

MODALIDADE:	Educação e Formação de Adultos (EFA)	EFA NS (Profissional)	
CURSO:	Programador/a Informático/a		
UFCD:	Programação em C/C++ - avançado	CÓDIGO UFCD:	0810
FORMADOR/A:	Bruno Silva	DATA:	

## OBJETIVOS

- Saber como trabalhar com ficheiros em C

Manipular ficheiros em C é bastante simples e envolve 3 fases:

1. **Abrir o ficheiro**
2. **Ler e/ou escrever do/no ficheiro**
3. **Fechar o ficheiro**

## FASE 1 – Abrir o ficheiro

Para ler um ficheiro vamos usar a função **fopen()**. Esta função vai *receber como parâmetros*:

1. o nome do ficheiro (ou a localização do ficheiro);
2. o modo de abertura do ficheiro e devolve um ponteiro do tipo FILE (em caso de erro retorna NULL). *(mas existem **mais métodos de abertura**)*

### Exemplo:

```
FILE *fp; /* inicialização do ponteiro fp do tipo ficheiro */
```

```
fp = fopen( "ficheiro.txt" , "r");
```

No código anterior, verificamos que o **2º parâmetro da função fopen()** tinha a letra **"r"**. **"r"** significa que o ficheiro é aberto apenas para **leitura** (read). Caso se pretenda abrir o ficheiro para escrita teríamos utilizado a letra **"w"** em vez de **"r"**.

*Vamos ver quais os vários modos de abertura / acesso dos ficheiros!*

Resumo dos modos de acesso	
Letra	Descrição
<b>r</b>	Abre o ficheiro para <b>leitura</b> (e coloca o cursor no <b>início</b> )
<b>r+</b>	Abre o ficheiro para <b>leitura</b> e <b>escrita</b> (e coloca o cursor no <b>início</b> )
<b>w</b>	Abre o ficheiro para <b>escrita</b> ( <b>para criar</b> informação num ficheiro)
<b>w+</b>	Abre o ficheiro para <b>leitura</b> e <b>escrita</b> ( <b>para criar</b> ou <b>substituir informação</b> num ficheiro)
<b>a</b>	Abre o ficheiro para <b>acrescentar</b> (e coloca o cursor no <b>final do ficheiro</b> )
<b>a+</b>	Abre o ficheiro para <b>leitura</b> e <b>acrescentar</b> (e coloca o cursor no <b>final do ficheiro</b> )

## FASE 2 – Ler e/ou escrever do/no ficheiro

### Leitura da Informação

Para **ler informação proveniente dos ficheiros**, podemos usar as seguintes funções:

- **fgetc()** (Leitura **carácter a carácter** do ficheiro ) → int **fgetc**(Ficheiro);

**ou**

- **fgets()** (Leitura **linha a linha de um ficheiro**. Esta função precisa de uma string) → char **\*fgets**(char \*x, int y, FILE \*fp)

**ou**

- **fscanf** (Escrita formatada) → int **fscanf**(FILE \*fp, const char \*format)

### Escrita da Informação

Para **escrever informação nos ficheiros**, podemos usar as seguintes funções:

**fputs()** → int **fputs**(int ch, FILE \*fp)

**ou**

**fprintf()** (Escrita formatada) → int **fprintf**(FILE \*fp, const char \*format)

## FASE 3 – Fechar o ficheiro

É importante referir que quando abrirmos um ficheiro, **é muito importante fechar o mesmo**, para que:

- Não esteja a **consumir recursos desnecessariamente**;
- O **ficheiro seja fechado corretamente e não comprometa a informação dentro do mesmo**;

Logo, no final das operações de ou mais ficheiros devemos *fechar o(s) arquivo(s) e usar uma das seguintes expressões*:

- **`fclose(fp);` /\* para fechar o ficheiro que foi aberto \*/**

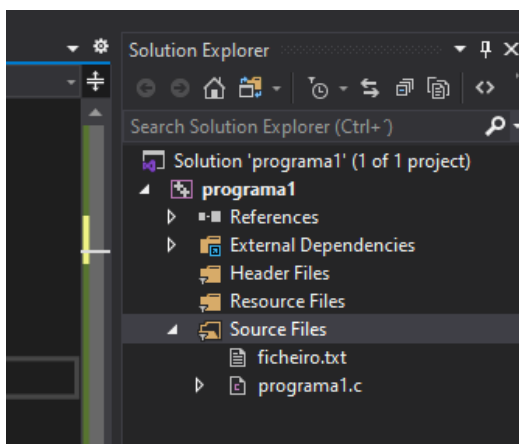
**Ou se tivermos muito ficheiros abertos:**

