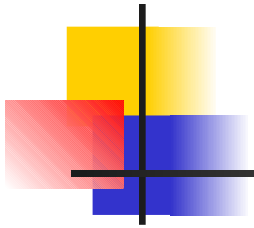


Algoritmos e Estruturas de Dados

II

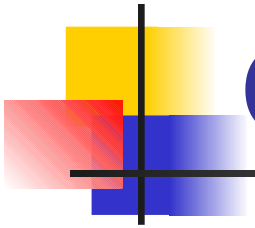
Prof. Ricardo J. G. B. Campello

Campos e Registros

Adaptado dos Originais de:

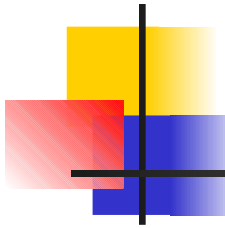
Leandro C. Cintra

Maria Cristina F. de Oliveira



Organização de Arquivos

- Informações em arquivos são, em geral, organizadas logicamente em campos e registros
  - campos e registros são conceitos lógicos
    - possuem associação com o **arquivo lógico**
- Dependendo de como a informação é mantida, campos lógicos sequer podem ser recuperados...



Seqüência de Bytes (stream)

- Exemplo:
  - Suponha que desejamos armazenar em um arquivo os nomes e endereços de várias pessoas
  - Suponha que decidimos representar os dados como uma seqüência simples de bytes
    - caracteres sem delimitadores, contadores, etc

Seqüência de Bytes (stream)

- Exemplo:

John Ames

123 Maple

Stillwater,

74075

Alan Mason

90 Eastgate

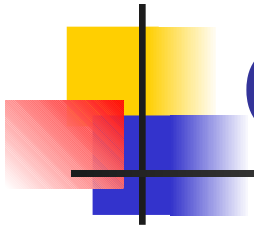
Ada,

74820

AmesJohn123 MapleStillwater74075MasonAlan90 EastgateAda74820

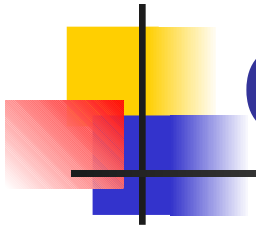
Seqüência de Bytes (stream)

- Uma vez escritas as informações, em princípio não existe como recuperar as unidades lógicas
  - perde-se a integridade das unidades fundamentais de organização dos dados
    - essas unidades são agregados de caracteres
    - tais agregados são chamados **campos** (fields)



Organização em Campos

- **Campo:**
  - **menor unidade lógica** de informação em arquivo
    - noção lógica (ferramenta conceitual)
    - não está associada a um conceito físico
- Existem várias maneiras de organizar um arquivo mantendo a identidade dos campos
  - A organização anterior não proporciona isso...



Organização em Campos

- Estruturas de Organização de Campos:
  - Comprimento fixo
  - Indicador de comprimento
  - Delimitadores
  - Uso de *tags*

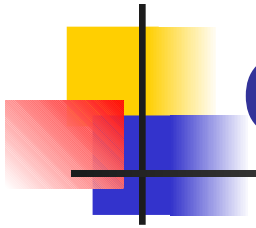
Organização em Campos

(a)	Maria	Rua 1	123	São Carlos
	João	Rua A	255	Rio Claro
	Pedro	Rua 10	56	Rib. Preto

(b)
05Maria05Rua 10312310São Carlos
04João05Rua A0325509Rio Claro
05Pedro06Rua 10025610Rib. Preto

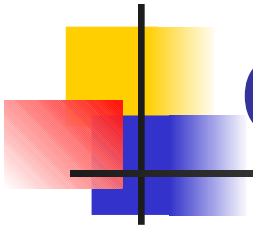
(c)
Maria|Rua 1|123|São Carlos|
João|Rua A|255|Rio Claro|
Pedro|Rua 10|56|Rib. Preto|

(d)
Nome=Maria|Endereço=Rua 1|Número=123|Cidade=São Carlos|
Nome=João|Endereço=Rua A|Número=255|Cidade=Rio Claro|
Nome=Pedro|Endereço=Rua 10|Número=56|Cidade=Rib. Preto|



Campos com Tamanho Fixo

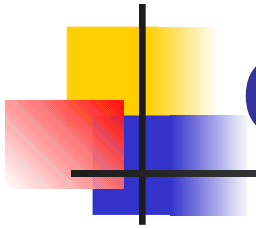
- Cada campo ocupa no arquivo um tamanho fixo, pré-estabelecido
  - por exemplo: 4 bytes
- O fato do tamanho ser conhecido garante que é possível recuperar cada campo



