

Manipulação de Arquivos

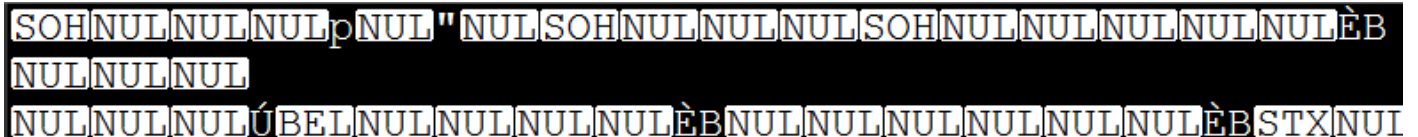
Arquivos correspondem a unidades de armazenamento, tipicamente gravados em disco magnético.

Dentre os arquivos em disco, existem dois tipos:

Texto (caracteres)

```
Entenda programação;1001;139.45
Banco de Dados a seu Dispor;1002;220.00
Oi C;1003;79.00
```

Binário (bytes).



Para utilizar um arquivo, é preciso associá-lo a uma variável lógica (stream) e, então, manipulá-la.

Lógica Básica:

- 1) Abrir ou criar o arquivo, associando o nome físico do arquivo ao seu nome lógico.
- 2) Manipular os dados do arquivo: consulta, inclusão, exclusão, alteração.
- 3) Fechar o arquivo.

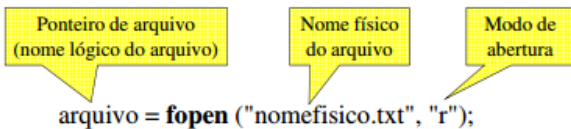
Ponteiro de Arquivo

O ponteiro de arquivo serve para referenciar o arquivo a ser tratado pelo programa. O ponteiro não aponta diretamente para o arquivo; contém as seguintes informações sobre o mesmo: nome, situação (aberto ou fechado) e posição atual sobre o arquivo. Para se definir uma variável ponteiro de arquivo, usa-se a seguinte declaração:

```
FILE *Arquivo;
```

Arquivo Texto

1) Abertura

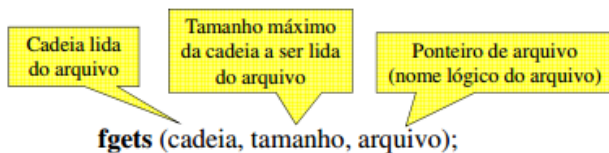


Modo de Abertura

r	Abre um arquivo texto existente para leitura
w	Cria um arquivo texto para escrita
a	Abre um arquivo texto para inserção no final
r+	Abre um arquivo texto existente para leitura e escrita

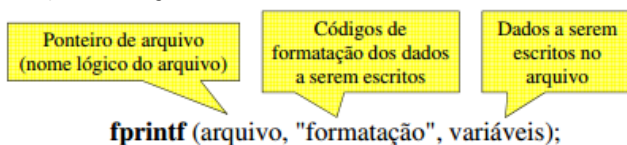
```
if ((arquivo = fopen("teste.txt", "r")) == NULL)
```

2) Leitura



```
FILE *arq; char  
cadeia[81];  
fgets(cadeia, 80, arq)
```

3) Gravação



```
FILE *arq; char  
char titulo[31]; int codlivro; float preco;  
fprintf(arq, "%-30s%-4d %.2f\n", titulo,  
        codlivro, preco);
```

4) Fim de Arquivo

`feof(arquivo)` -> retorna 0 se não está no final de arquivo, senão $\neq 0$.

```
while (!feof(arq))
```

5) Fechar Arquivo

```
Fclose(arq)
```

Arquivo Binário

1) Abertura

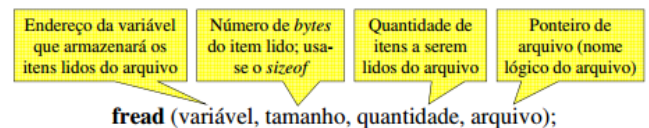


Modo de Abertura

rb	Abre um arquivo texto existente para leitura
wb	Cria um arquivo texto para escrita
ab	Abre um arquivo texto para inserção no final
r+b	Abre um arquivo texto existente para leitura e escrita

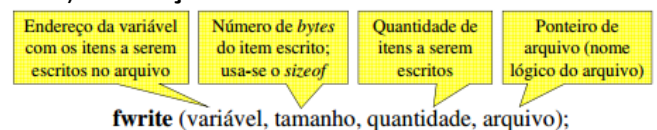
```
if ((arquivo = fopen("nomefisico.dat", "r+b") ==  
NULL)
```

2) Leitura



```
FILE *arq; char  
typedef struct { char titulo[30]; int  
                codlivro; float preco; }  
                reglivro;  
fread(&livro, sizeof(livro), 1, arq);
```

3) Gravação



```
FILE *arq; char  
typedef struct { char titulo[30]; int  
                codlivro; float preco; }  
                reglivro;  
fwrite(&livro, sizeof(livro), 1, arq);
```

4) Fim de Arquivo

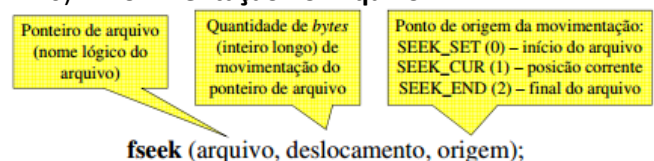
`feof(arquivo)` -> retorna 0 se não está no final de arquivo, senão $\neq 0$.

```
while (!feof(arq))
```

5) Fechar Arquivo

```
Fclose(arq)
```

6) Movimentação no Arquivo



```
fseek(arq, sizeof(reglivro), 1)
```

Outras Funções:

rewind (arquivo);

O comando rewind() faz com que o ponteiro de um arquivo aponte para o início do mesmo. Para tanto, deve ser passado o nome lógico do arquivo aberto.

posicao = ftell (arquivo);

A função ftell() retorna a posição corrente do ponteiro de um arquivo binário, ou seja, o número de bytes desde o início do arquivo. Para tanto, deve ser passado o nome lógico do arquivo aberto.

ferror (arquivo);

A função ferror() retorna verdadeiro se ocorreu um erro durante a última operação com o arquivo; caso contrário, retorna falso. Para tanto, deve ser passado o nome lógico do arquivo aberto. Como cada "operação de arquivo" modifica a condição de erro, a função deve ser chamada logo após a operação a ser avaliada.

fflush (arquivo);

A função fflush() escreve o conteúdo do buffer em um arquivo, esvaziando-o. Para tanto, deve ser passado o nome lógico do arquivo aberto para escrita. Se a limpeza do buffer ocorrer devidamente, a função retorna o valor 0; caso contrário, retorna EOF.

Pode ser usada para limpar buffer da entrada padrão – fflush (stdin) - ou liberar buffer de escrita para a saída padrão - fflush (stdout) .

remove (nome_arquivo);

A função remove() remove um arquivo. Para tanto, deve ser passado o nome físico do arquivo fechado. Se a remoção do arquivo ocorrer devidamente, a função retorna o valor 0; caso contrário, retorna um valor diferente de zero.

rename (nome_atual, nome_novo);

A função rename() renomeia um arquivo. Para tanto, devem ser passados o nome físico atual do arquivo fechado e o novo nome físico do mesmo. Se a renomeação do arquivo ocorrer devidamente, a função retorna o valor 0; caso contrário, retorna um valor diferente de zero.

A função também serve para mover um arquivo.

É preciso incluir a nova localização como parte do novo nome do arquivo.