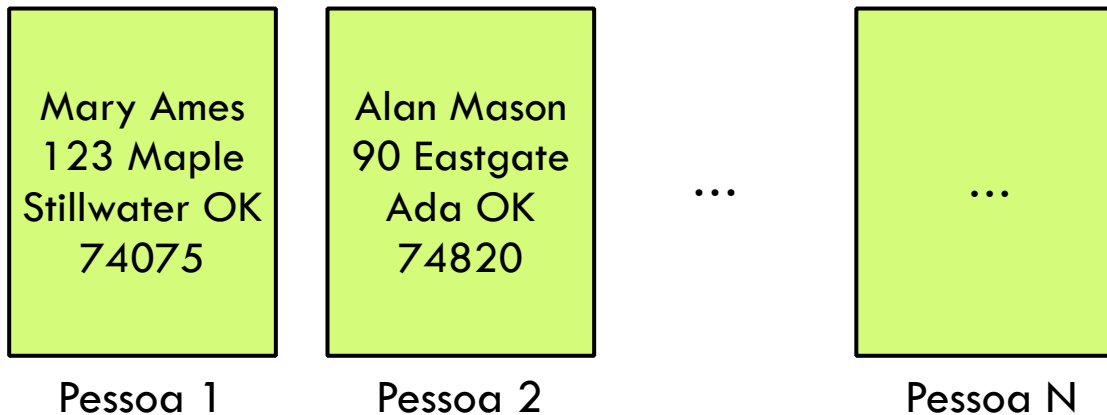
Estruturas de Campos

- Diferentes formas de representar a organização de arquivos



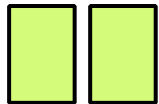
Arquivo de Stream

- **Info:** coleção de nomes e endereços



Arquivo de Stream

- **Info:** coleção de nomes e endereços



AmesMary123 MapleStillwaterOK740575MasonAlan90
EastgateAdaOK74820

Arquivo

Arquivo de Stream

- **Info:** coleção de nomes e endereços

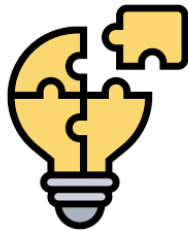


AmesMary123 MapleStillwaterOK740575MasonAlan90
EastgateAdaOK74820

Arquivo

- * **Problema:** perdemos a integridade dos dados
 - Não conseguimos ler novamente na mesma estrutura inicial
- * Precisamos: organizar o arquivo de alguma forma “inteligente”

Estruturas de Campos



Soluções

1. Forçar os campos a terem tamanhos fixos
2. Começar cada campo com um indicador de tamanho
3. Usar delimitadores ao fim de cada campo
4. Usar expressões keyword=valor

Método 1: tamanho fixo

- **Funcionamento:** tamanhos fixos, podemos prever e recuperar a informação contando os bytes

```
struct Person {  
    char last[11];  
    char first[11];  
    char address[16];  
    char city[16];  
    char state[3];  
    char zip[10];  
};
```

Método 1: tamanho fixo

- **Funcionamento:** tamanhos fixos, podemos prever e recuperar a informação contando os bytes

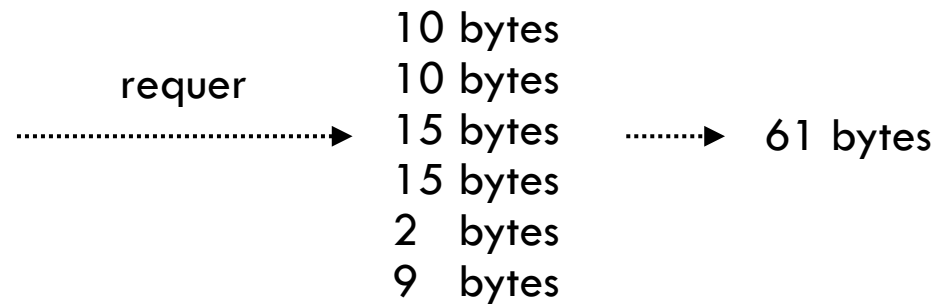
```
struct Person {  
    char last[11];  
    char first[11];  
    char address[16];  
    char city[16];  
    char state[3];  
    char zip[10];  
};
```

requer

10 bytes	
10 bytes	
15 bytes► 61 bytes
15 bytes	
2 bytes	
9 bytes	

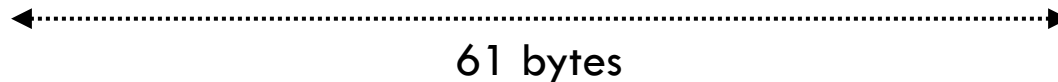
Método 1: tamanho fixo

```
struct Person {  
    char last[11];  
    char first[11];  
    char address[16];  
    char city[16];  
    char state[3];  
    char zip[10];  
};
```



