

2º Semestre

Técnicas de Programação

Luiz Fernando Carvalho

luizfcarvalhoo@gmail.com

Arquivos

- Para ler e escrever tipos de dados maiores que um byte, o sistema de arquivos C ANSI fornece duas funções:

```
unsigned fread(void *buffer, int numero_de_bytes, int cont, FILE *arq)
```

```
unsigned fwrite(void *buffer, int numero_de_bytes, int cont, FILE *arq)
```

- Essas funções permitem a leitura/escrita de blocos de qualquer tipo de dado;
 - Para `fread()`, `buffer` é um ponteiro para uma região de memória (ou variável) que receberá os dados do arquivo;
 - Para `fwrite()`, `buffer` é um ponteiro para as informações que serão escritas no arquivo;
 - `numero_de_bytes`: a quantidade de bytes a ser lido/escrito;
 - `cont`: quantos itens de comprimento `numero_de_bytes` serão lidos/escritos;
 - `arq`: é o ponteiro para (uma stream) um arquivo;

Arquivos

- Para ler e escrever tipos de dados maiores que um byte, o sistema de arquivos C ANSI fornece duas funções:

```
unsigned fread(void *buffer, int numero_de_bytes, int cont, FILE *arq)
```

```
unsigned fwrite(void *buffer, int numero_de_bytes, int cont, FILE *arq)
```

- Perceba que:
 - `fread()` retorna o número de itens lidos;
 - Esse valor pode ser menor que `cont` se o final do arquivo for atingido ou ocorrer um erro.
 - `fwrite()` retorna o número de itens escritos
 - Caso esse valor seja diferente de `cont`, ocorreu algum erro.
 - O número total de bytes escritos é:

```
numero_de_bytes * cont
```

fread e fwrite

- Quando o arquivo for aberto para dados binários, fread e fwrite podem ler e escrever qualquer tipo de informação:
 - int;
 - float;
 - double;
 - array/vetor;
 - Struct;

Exemplo fread e fwrite

```
1 int main(){
2     FILE *arq;
3     char str[11] = "Computacao";
4     int i = 101, vet[5] = {1, 2, 3, 4, 5};
5     char str2[11];
6     int i2, vet2[5];
7
8     arq = fopen("teste.dat", "wb");
9     if(arq == NULL){
10         //testa a abertura do arquivo
11     }
12     fwrite(&i, sizeof(int), 1, arq);
13     fwrite(vet, sizeof(int), 5, arq);
14     fwrite(str, sizeof(char), 10, arq);
15
16     fclose(arq);
```

```
17     arq = fopen("teste.dat", "rb");
18     if(arq == NULL){
19         //testa a abertura do arquivo
20     }
21     fread(&i2, sizeof(int), 1, arq);
22     fread(vet2, sizeof(int), 5, arq);
23     fread(str2, sizeof(char), 10, arq);
24     str2[10] = '\0';
25
26     printf("%d\n", i2);
27     for(i=0; i<5; i++){
28         printf("%d ", vet2[i]);
29     }
30     puts(str2);
31     fclose(arq);
32     return 0;
33 } //Finaliza a função main
```

Exemplo fread e fwrite

- Uma das mais úteis aplicações de fread e fwrite envolve ler e escrever tipos de dados definidos pelo usuário, especialmente structs.

```
1 typedef struct{
2     char nome[30];
3     int idade;
4     float altura;
5 }Pessoa;
6
7 int main(){
8     FILE *arq;
9     Pessoa p;
10
11     arq = fopen("teste.dat", "w+b");
12     if(arq == NULL){
13         //testa a abertura do arquivo
14     }
15     fwrite(&p, sizeof(Pessoa), 1, arq);
16
17     fclose(arq);
```

```
Pessoa p2;
```

```
fread(&p2, sizeof(Pessoa), 1, arq);
printf("Nome: %s\n", p2.nome);
printf("Idade: %d\n", p2.idade);
printf("Altura: %f", p2.altura);
```

Movimentando pelo Arquivo

- Normalmente, o acesso a um arquivo ocorre de modo sequencial;
- Porém é possível fazer buscas e acessos aleatórios nos arquivos;

```
int fseek(FILE *arq, int numero_de_bytes, int origem)
```

- Esta função move a posição corrente de leitura ou escrita no arquivo em `numero_de_bytes`, a partir de um ponto especificado
- Sobre os parâmetros:
 - `numero_de_bytes`: é o total de bytes a partir de origem a ser “pulado”.
 - `origem`: determina a partir de onde os `numero_de_bytes` de movimentação serão contados.
 - Retorno: devolve o valor zero quando a movimentação é bem sucedida;

