• **Manipulação de arquivos em Linguagem C**

O sistema de entrada e saída do ANSI C é composto por uma série de funções, cujos protótipos estão reunidos em **stdio.h** .
Todas estas funções trabalham com o conceito de "ponteiro de arquivo".
Este não é um tipo propriamente dito, mas uma definição usando o comando typedef.
Esta definição também está no arquivo **stdio.h**.

Podemos declarar um ponteiro de arquivo da seguinte maneira: FILE *p;

Nesta declaração **p** será então um ponteiro para um arquivo.

É usando este tipo de ponteiro que vamos poder manipular arquivos no C.

• **Função para abertura/criação de um arquivo: fopen**

Esta é a função de abertura/criação de arquivos.

Seu protótipo é: FILE *fopen (char *nome_do_arquivo,char *modo);

O nome_do_arquivo determina qual arquivo deverá ser aberto.
Este nome deve ser válido no sistema operacional que estiver sendo utilizado.

O modo de abertura diz à função **fopen()** que tipo de uso você vai fazer do arquivo.

A tabela abaixo mostra os valores de modo válidos:

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 conteúdo 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.
"rb"	Abre um arquivo binário para leitura. Igual ao modo "r" anterior, só que o arquivo é binário.
"wb"	Cria um arquivo binário para escrita, como no modo "w" anterior, só que o arquivo é binário.
"ab"	Acrescenta dados binários no fim do arquivo, como no modo "a" anterior, só que o arquivo é binário.
"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.
"r+b"	Abre um arquivo binário para leitura e escrita. O mesmo que "r+" acima, só que o arquivo é binário.
"w+b"	Cria um arquivo binário para leitura e escrita. O mesmo que "w+" acima, só que o arquivo é binário.
"a+b"	Acrescenta dados ou cria uma arquivo binário para leitura e escrita. O mesmo que "a+" acima, só que o arquivo é binário

• **Função para fechar um arquivo aberto: fclose**

Quando acabamos de usar um arquivo que abrimos, devemos fechá-lo.
Para tanto usa-se a função **fclose()** cujo protótipo é: int fclose (FILE *p);
O ponteiro **p** passado à função **fclose()** determina o arquivo a ser fechado.
A função retorna zero no caso de sucesso.

• **Exemplificando**

```
/* bibliotecas utilizadas */
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

/* Corpo do programa */
int main ()
{
       /* Variáveis locais */
       FILE   *arq;

       printf ("Teste com arquivos texto");

       arq = fopen ("Cadastro.txt" , "wt");

       if ( arq == NULL )
       {
              printf ("\nImpossível criar arquivo Cadastro.txt!");
              printf ("\nPressione qualquer tecla.");
       }
       else
          if ( fclose(arq) != 0 )
                 printf("\nProblemas ao fechar arquivo Cadastro.txt!");
          else
          {
                 printf("\nArquivo Cadastro.txt fechado e criado com sucesso!");
                 printf("\nConfirmando:\n");
                 system ("dir cadastro.txt");
          }

       getch();

       return 0;
}
```

• **Função para escrever STRING um arquivo aberto: fputs**

Protótipo: char *fputs (char *str,FILE *fp);

Escreve uma string num arquivo.

• **Função para ler STRING um arquivo aberto: fgets**

Protótipo: char *fgets (char *str, int tamanho, FILE *fp);

A função recebe 3 argumentos: a string a ser lida, o limite máximo de caracteres a serem lidos e o ponteiro para FILE, que está associado ao arquivo de onde a string será lida.
A função lê a string até que um caractere de nova linha seja lido ou *tamanho-1* caracteres tenham sido lidos.
Se o caracter de nova linha ('\n') for lido, ele fará parte da string, o que não acontecia com gets.
A string resultante sempre terminará com '\0' (por isto somente *tamanho-1* caracteres, no máximo, serão lidos).

• **Função para checar se a última operação de input/output foi bem sucedida: ferror**

Protótipo: int ferror (FILE *fp);

A função retorna zero, se nenhum erro ocorreu e um número diferente de zero se algum erro ocorreu durante o acesso ao arquivo.
ferror() se torna muito útil quando queremos verificar se cada acesso a um arquivo teve sucesso, de modo que consigamos garantir a integridade dos nossos dados.