Ames	Mary	123 Maple	Stillwater	OK 74075
Mason	Alan	90 Eastgate	Ada	OK 74820

Arquivo



Método 1: tamanho fixo

Ames	Mary	123 Maple	Stillwater	OK 74075
Mason	Alan	90 Eastgate	Ada	OK 74820

Arquivo



61 bytes

* Desvantagens:

- usar os campos de tamanhos fixos torna o arquivo maior
- inapropriado para dados de grande variabilidade (nomes e endereços)

Método 2: Indicador de tamanho

- **Funcionamento:** manter o tamanho do campo logo a frente dele

04Ames04Mary09123 Maple10Stillwater02OK057407505Mason04Alan1190
Eastgate03Ada02OK0574820

Arquivo

Método 3: Delimitadores

- **Funcionamento:** usar caracteres especiais para separar os campos (vazio, \n, Tab, etc)

```
Ames | Mary | 123 Maple | Stillwater | OK | 74075 |  
Mason | Alan | 90 Eastgate | Ada | OK | 74820 |
```

Arquivo

Método 4: Expressão chave-valor

- **Funcionamento:** usar expressão explícita de chave e valor

```
last=Ames | first=Mary | address=1 23 Maple | city=Stillwater | state=OK |  
zip=74075 |
```

Arquivo

Método 4: Expressão chave-valor

- **Funcionamento:** usar expressão explícita de chave e valor

```
last=Ames | first=Mary | address=1 23 Maple | city=Stillwater | state=OK |  
zip=74075 |
```

Arquivo

- * **Vantagens:**

- o próprio campo apresenta informação do dado
- bom para manipular eventuais valores ausentes

- * **Desvantagem:**

- gasta muito espaço de dados, por causa das chaves

Roteiro

- 1 Introdução
- 2 Estruturas de Campos (*Fields*)
- 3 Estruturas de Registros (*Records*)
- 4 Exercício
- 5 Revisão
- 6 Referências

Estruturas de Registros



Registro

Conjunto de campos que representam um conceito/objeto

Escrever um registro► Salvar o estado de um objeto

Ler um registro► Recuperar o estado de um objeto

Objeto reside na► **Memória**

Registro reside em► **Arquivos**

Estruturas de Registros



Registro

1. Registros de tamanhos fixos

2. Registros com uma quantidade de campos

3. Começar cada registro com a quantidade de bytes

4. Usar segundo arquivo mantendo o endereço inicial de cada registro

5. Usar delimitadores ao final de cada registro

Método 1: registros de tamanho fixo

- Todos registros tem o mesmo tamanho de bytes
- Um dos métodos mais usados
- Registro de tamanho fixo não implica em um número fixo de campos no registro

a) registros com tamanhos fixos com campos de tamanhos fixo

Ames	Mary	123 Maple	Stillwater	OK 74075
Mason	Alan	90 Eastgate	Ada	OK 74820

Registro 1

Registro 2

Método 1: registros de tamanho fixo

- Todos registros tem o mesmo tamanho de bytes
- Um dos métodos mais usados
- Registro de tamanho fixo não implica em um número fixo de campos no registro

a) registros com tamanhos fixos com campos de tamanhos fixo

Ames	Mary	123 Maple	Stillwater	OK 74075	Registro 1
Mason	Alan	90 Eastgate	Ada	OK 74820	Registro 2

b) registros com tamanhos fixos com campos tamanhos variados

Ames	Mary	123 Maple	Stillwater	OK	74075	←.....→
Mason	Alan	90 Eastgate	Ada	OK	74820	←.....→

Dados não usados

Método 2: número fixo de campos

- Registros vão conter um número fixo de campos

Ames | Mary | 1 23 Maple | Stillwater | OK | 74075 | Mason | Alan | 90 Eastgate | Ada | OK | 74820 |