Campos com Tamanho Fixo

- Exemplo em C:

```
struct  set_of_fields {
        char last_name[10];
        char first_name[10];
        char address[15];
        char city[2];
        char zip_code[9];
};
```



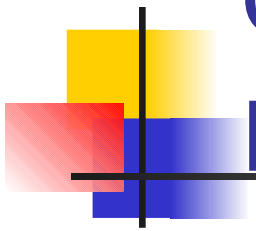
Campos com Tamanho Fixo

- O espaço alocado (e não usado) aumenta desnecessariamente o tamanho do arquivo
  - desperdício de memória secundária: **fragmentação**
- Solução inapropriada quando se tem uma grande variabilidade nos tamanhos dos campos
- Razoável apenas se o comprimento dos campos é realmente fixo, ou apresenta pouca variação

Campos com Indicador de Comprimento

- O tamanho de cada campo é armazenado imediatamente antes da informação
  - armazenamento binário ou ASCII
  - armazenamento binário:
    - requer um único byte se o tamanho do campo é inferior a 256 bytes

Campos Separados por Delimitadores



- Caractere especial (delimitador) inserido ao final de cada campo
  - delimitador não pode ser um caractere válido
    - pode ser um caractere ASCII não imprimível
      - por exemplo: LF (linefeed = ASCII 10)
    - espaços em branco não serviriam...
    - para ilustração, podemos utilizar “|”, “#”, ...

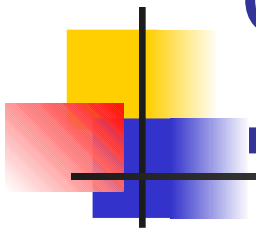
Campos Separados por Delimitadores

```
Define Constant: DELIMITER = ';'

PROGRAM: readstrm

    get input file name and open as INPUT
    initialize FIELD_COUNT and FIELD_CONTENT

    FIELD_LENGTH := readfield (INPUT, FIELD_CONTENT)
    while ( FIELD_LENGTH > 0 )

        increment the FIELD_COUNT
        write FIELD_COUNT and FIELD_CONTENT to the screen
        FIELD_LENGTH := readfield (INPUT, FIELD_CONTENT)

    endwhile

    close INPUT
end PROGRAM          // lê e imprime campos de um arquivo
```

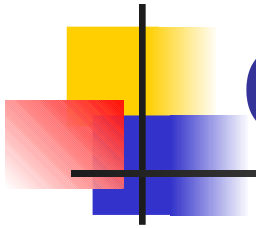
Campos Separados por Delimitadores

```
FUNCTION: readfield (INPUT, FIELD_CONTENT)

    initialize I
    initialize CH

    while (not EOF (INPUT) and CH does not equal DELIMITER)

        read a character from INPUT into CH
        increment I
        FIELD_CONTENT [I] := CH

    endwhile
    return (length of field that was read)

end FUNCTION                    // lê um campo de um arquivo
```

Campos com *Tags* "keyword=value"

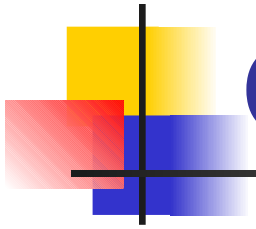


- Vantagem:
  - possui semântica local
    - campo fornece informação sobre si próprio
  - permite identificar localmente o conteúdo do arquivo
    - um campo “perdido” não compromete o arquivo
  - permite campos existirem ou não
- Desvantagem:
  - as *keywords* podem ocupar uma porção significativa do arquivo



Organização em Registros

- **Registro**:
  - um conjunto de campos agrupados
- Arquivo organizado em registros
  - nível de organização mais alto
- Assim como os campos, um registro é uma ferramenta conceitual
  - está associado ao arquivo lógico
  - outro nível de organização imposto aos dados



Organização em Registros

- Estruturas de Organização de Registros:
  - Tamanho **fixo**
    - Campos de tamanho fixo
    - Campos de tamanho variável
  - Tamanho **variável**
    - Número pré-determinado de campos
    - Uso de delimitadores
    - Indicador de tamanho
    - Uso de índice

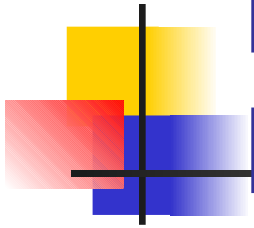
Registros de Tamanho Fixo

Registro de tamanho fixo e campos de tamanho fixo:

