

Aula: ARQUIVOS

Introdução

- As estruturas vistas anteriormente armazenam as informações na **memória principal** do computador.
 - Nem sempre é conveniente.
- **Problemas:**
 - A informação é perdida;
 - As estruturas de dados são limitadas;
 - Existe uma **quantidade** de informação que pode ser armazenada para resolver o problema.

1

Introdução

- Algumas informações geradas pelos programas precisam ser mantidas para posterior uso
 - Exs: cadastro de alunos, cadastro de vendas de uma empresa, estatísticas ...

2

Arquivos

- Um **arquivo** é armazenado em um dispositivo de memória secundária. Pode ser lido e escrito por um programa.
- Em programação, existem vários tipos de informações que podem ser armazenadas em arquivo

3

Arquivos - Exemplos

Arquivo de inteiros

1
56
0
23

Arquivo de reais

21.5
-2.67
3.75
0.6

Arquivo de vetores de tamanho 4 composto por inteiros

(1, 23, -12, 4)
(56, 0, 65, 2)
(0, 41, -7, 60)

Arquivo de cadeia de caracteres

Casa
Lua
Giz
Rio

Representação Simbólica

4

Arquivos - Exemplos

Arquivo de Registros

Nome					
CPF			RG		
HT 1	HT 2	HT 3	HT 4	HT 5	HT 6
Salário					
FGTS 1.1			FGTS 1.2		
FGTS 2.1			FGTS 2.2		
Nome					
CPF			RG		
HT 1	HT 2	HT 3	HT 4	HT 5	HT 6
Salário					
FGTS 1.1			FGTS 1.2		
FGTS 2.1			FGTS 2.2		
Nome					
CPF			RG		
HT 1	HT 2	HT 3	HT 4	HT 5	HT 6
Salário					
FGTS 1.1			FGTS 1.2		
FGTS 2.1			FGTS 2.2		

Representação Simbólica

CARACTERÍSTICAS

Vetores / Matrizes

- o Armazenam pequena quantidade de informação;
- o Ao término do programa, todas as informações armazenadas são perdidas;
- o Restrito as definições da linguagem de programação;
- o Tamanho da variável deve ser definida antes do uso;

Arquivo

- o Armazenam grandes quantidades de informações;
- o Mantém informações armazenadas em disco;
- o Variável do tipo *file* é um tipo de dado que existe independentemente de qualquer programa e pode ser acessado por outros programas;
- o Depende do espaço disponível no disco;

6

Arquivos

- Na linguagem C:
 - Podem ser criados arquivos:
 - "modo texto"
 - "modo binário"



7

Arquivos - modo texto

- Sequência de caracteres agrupadas em linhas.
- Números são guardados como cadeias de caracteres.
- Cada linha é separada pelo **caracter 10 decimal (LF)** ou o **"\n"**.
- Existe uma **indicação de fim de arquivo** que é enviada pelo DOS e é reconhecida pelo compilador C.

8

Arquivos - modo texto

- Exemplo de arquivo modo texto:

```
Ainda que eu falasse\n
a língua dos homens\n
Que eu falasse\n
a língua dos anjos\n
Sem amor eu nada seria\n
EOF
```

9

Arquivos - modo binário

- As informações são armazenadas de acordo com o **número de bytes** que ocupam na **memória**.
 - 4 bytes para inteiro, 4 para float, 1 para char...
- Qualquer caractere é lido ou gravado sem alteração, mantendo-se a ordem de gravação realizada.
- Não existe indicação de fim de arquivo.