Na maioria dos casos, se um arquivo pode ser aberto, ele pode ser lido ou gravado.
Porém, existem situações em que isto não ocorre.
Por exemplo, pode acabar o espaço em disco enquanto gravamos, ou o disco pode estar com problemas e não conseguimos ler, etc.

```
/* bibliotecas utilizadas */
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>


/* Corpo do programa */
int main ()
{
       /* Variáveis locais */
       FILE   *arq;

       printf ("Teste com arquivos texto");

       arq = fopen ("Cadastro.txt" , "wt");

       if ( arq == NULL )
       {
              printf ("\nImpossível criar arquivo Cadastro.txt!");
              printf ("\nPressione qualquer tecla.");
       }
       else
          if ( fclose(arq) != 0 )
                 printf("\nProblemas ao fechar arquivo Cadastro.txt!");
          else
          {
                 printf("\nArquivo Cadastro.txt fechado e criado com sucesso!");
                 printf("\nConfirmando:\n");
                 system ("dir cadastro.txt");
          }

       getch();

       return 0;
}
```

- **Função para gravar UM ÚNICO CARACTERE no arquivo: putc**

Protótipo: int putc (int ch,FILE *fp);

O programa a seguir lê um caractere e grava-o em um arquivo em disco (o arquivo arquivo.txt, que será aberto no diretório corrente).

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

int main ()
{
 FILE *fp;
 char caract;

 fp = fopen("arquivo.txt","w");   /* Arquivo ASCII, para escrita */
 if(fp==NULL)
 {
     printf( "Erro na abertura do arquivo");
```

```
     getch();
 }
 else
 {
      printf("Entre com o caractere a ser gravado no arquivo:");
      caract = getche();
     putc( caract, fp); /* Grava caract em arquiovo.txt */
     fclose(fp);

      printf ("\n\n\n");
      system ("type arquivo.txt");
      getch();
 }

 return 0;
}
```

- **Melhorando o programa que grava STRINGS em Cadastro.txt**

```
/* bibliotecas utilizadas */
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>


/* Variáveis globais */
FILE   *arq;
char   STRING[100];
char   opc='s';

/* Corpo do programa */
int main ()
{
      printf ("Teste com arquivos texto");

      arq = fopen ("Cadastro.txt" , "wt");

      if ( arq == NULL )
      {
            printf ("\nImpossível criar arquivo Cadastro.txt!");
            printf ("\nPressione qualquer tecla.");
      }
      else
      {
        /* Grava os textos digitados pelo usuário */
         while ( opc != 'n' && opc != 'N' )
         {
                    printf ("\nDigite o texto a ser gravado no arquivo: ");
                    fflush (stdin);
                    gets (STRING);
                    fputs(STRING, arq);

                    if ( ferror(arq) != 0 )
                    {
                     printf("\nProblemas ao gravar o texto [%s] arquivo Cadastro.txt!", STRING);
                          printf ("\nPressione qualquer tecla.");
                    }

                    putc( '\n', arq); /* Grava \n em Cadastro.txt */

                    if ( ferror(arq) != 0 )
                    {
                          printf("\nProblemas ao gravar o texto [%s] arquivo Cadastro.txt!", STRING);
                          printf ("\nPressione qualquer tecla.");
                    }
```

```
                do
                {
                   printf ("\nDeseja gravar outro texto [s/n]");
                   fflush (stdin);
                   opc = getche();
                }
                while ( opc!='s' && opc !='S' && opc != 'n' && opc != 'N' );
        }

        /* Não deseja mais gravar textos no arquivo */
        if ( fclose(arq) != 0 )
                printf("\nProblemas ao fechar arquivo Cadastro.txt!");
         else
         {
                printf("\nArquivo Cadastro.txt fechado e criado com sucesso!");
                printf("\nConteúdo de Cadastro.txt:\n");
                system ("type cadastro.txt");
         }
       }
       getch();

       return 0;
}
```