Maria	Rua 1	123	São Carlos
João	Rua A	255	Rio Claro
Pedro	Rua 10	56	Rib. Preto

Registro de tamanho fixo e campos de tamanho variável:

Maria|Rua 1|123|São Carlos| ← Espaço vazio →|
João|Rua A|255|Rio Claro| ← Espaço vazio →|
Pedro|Rua 10|56|Rib. Preto| ← Espaço vazio →|



Registros de Tamanho Fixo

- Assume que todos os registros ocupam o mesmo número de bytes:
  - campos podem ou não ocupar o mesmo no. de bytes
- Um dos métodos mais comuns de organização de arquivos:
  - simples, e
  - **permite acesso direto aos registros por RRN**
- Porém, pode ser inapropriado...
  - desperdício de memória secundária – **fragmentação**

Registros de Tamanho Variável

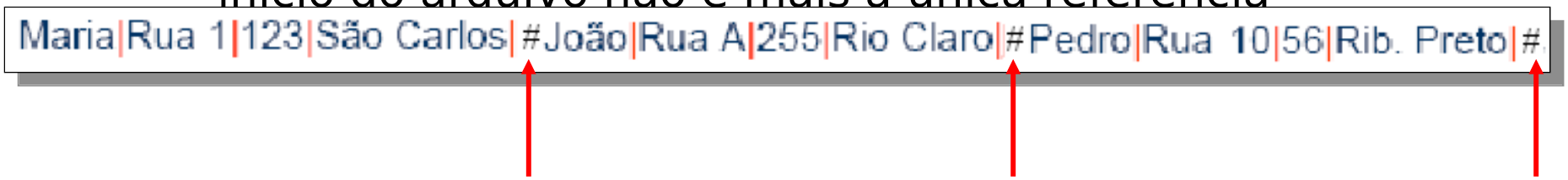
- Ao invés de especificar que cada registro contém um número fixo de bytes, podemos especificar um **número fixo de campos**
  - O tamanho do registro é **variável em bytes**
    - Acesso direto por RRN inviabilizado
- Por ex., campos separados por delimitadores:

Maria|Rua 1|123|São Carlos|João|Rua A|255|Rio Claro|Pedro|Rua 10|56|Rib. Preto|

4 campos | 4 campos | 4 campos

Registros de Tamanho Variável

- Pode-se ainda separar **registros** com delimitadores
  - análogos aos de fim de campo
    - porém outro caractere deve ser utilizado
    - delimitador de campos pode ser mantido
- Vantagens:
  - permite número variável de campos
  - início do arquivo não é mais a única referência



Registros de Tamanho Variável

Registro iniciados por indicador de tamanho:

27Maria|Rua 1|123|São Carlos|25João|Rua A|255|Rio Claro|27Pedro|Rua 10|56|Rib. Preto|

Arquivos de dados + arquivo de índices:

Dados: Maria|Rua 1|123|São Carlos|João|Rua A|255|Rio Claro|Pedro|Rua 10|56|Rib. Preto|

Índice: 00 27 52

Registros de Tamanho Variável

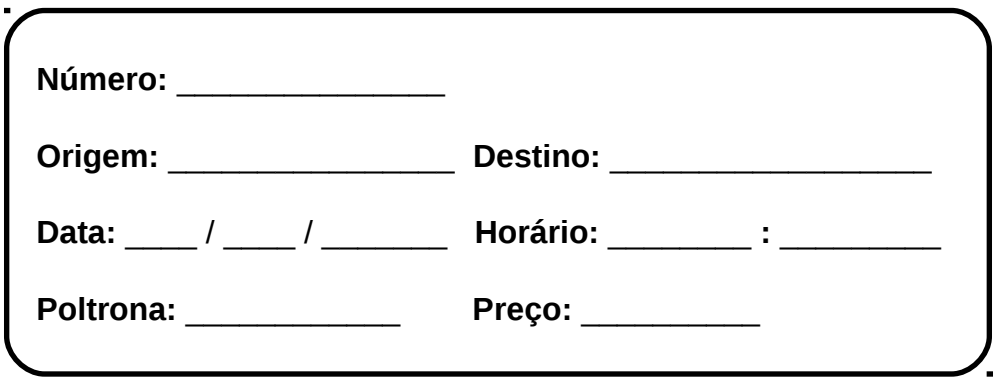
- Indicador de Tamanho:
  - indicador que precede o registro
    - fornece o seu tamanho total, em bytes
  - No início de cada registro:
    - sabe-se onde termina aquele registro
    - acessa-se diretamente o registro seguinte

