

LABORATÓRIO DE PROGRAMAÇÃO II

ARQUIVOS

Bibliografia:

Cap 8 - Estudo Dirigido de Linguagem C

José Augusto Manzano, Ed. Érica



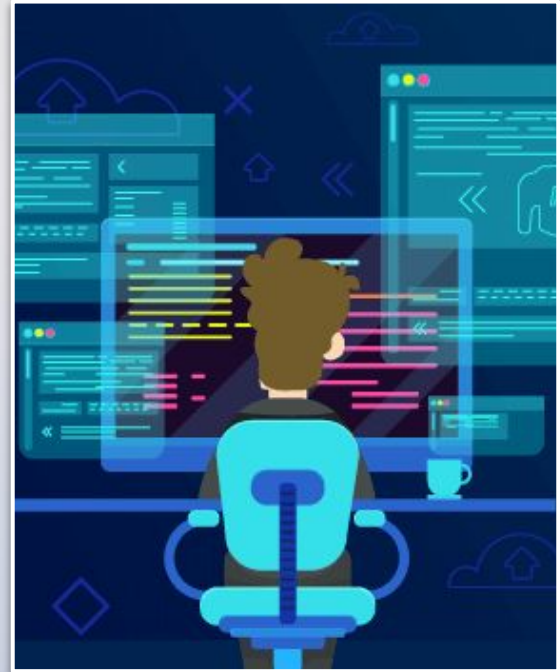
Agenda

Objetivos:

Desenvolver a habilidade de fazer leitura e escrita de arquivos de texto em disco.

Agenda:

- Definição de Arquivo
- Características
- Modos de Acesso
- Arquivo Texto
- Operações com Arquivo Texto
- Exemplos
- Resumo
- Atividade



Definição de Arquivo

Um arquivo é um conjunto de dados, organizados sequencialmente, armazenados em um dispositivo de memória permanente (disco rígido, CD, USB-Drive).





Características

Dados são armazenados na memória permanente. O tamanho do arquivo não é fixo e nem predefinido.

Pode armazenar grandes quantidades de dados. Dados podem ser consultados a qualquer momento. O acesso ao disco é mais lento em relação à memória volátil.

dados são armazenados como caracteres

Arquivos em disco podem ser dos tipos **texto** ou **binário**.

são representados por uma sequência de bytes sem o conceito de quebra de linha. Ele armazena o **dado** literal






Modos de acesso



O acesso pode ser sequencial, isto é, registros são lidos um a um.

O acesso pode ser aleatório (início ou fim do arquivo, ou posicionamento atual do **ponteiro**).



um ponteiro é uma variável que contém um endereço de memória





Arquivo Texto



Exemplos: txt, cpp, bat

Caracteres podem ser agrupados em linhas

Pode ser lido e compreendido por pessoas

Pode ser editado por editores de texto convencionais

Já que os dados são convertidos para caracteres, o arquivo tende a ser maior que o tipo binário (ex: número)

Menor velocidade de manipulação em relação aos binários



Operações com Arquivos Texto

Sintaxe:

A estrutura está presente na biblioteca stdio.h

```
FILE *<ponteiro>;
```

Abertura:

```
<ponteiro> = fopen("<nomeArquivo>.txt", "<tipo de abertura>");
```

(A extensão .txt pode ser omitida)

Tipo de abertura:

"r": *read*, ou seja, apenas leitura

"w": *write*, cria um arquivo novo, para ser trabalhado. Caso haja arquivo anterior, ele é apagado

"a": *append*, abre o arquivo permitindo adicionar informações a partir do seu final. Se não existir, o arquivo é criado

Fechamento (uma vez aberto, o arquivo deve ser fechado):

```
fclose(<ponteiro>);
```



Operações com Arquivos Texto

Abertura:

```
FILE * fopen(char*, char*) //protótipo  
<ponteiro>=fopen("<nomeArquivo>", "<tipo de abertura>");
```

Abre o arquivo para gravação, adição, ou leitura de dados

- - Se não puder abrir o arquivo, retorna ponteiro nulo (NULL)
- - O sistema operacional busca o arquivo pelo nome
- - O sistema operacional prepara o *buffer* para armazenar o arquivo
- O sistema operacional abre uma fila de *bytes* e liga um arquivo em disco à fila

Operações com Arquivos Texto



```
<ponteiro>=fopen("<nomeArquivo>", "<tipo de abertura>");
```

| Tipo | Significado | Tipo | Significado |
|----------|-----------------------------|-----------|------------------------------------|
| w | Cria para gravação | w+ | Cria para Leitura/Gravação |
| a | Anexa ou cria para gravação | a+ | Abre ou Cria para Leitura/Gravação |
| r | Leitura | r+ | Abre para Leitura/Gravação |

Na operação de escrita (**write**):

O sistema operacional modifica o conteúdo do arquivo no **buffer**, e então ele é transferido para o disco quando o arquivo é fechado.