6 campos compõem um registro

Arquivo

Método 3: indicador de tamanho

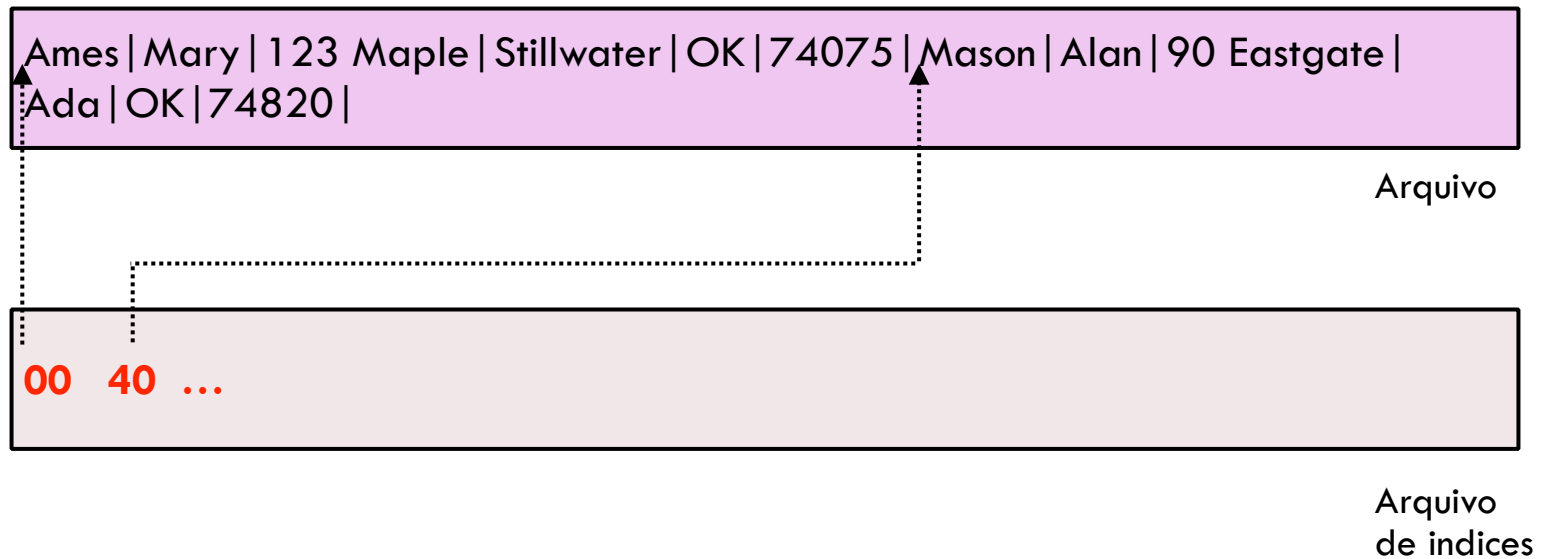
- Cada registro começa com um indicativo de tamanho em bytes

```
40Ames | Mary | 1 23 Maple | Stillwater | OK | 74075 | 36Mason | Alan | 90 Eastgate |  
Ada | OK | 74820 |
```

Arquivo

Método 4: arquivo de index

- Usar um arquivo adicional para guardar o endereço inicial de cada registro



Método 5: delimitadores

- No nível de registros, usar um caracter ao fim de cada registro

```
Ames | Mary | 123 Maple | Stillwater | OK | 74075 | # Mason | Alan | 90 Eastgate |  
Ada | OK | 74820 |
```

Arquivo

Roteiro

- 1 Introdução
- 2 Estruturas de Campos (*Fields*)
- 3 Estruturas de Registros (*Records*)
- 4 Exercício
- 5 Revisão
- 6 Referências

Exercícios

- 1) Implemente funções de escrita para o objeto Pessoa (*Person*) usando os seguintes métodos:

/ métodos de escrita em arquivos. Grave os objetos como texto, simulando uma escrita de byte */*

```
void writeFixedLengths(FILE *arq, Person obj);    // método 1  
void writeLengthIndicators(FILE *arq, Person obj); // método 2  
void writeKeywordTags(FILE *arq, Person obj);    // método 4
```

Exercícios

- 2) Implemente funções de escrita para o objeto Pessoa (*Person*) usando os seguintes métodos:

/ métodos de leitura em arquivos. Recupere os objetos já gravados em texto, simulando uma leitura de byte */*

```
void readFixedLengths(FILE *arq, Person obj);    // método 1  
void readLengthIndicators(FILE *arq, Person obj); // método 2  
void readKeywordTags(FILE *arq, Person obj);     // método 4
```

Roteiro

- 1 Introdução
- 2 Estruturas de Campos (*Fields*)
- 3 Estruturas de Registros (*Records*)
- 4 Exercício
- 5 Revisão
- 6 Referências

Revisão

- O menor nível de organização de um arquivo é um stream de *bytes*
- Campos (*fields*): pedaços fundamentais de informação
- Campos são agrupados em registros (objeto)
- Reconhecer campos/registros requer impor uma estrutura de organização nos arquivos