Registros de Tamanho Variável

- Índice Externo:
  - Armazena o byte offset de cada registro
    - deslocamento relativo ao início do arquivo
    - também usado para calcular o tamanho dos regs.
  - Através do índice:
    - **é possível acessar os registros por RRN**
      - mas isso demanda a leitura do arquivo de índice
    - sabe-se onde termina aquele registro

Exercícios

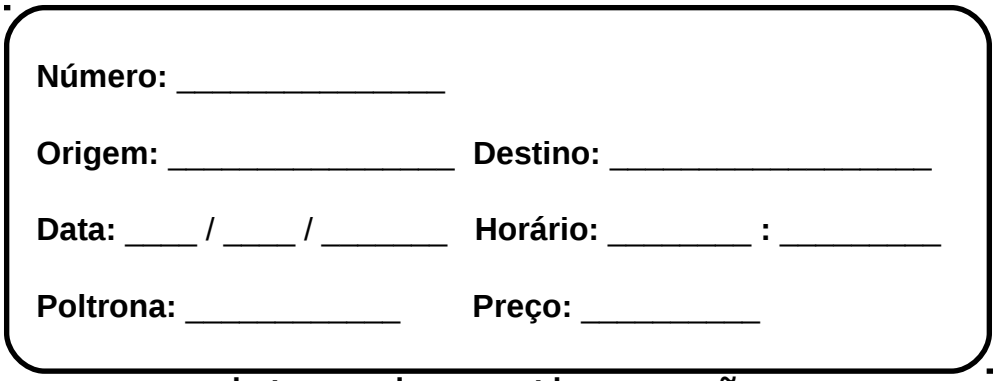
- Seja o seguinte conjunto de campos:



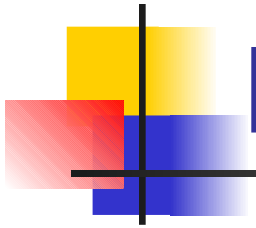
- Considere que campos como estes acima serão armazenados em um arquivo como uma seqüência organizada de caracteres.
- Dê 3 exemplos de realização desses campos, organizados via:
  - comprimento fixo, indicador de comprimento, delimitadores e tags

Exercícios

- Seja o seguinte tipo de registro:

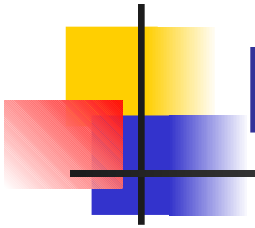


- Considere que registros desse tipo serão armazenados em um arquivo como uma seqüência organizada de caracteres.
- Mostre um arquivo lógico com 3 desses registros, organizados via:
  - no. fixo de campos de tamanho fixo, no. fixo de campos de tamanho variável, delimitadores, indicador de tamanho, e índice
  - OBS: para campos de tamanho variável, assuma uso de delimitadores



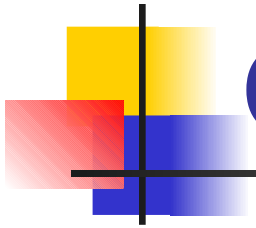
Exercícios

- Elabore um tipo de registro para cadastro de clientes ou funcionários de um banco, e repita os exercícios anteriores.
- Exemplifique uma situação prática onde o uso de registros de tamanho fixo com campos também fixos é apropriado.
- Implemente em linguagem C ANSI o programa **`readstrm`** e sua função **`readfield`**, mostrados nos slides na forma de pseudo-código. Dica: veja (Folk & Zoellick, 1987).



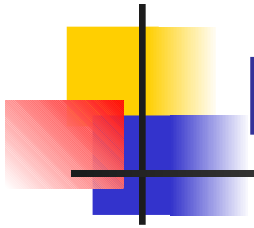
Exercícios

- Incremente os programas do exercício anterior, agora assumindo que os campos estão organizados no arquivo em registros de tamanho variável, também separados por delimitadores (#). Imprima uma linha em branco entre cada registro.
- Modifique o exercício anterior considerando que os registros possuem indicador de tamanho ao invés de delimitadores. Assuma que cada registro no arquivo não possui mais que 256 caracteres e que, portanto, o indicador de tamanho é um caractere (byte) interpretado como um inteiro no início do registro.



Outros Exercícios

- Lista de Exercícios (CoTeia)
- Capítulo 4 (Folk & Zoellick, 1987)



Bibliografia

- **M. J. Folk and B. Zoellick, *File Structures: A Conceptual Toolkit*, Addison Wesley, 1987.**