Área de armazenamento temporária



Operações com Arquivos Texto



ponteiro>=fopen("<nomeArquivo>", "<tipo de abertura>");

| Tipo | Significado | Tipo | Significado |
|----------|-----------------------------|-----------|------------------------------------|
| w | Cria para gravação | w+ | Cria para Leitura/Gravação |
| a | Anexa ou cria para gravação | a+ | Abre ou Cria para Leitura/Gravação |
| r | Leitura | r+ | Abre para Leitura/Gravação |

Na operação de leitura (**read**):

O sistema operacional recupera o arquivo solicitado.

O arquivo, ou parte dele, é armazenado no *buffer*.



Operações com Arquivos Texto



- Fechamento: `int fclose(FILE*)` //protótipo
`fclose(<ponteiro>);`

Ao fechar, as modificações realizadas no *buffer* são transferidas para o disco e a área do *buffer* é liberada.

Retorna zero se o arquivo for fechado com sucesso.



Operações com Arquivos Texto

| Modo | Significado |
|------|---|
| "r" | Abre um arquivo texto para leitura. O arquivo deve existir antes de ser aberto. |
| "w" | Abrir um arquivo texto para gravação. Se o arquivo não existir, ele será criado. Se já existir, o contendo anterior será destruído. |
| "a" | Abrir um arquivo texto para gravação. Os dados serão adicionados no fim do arquivo ("append"), se ele já existir, ou um novo arquivo será criado, no caso de arquivo não existente anteriormente. |
| "r+" | Abre um arquivo texto para leitura e gravação. O arquivo deve existir e pode ser modificado. |
| "w+" | Cria um arquivo texto para leitura e gravação. Se o arquivo existir, o conteúdo anterior será destruído. Se não existir, será criado. |
| "a+" | Abre um arquivo texto para gravação e leitura. Os dados serão adicionados no fim do arquivo se ele já existir, ou um novo arquivo será criado, no caso de arquivo não existente anteriormente. |



Exemplos

(Abra os arquivos abaixo)

✓ Manipulação de Caracteres

- **a_criarArq - w - Caractere.c**
- **b_lerArq - r - Caractere.c**
- **c_anexarArq - a - Caractere.c**

Limitação: teste gravar uma frase. O que acontece?






Exemplos



(Abra os arquivos abaixo)

- ✓ Manipulação de Strings
 - **d_criarArq - w - String.c**
 - **e_lerArq - r - String.c**
 - **f_anexarArq - a - String.c**
- 





Exemplos

(Abra os arquivos abaixo)

✓ Manipulação de Texto Formatado

- **g_criarArq - w – Formatado.c**
- **h_lerArq - r – Formatado.c**
- **i_anexarArq - a – Formatado.c**



Resumo

Modos de Abertura

| Função | Significado | Modo | Significado | Modo | Significado |
|---------------|---------------|----------|-----------------------------|-----------|------------------|
| fopen | Abre arquivo | w | Cria para gravação | w+ | Leitura/Gravação |
| | | a | Anexa ou cria para gravação | a+ | Leitura/Gravação |
| | | r | Leitura | r+ | Leitura/Gravação |
| fclose | Fecha arquivo | - | - | - | - |



Resumo

Funções para Manipulação de Dados

| Dado | Gravação | Leitura | EOF |
|------------------|-----------------------------|----------------------------|-----------------------------|
| CARACTERE | fputc(c, arq) | fgetc(arq) | !feof(arq) ou != EOF |
| STRING | fputs(s, arq) | fgets(s, tamanho, arq) | fgets(s, size, arq) != NULL |
| FORMATADO | fprintf(arq, formato, dado) | fscanf(arq, formato, dado) | !feof(arq) ou != EOF |





Atividade

1. Faça uma solução em Linguagem C, usando modularidade, para escrever, ler e adicionar um vetor de caracteres, caractere por caractere, em um arquivo texto no disco.

Deve ser exibido o seguinte menu ao usuário:

1. Criar arquivo
2. Ler arquivo
3. Adicionar dados
4. Sair





Atividade

2. Faça como no Exercício 1. Entretanto, o vetor de caracteres deverá ser escrito/lido de uma só vez, e não caractere por caractere

Deve ser exibido o seguinte menu ao usuário:

1. Criar arquivo
2. Ler arquivo
3. Adicionar dados
4. Sair





Atividade

3. Faça como no Exercício 2. Entretanto, ao invés de um vetor de caracteres, um registro deverá ser escrito/lido.

```
struct aluno{  
    char nome[30];  
    int matricula;  
};
```

Menu ao usuário:

1. Criar arquivo
2. Ler arquivo
3. Adicionar registro
4. Sair

