Arquivos em C

Por: Eduardo Casavella

Arquivos

Os arquivos permitem gravar os dados de um programa de forma permanente em mídia digital.

Vantagens de utilizar arquivos

- •Armazenamento permanente de dados: as informações permanecem disponíveis mesmo que o programa que as gravou tenha sido encerrado, ou seja, podem ser consultadas a qualquer momento.
- •Grande quantidade dados pode ser armazenada: A quantidade de dados que pode ser armazenada depende apenas da capacidade disponível da mídia de armazenamento. Normalmente a capacidade da mídia é muito maior do que a capacidade disponível na memória RAM.
- •Acesso concorrente: Vários programas podem acessar um arquivo de forma concorrente.

A linguagem C trata os arquivos como uma sequência de bytes. Esta sequência pode ser manipulada de várias formas e para tanto, existem funções em C para criar, ler e escrever o conteúdo de arquivos independente de quais sejam os dados armazenados.

Tipos de arquivos

Em C trabalhamos com dois tipos de arquivos:

1) Arquivo texto: Armazena caracteres que podem ser mostrados diretamente na tela ou modificados por um editor de texto.

Exemplos de arquivos texto: documentos de texto, código fonte C, páginas XHTML.

2) Arquivo binário é uma sequência de bits que obedece regras do programa que o gerou.

Exemplos: Executáveis, documentos do Word, arquivos compactados.

O ponteiro para arquivo

Em C, o arquivo é manipulado através de um ponteiro especial para o arquivo.

A função deste ponteiro é “apontar” a localização de um registro.

Sintaxe:
FILE < *ponteiro >

O tipo FILE está definido na biblioteca stdio.h.

Exemplo de declaração de um ponteiro para arquivo em C:

FILE *pont_arq;

Lembrando que FILE deve ser escrito em letras maiúsculas.

Operações com arquivos do tipo texto

Abertura de arquivos.

Para trabalhar com um arquivo, a primeira operação necessária é abrir este arquivo.

Sintaxe de abertura de arquivo:
< ponteiro > = fopen(“nome do arquivo”,”tipo de abertura”);

A função fopen recebe como parâmetros o nome do arquivo a ser aberto e o tipo de abertura a ser realizado.

Depois de aberto, realizamos as operações necessárias e fechamos o arquivo.

Para fechar o arquivo usamos a função fclose.

Sintaxe de fechamento de arquivo
fclose< ponteiro >;

Lembrando que o ponteiro deve ser a mesma variável ponteiro associada ao comando de abertura de arquivo.

Tipos de abertura de arquivos

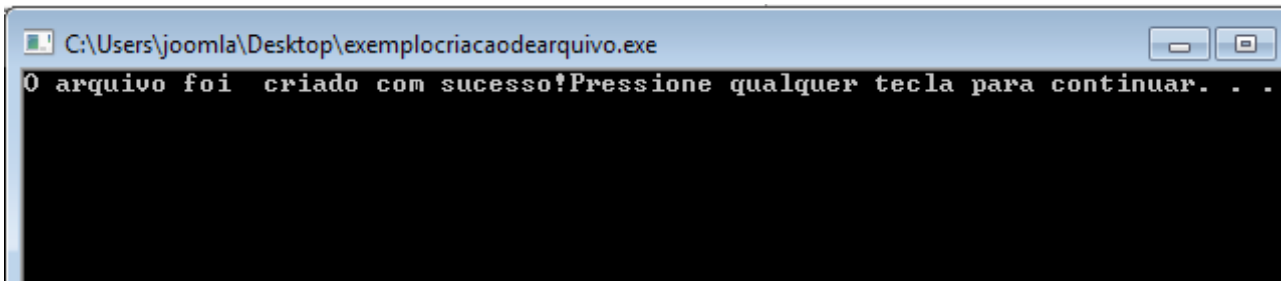
r: Permissão de abertura somente para leitura. É necessário que o arquivo já esteja presente no disco.

w: Permissão de abertura para escrita (gravação). Este código cria o arquivo caso ele não exista, e caso o mesmo exista ele recria o arquivo novamente fazendo com que o conteúdo seja perdido. Portanto devemos tomar muito cuidado ao usar esse tipo de abertura.

a: Permissão para abrir um arquivo texto para escrita(gravação), permite acrescentar novos dados ao final do arquivo. Caso não exista, ele será criado.
Exemplo de criação de arquivos.

```c
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
   // criando a variável ponteiro para o arquivo
   FILE *pont_arq;

   //abrindo o arquivo
   pont_arq = fopen("arquivo.txt", "a");

   // fechando arquivo
   fclose(pont_arq);

   //mensagem para o usuário
   printf("O arquivo foi  criado com sucesso!");

   system("pause");
   return(0);
}
```

Tela de execução



Tela de execução do exemplo criação de arquivo

Problemas na abertura de arquivos

Na prática, nem sempre é possível abrir um arquivo. Podem ocorrer algumas situações que impedem essa abertura, por exemplo:

- Você está tentando abrir um arquivo no modo de leitura, mas o arquivo não existe;
- Você não tem permissão para ler ou gravar no arquivo;
- O arquivo está bloqueado por estar sendo usado por outro programa.