- **`fcloseall();` /\* para fechar todos os ficheiros de uma vez \*/**

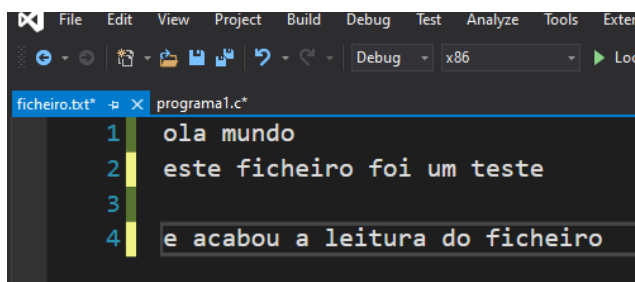
## Exercícios Práticos

### Parte 1 – Exemplos de Abrir e ler ficheiros

Para os próximos exemplos práticos, vamos assumir que criamos o ficheiro de texto (*ficheiro.txt*), na mesma diretoria que o ficheiro de programação:



E dentro do *ficheiro.txt*, vamos colocar a seguinte informação:



```

1  ola mundo
2  este ficheiro foi um teste
3
4  e acabou a leitura do ficheiro

```

**Exercício 1** – Exemplo com leitura do ficheiro via *fgets*:

Agora vamos *abrir* e *ler* (modo de acesso “r”) o *ficheiro.txt* dentro do programa

```

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

void main() {
    FILE* fp;
    char linha[1000]; /* Para armazenar as linhas lidas dos ficheiros */

    /* Criar ficheiro com o modo de acesso "r" */
    if ((fp = fopen("ficheiro.txt", "r")) == NULL) {
        printf("Impossivel abrir/criar o ficheiro pretendido!\n");
        exit(1);
    }
    else {
        printf("Informacao Ficheiro:\n\n");

        /* Precisamos de um ciclo pois a função lê linha a linha */
        while (fgets(linha, 1000, fp) != NULL) { /* Enquanto nao atingir o fim do ficheiro */
            printf("%s", linha); /* Importante meter %s pois estamos a trabalhar com strings */
        }

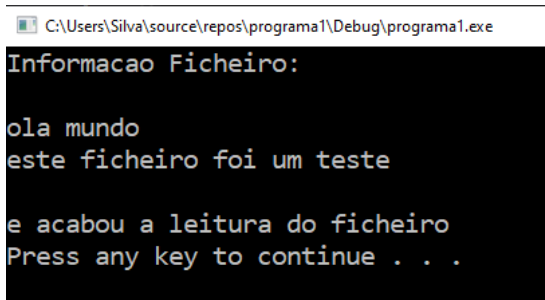
        printf("\n");
    }

    fclose(fp); /* Fechar o ficheiro que foi aberto */

    system("PAUSE");
}

```

**Resultado:**



```

C:\Users\Silva\source\repos\programa1\Debug\programa1.exe
Informacao Ficheiro:

ola mundo
este ficheiro foi um teste

e acabou a leitura do ficheiro
Press any key to continue . . .

```

### Exercício 2 – Exemplo com leitura do ficheiro via *fscanf* / *fprintf*:

Agora vamos **abrir** e **ler** (modo de acesso “r”) o **ficheiro.txt** dentro do programa

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

void main() {
    FILE* fp;

    /* Criar ficheiro com o modo de acesso "r" */
    if ((fp = fopen("ficheiro.txt", "r")) == NULL) {
        printf("Impossivel abrir/criar o ficheiro pretendido!\n");
        exit(1);
    }
    else {
        printf("Informacao Ficheiro:\n\n");

        /* Para armazenar os caracteres lidos.*/
        char ch;

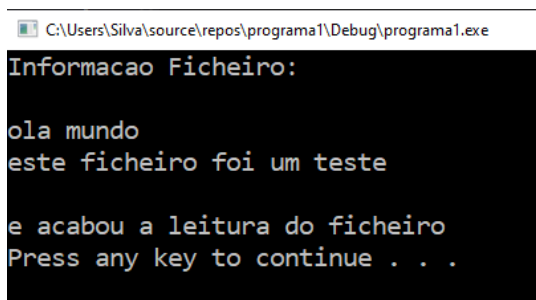
        /* Precisamos de um ciclo pois a função lê carácter a carácter */
        while (fscanf(fp, "%c", &ch) != EOF) { /*Enquanto nao atingir o fim do ficheiro */
            fprintf(stdout, "%c", ch); /* mostra os resultados */
        }

        printf("\n");
    }

    fclose(fp); /* Fechar o ficheiro que foi aberto */

    system("PAUSE");
}
```

### Resultado:



```
C:\Users\Silva\source\repos\programa1\Debug\programa1.exe
Informacao Ficheiro:
ola mundo
este ficheiro foi um teste
e acabou a leitura do ficheiro
Press any key to continue . . .
```

### Parte 2 – Exemplos de Abrir e Escrever em ficheiros

#### Exercício 3 – Exemplo com leitura do ficheiro via *fputs*:

Agora vamos **abrir/criar** e **escrever** (modo de acesso “w”) o **ficheiro.txt** dentro do programa

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

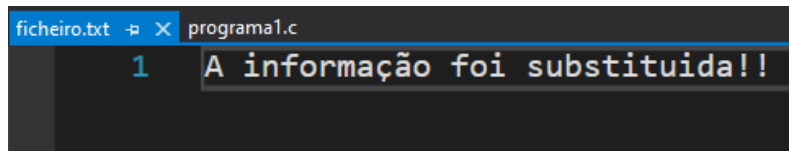
void main() {
    FILE* fp;
    char texto[35] = "A informação foi substituída!!";

    /* Criar ficheiro com o modo de acesso "w" (se o ficheiro existir este será apagado e criado de novo */
    if ((fp = fopen("ficheiro.txt", "w")) == NULL) {
        printf("Impossível abrir/criar o ficheiro pretendido!\n");
        exit(1);
    }
    else {
        /*Coloca o texto que declaramos no início*/
        fputs(texto, fp);
    }

    fclose(fp); /* Fechar o ficheiro que foi aberto */

    system("PAUSE");
}
```

#### Resultado:



ficheiro.txt ✕ programa1.c

1 A informação foi substituída!!

### Parte 3 - Acesso Sequencial e Acesso Direto nos Ficheiros

O **acesso a informação** pode ser realizado através de **2 formas**:

- **Acesso Sequencial**: como tem sido feito até agora, isto é, percorrendo o ficheiro até localizarmos o que pretendemos;
- **Acesso Direto**: Colocamo-nos na posição que queremos, sem ter de percorrer toda a informação até encontrar o ponto requerido;

A função **fseek()** é usada para posicionarmos no ficheiro. A função devolve 0 em caso de erro, e um valor diferente de 0 em caso de sucesso. O **último campo indica** de que ponto queremos iniciar o **posicionamento**. Neste ponto só podemos indicar **3 valores**, que estão definidos em **constantes**:

Constante	Valor	Posicionamento
SEEK_SET	0	A partir do início do ficheiro
SEEK_CUR	1	A partir da posição atual no ficheiro
SEEK_END	2	A partir do final do ficheiro

A função **ftell()** permite devolver a posição atual do ponteiro/cursor no ficheiro.

A função **rewind()** permite colocar o ponteiro/cursor no início do ficheiro, ou seja, volta para o início.

**Exercício 4** – Vamos aplicar os conceitos referidos anteriormente:

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

void main() {

    FILE *fp;
    char data[100]; /* Para armazenar o texto que vem do ficheiro */

    fp = fopen("teste.txt", "w+");
    fputs("Isto e uma experiencia utilizando o tema de acesso a informação de forma direta!!", fp);
    fseek(fp, 0, SEEK_SET);
    fgets(data, 60, fp);
    printf("Antes de usar a funcao fseek:\n %s\n\n", data);

    // Coloca o cursor no inicio do ficheiro e so le a partir do caracter numero 21
    fseek(fp, 21, SEEK_SET);
    fgets(data, 100, fp);
    printf("Depois de usar o metodo SEEK_SET a partir do caracter numero 21:\n %s\n\n", data);
}
```

```
// Retirar 10 caracteres (-10) em relação a posição atual do cursor (logo 21 - 10 = 11)
fseek(fp, -10, SEEK_CUR);
fgets(data, 100, fp);
printf("Depois de usar o metodo SEEK_CUR e retirar 10 caracteres em relação a posição atual do cursor:\n %s\n\n", data);

// Quais são os ultimos 7 caracteres no fim do ficheiro?
fseek(fp, -7, SEEK_END);
fgets(data, 100, fp);
printf("Depois de usar o metodo SEEK_END para saber so ultimos 7 caracteres no fim do ficheiro:\n %s\n\n", data);

/* Para voltar ao inicio do ficheiro */
rewind(fp); // ou podemos usar: fseek(fp, 0, SEEK_SET);

/*Para fechar o ficheiro*/
fclose(fp);

system("PAUSE");
exit(0);
}
```