10

Arquivos - modo binário

- Exemplo de arquivo no modo binário:

```
0000 FF D8 FF E1 1D FE 45 78 69 66 00 00 49 49 2A 00
0010 08 00 00 00 09 00 0F 01 02 00 06 00 00 00 7A 00
0020 00 00 10 01 02 00 14 00 00 00 80 00 00 00 12 01
0030 03 00 01 00 00 00 01 00 00 00 1A 01 05 00 01 00
0040 00 00 A0 00 00 00 1B 01 05 00 01 00 00 00 A8 00
0050 00 00 28 01 03 00 01 00 00 00 02 00 00 00 32 01
0060 02 00 14 00 00 00 B0 00 00 00 13 02 03 00 01 00
0070 00 00 01 00 00 00 69 87 04 00 01 00 00 00 C4 00
0080 00 00 3A 06 00 00 43 61 6E 6F 6E 00 43 61 6E 6F
0090 6E 20 50 6F 77 65 72 53 68 6F 74 20 41 36 30 00
00A0 00 00 00 00 00 00 00 00 00 00 00 00 B4 00 00 00
00B0 01 00 00 00 B4 00 00 00 01 00 00 00 32 30 30 34
00C0 3A 30 36 3A 32 35 20 31 32 3A 33 30 3A 32 35 00
00D0 1F 00 9A 82 05 00 01 00 00 00 86 03 00 00 9D 82
00E0 05 00 01 00 00 00 8E 03 00 00 00 90 07 00 04 00
```



Arquivos

- Diferença principal:
 - **Modo texto**: possui caractere de nova linha e de fim de arquivo
 - **Modo binário**: informações armazenadas na forma de bytes, sem marca de final de arquivo ou final de linha.



Arquivos - modo binário

- Arquivos binários podem conter diferentes estruturas manipuladas pelos programas:
 - Vetores
 - Cadeia de caracteres
 - Matrizes
 - **Registros**



13

Arquivos - modo binário

- Em geral, arquivos são formados por uma **coleção de registros**. Cada registro é composto por campos.
 - Um dos campos é considerado **campo-chave** e é o campo que diferencia um registro dos demais.
- Um **sistema de banco de dados** é formado por um ou vários arquivos, com programas para: **inclusão, exclusão, alteração, consultas...**

14

Funções em C (stdio.h)

- **fopen()** = abre um arquivo
- **fclose()** = fecha um arquivo
- **ferror()** = retorna verdadeiro se ocorreu um erro
- **fputc()** = escreve um caracter em um arquivo
- **fgetc()** = lê um caracter de um arquivo
- **fputs()** = escreve uma string em um arquivo
- **fgets()** = lê uma string de um arquivo
- **fwrite()** = escreve uma estrutura (struct) em um arquivo
- **fread()** = lê uma estrutura (struct) de um arquivo
- **fseek()** = posiciona o arquivo em um byte específico
- **feof()** = retorna verdadeiro se atingiu o final do arquivo
- **rewind()** = coloca o ponteiro do arquivo no seu início
- **remove()** = apaga um arquivo
- **fflush()** = descarrega o conteúdo de um arquivo

15

Funções em C (stdio.h)

- A biblioteca também define a estrutura de arquivo a ser utilizada:
 - `FILE`
- Macros:
 - `NULL` = 0 (define um ponteiro nulo)
 - `EOF` = -1 (retorno da função)
 - `FOPEN_MAX` = máximo de arqs que podem ser abertos
 - `SEEK_SET` = 0 (início do arquivo)
 - `SEEK_CUR` = 1 (posição atual)
 - `SEEK_END` = 2 (final do arquivo)

16

Ponteiro para Arquivo

- Definição de variável do tipo arquivo:
 - `FILE *arq;`
- **arq** é uma variável ponteiro capaz de identificar um arquivo no disco.
 - aponta para informações do arquivo: nome, status e posição do arquivo.

17

Criar ou Abrir Arquivo

- Definição de variável do tipo arquivo:
`FILE *arq;`
- Função `fopen(nome_arquivo, modo_abertura)`
 - abre ou cria um arquivo, retornando o ponteiro apontado para o mesmo.
- Ex:
`arq = fopen(nome_arquivo, modo_abertura)`

18

Criar ou Abrir Arquivo

`arq = fopen(nome_arquivo, modo_abertura)`

nome_arquivo: `string` contendo o nome do arquivo para abrir ou criar, podendo incluir um `path`

modo_abertura: `string` que representa como o arquivo será aberto: escrita, leitura ...