Quando o arquivo não pode ser aberto a função fopen retorna o valor NULL.

É altamente recomendável criar um trecho de código a fim de verificar se a abertura ocorreu com sucesso ou não.

Exemplo:

```
if (pont_arq == NULL)
```

```
{
      printf("ERRO! O arquivo não foi aberto!\n");
}
else
    {
       printf("O arquivo foi aberto com sucesso!");
    }
```

Gravando dados em arquivos

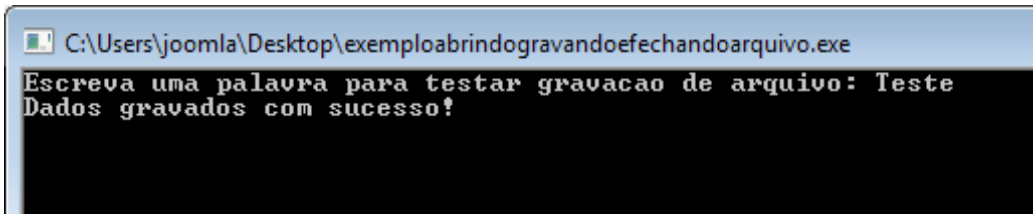
A função **fprintf** armazena dados em um arquivo. Seu funcionamento é muito semelhante ao printf, a diferença principal é a existência de um parâmetro para informar o arquivo onde os dados serão armazenados.

Sintaxe:

fprintf(nome_do_ponteiro_para_o_arquivo, “%s”,variavel_string)

```c
//Exemplo: Abrindo, gravando e fechando arquivo
#include <stdio.h>
#include <conio.h>

int main(void)
{
   FILE *pont_arq; // cria variável ponteiro para o arquivo
   char palavra[20]; // variável do tipo string

   //abrindo o arquivo com tipo de abertura w
   pont_arq = fopen("arquivo_palavra.txt", "w");

   //testando se o arquivo foi realmente criado
   if(pont_arq == NULL)
   {
       printf("Erro na abertura do arquivo!");
       return 1;
   }

   printf("Escreva uma palavra para testar gravacao de arquivo: ");
   scanf("%s", palavra);

   //usando fprintf para armazenar a string no arquivo
   fprintf(pont_arq, "%s", palavra);

   //usando fclose para fechar o arquivo
   fclose(pont_arq);

   printf("Dados gravados com sucesso!");

   getch();
   return(0);
}
```

Tela de execução



Tela de execução do exemplo abrindo, gravando e fechando arquivo

Leitura de arquivos

Leitura caracter por caracter – Função getc()

Faz a leitura de um caracter no arquivo.

Sintaxe:

getc(ponteiro_do_arquivo);

Para realizar a leitura de um arquivo inteiro caracter por caracter podemos usar getc dentro de um laço de repetição.

```
do

{

	//faz a leitura do caracter no arquivo apontado por pont_arq

	c = getc(pont_arq);



	//exibe o caracter lido na tela

	printf("%c" , c);



}while (c != EOF);
```

A leitura será realizada até que o final do arquivo seja encontrado.

Leitura de strings – Função fgets()

É utilizada para leitura de strings em um arquivo. Realiza a leitura dos caracteres até o final da linha quando encontra o caracter \n. A leitura é efetuada de tal forma que a string lida é armazenada em um ponteiro do tipo char. A função pode ser finalizada quando encontrar o final do arquivo, neste caso retorna o endereço da string lida. Se ocorrer algum erro na leitura do arquivo, a função retorna NULL.

```
1//Leitura de arquivo
2#include <stdio.h>
3#include <conio.h>
4
5int main(void)
6{
7   FILE *pont_arq;
8   char texto_str[20];
9
10  //abrindo o arquivo_frase em modo "somente leitura"
11  pont_arq = fopen("arquivo_palavra.txt", "r");
12
13  //enquanto não for fim de arquivo o looping será executado
14  //e será impresso o texto
15  while(fgets(texto_str, 20, pont_arq) != NULL)
16     printf("%s", texto_str);
17
18
19  //fechando o arquivo
20  fclose(pont_arq);
```

```
21      getch();
22      return(0);
23  }
```