Revisão

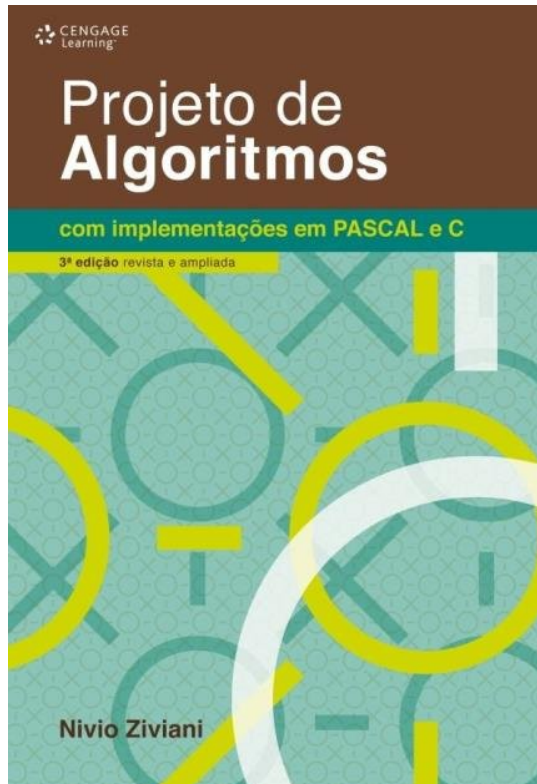


- Estruturas: existem algumas opções de organizar informação
 - Fixar tamanho dos campos/registros
 - Começar campos/registros com indicadores de tamanho
 - Usar delimitadores para dividir os campos/registros

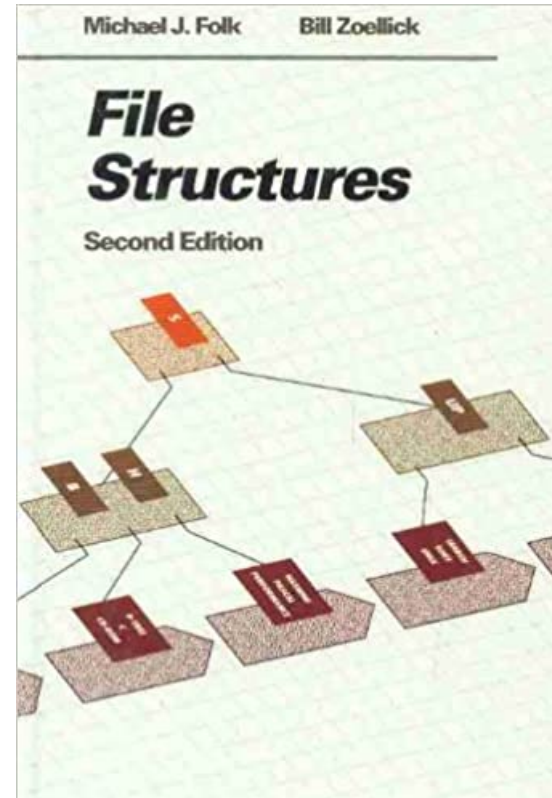
Roteiro

- 1 Introdução
- 2 Estruturas de Campos (*Fields*)
- 3 Estruturas de Registros (*Records*)
- 4 Exercício
- 5 Revisão
- 6 Referências

Referências sugeridas

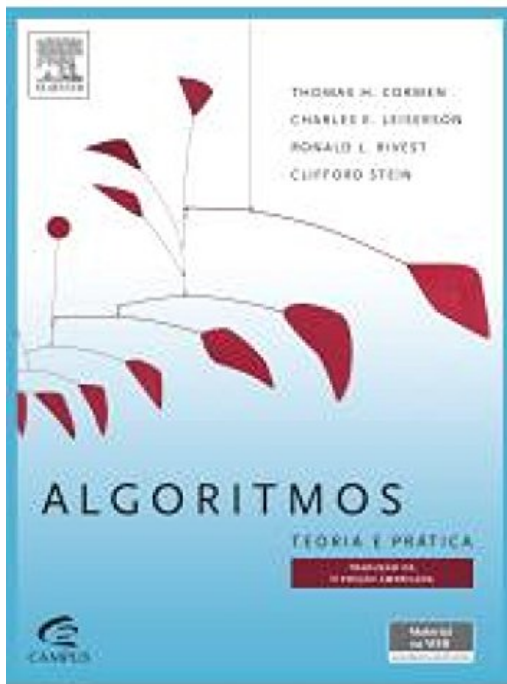


[Ziviani, 2010]



[Folk & Zoellick, 1992]

Referências sugeridas



[Cormen et al, 2018]



[Drozdek, 2017]

Perguntas?

Prof. Rafael G. **Mantovani**

rafaelmantovani@utfpr.edu.br