19

Criar ou Abrir Arquivo

```
arq = fopen(nome_arquivo, modo_abertura)
```

Modo de Abertura:

w	Cria um arquivo texto para escrita (apaga se ele já existir)
r	Abre um arquivo texto para leitura
a	Abre um arquivo texto para anexar novos dados (no final)
wb	Cria um arquivo binário para escrita (apaga se ele já existir)
rb	Abre um arquivo binário para leitura
ab	Abre um arquivo binário para anexar novos dados (no final)

20

Criar ou Abrir Arquivo

Outros modos de Abertura:

w+	Cria um arquivo texto para escrita e leitura (apaga se o arquivo já existir)
r+	Abre um arquivo texto para leitura e escrita (o arquivo deve existir)
a+	Anexa novos dados ou cria um arquivo texto para leitura e escrita (se o arquivo não existir, cria o arquivo)
wb+	Cria um arquivo binário para escrita e leitura (apaga se o arquivo já existir)
rb+	Abre um arquivo binário para leitura e escrita (o arquivo deve existir)
ab+	Anexa novos dados ou cria um arquivo binário para leitura e escrita (se o arquivo não existir, cria o arquivo)

21

Criar ou Abrir Arquivo

- Exemplo:

```
FILE *arq1, *arq2;
```

```
arq1 = fopen("arquivo1.txt", "w");
```

```
arq2 = fopen("texto.txt", "a+");
```

22

Criar ou Abrir Arquivo

- Importante verificar se o arquivo foi criado ou aberto com sucesso!

```
FILE *arq1, *arq2;
```

```
arq1 = fopen("arquivo1.txt", "w");
```

```
if (arq1==NULL) {
```

```
    printf("Erro na criação do arquivo);
```

```
    return(0);
```

```
}
```

```
else { ...}
```

23

Fechar Arquivo

- **Função fclose():** fecha um arquivo. É importante que todo arquivo aberto seja fechado antes de terminar o programa!

int **fclose(arq);**

arq: ponteiro para um arquivo obtido pela função fopen()

24

Arquivo - modo texto

- **Gravação e Leitura:**
 - **função fputs():** escreve uma cadeia de caracteres em um arquivo
`fputs(cadeia_caracteres, arq)`
 - **função fgets():** lê uma cadeia de caracteres até **tam** ou até encontrar **\n**
`fgets(cadeia_caracteres, tam, arq)`

EXEMPLO

25

Arquivo - modo binário

- **Gravação:**
 - **função fwrite():** escreve qualquer tipo de dados em um arquivo
`fwrite(&dados, tamanho, qtde, arq)`
- **Leitura:**
 - **função fread():** lê qualquer tipo de dados em um arquivo
`fread(&dados, tamanho, qtde, arq)`

dados = struct, vetor, matriz ou uma variável

26

Arquivo - modo binário

Exemplos:

`fwrite(&agenda, sizeof(agenda), 1, arq)`

`fwrite(&x, sizeof(int), 1, arq)`

`fread(&agenda, sizeof(agenda), 1, arq)`

`fread(&x, sizeof(int), 1, arq)`

EXEMPLO

27

Arquivo - modo binário

Matriz em arquivo:

[EXEMPLO2](#)

28

Função fseek

- Modifica a posição do ponteiro no arquivo

`fseek(arq, qtde, origem)`

qtde: número de bytes a partir da **origem** que deve ser feito o deslocamento do ponteiro

origem: uma das macros definidas para `stdio.h`:

- `SEEK_SET` = 0 (início do arquivo)
- `SEEK_CUR` = 1 (posição atual)
- `SEEK_END` = 2 (final do arquivo)

29

Função fseek

[EXEMPLO3](#)

30

Exercício 1

- Faça um programa que irá funcionar como uma agenda, contendo nome, telefone e email de cada contato. O programa deve possuir funções para criar a agenda, permitir adicionar novos nomes e fazer buscas de dois tipos: 1) mostrar a lista toda de nomes e 2) buscar dados a partir de um nome fornecido.

31

Exercício 2

- Faça um programa que gera um arquivo contendo uma lista de cidades e suas temperaturas, como ilustrado abaixo. O programa deve possuir funções para criar o arquivo e para ler os dados do arquivo. Faça uma função que calcula a temperatura média das cidades registradas no arquivo e mostra quais cidades possuem temperatura acima da média.

• Exemplo:

São Carlos	23.2	saída da função:
Piracicaba	25.4	média = 21.7
São Paulo	20.8	acima da média: São Carlos e
Curitiba	17.5	Piracicaba