Movimentando pelo Arquivo

```
int fseek(FILE *arq, int numero_de_bytes, int origem)
```

- Os valores possíveis para origem são definidos por macros em `stdio.h`:

Nome	Valor	Significado
SEEK_SET	0	Início do arquivo
SEEK_CUR	1	Posição corrente do arquivo
SEEK_END	2	Fim do arquivo

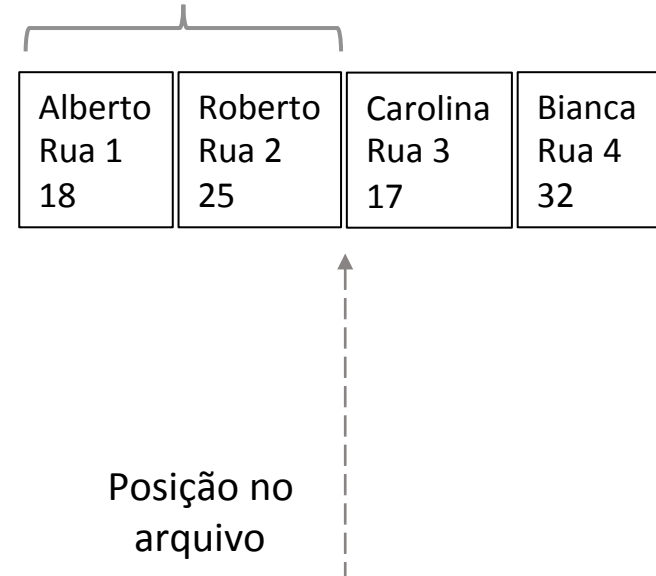
- `numero_de_bytes` pode ser negativo quando usado com `SEEK_CUR` e `SEEK_END`


```

1 typedef struct{
2     char nome[30], rua[30];
3     int idade;
4 }Pessoa;
5
6 int main(){
7     FILE *arq;
8     Pessoa p, grupo[4] = {"Alberto", "Rua 1", 18,
9                           "Roberto", "Rua 2", 25,
10                          "Carolina", "Rua 3", 17,
11                          "Bianca", "Rua 4", 32};
12
13     arq = fopen("cadastro.dat", "wb");
14
15     fwrite(grupo, sizeof(Pessoa), 4, arq);
16     fclose(arq);
17
18     arq = fopen("cadastro.dat", "rb");
19     fseek(arq, 2*sizeof(Pessoa), SEEK_SET);
20
21     fread(&p, sizeof(Pessoa), 1, arq);
22     fclose(arq);
23
24     puts(p.nome);
25     puts(p.rua);
26     printf("%d", p.idade);

```

$2 * \text{sizeof}(\text{Pessoa})$



Alberto Rua 1 18	Roberto Rua 2 25	Carolina Rua 3 17	Bianca Rua 4 32
------------------------	------------------------	-------------------------	-----------------------

Posição no
arquivo

```

1 typedef struct{
2     char nome[30], rua[30];
3     int idade;
4 }Pessoa;
5
6 int main(){
7     FILE *arq;
8     Pessoa p, grupo[4] = {"Alberto", "Rua 1", 18,
9                           "Roberto", "Rua 2", 25,
10                          "Carolina", "Rua 3", 17,
11                          "Bianca", "Rua 4", 32};
12
13     arq = fopen("cadastro.dat", "wb");
14
15     fwrite(grupo, sizeof(Pessoa), 4, arq);
16     fclose(arq);
17
18     arq = fopen("cadastro.dat", "rb");
19
20     fseek(arq, -1*sizeof(Pessoa), SEEK_END);
21
22     fread(&p, sizeof(Pessoa), 1, arq);
23     fclose(arq);
24
25     puts(p.nome);
26     puts(p.rua);
27     printf("%d", p.idade);

```

$-1 * \text{sizeof}(\text{Pessoa})$

Alberto Rua 1 18	Roberto Rua 2 25	Carolina Rua 3 17	Bianca Rua 4 32
------------------------	------------------------	-------------------------	-----------------------

Posição no
arquivo

Volta “uma struct”
em relação ao final do
arquivo

Movimentando pelo Arquivo

```
void rewind(FILE *arq)
```

- A função `rewind` é uma outra opção para movimentação no arquivo;
- Com ele é possível voltar ao início do arquivo;
- Portanto, não é mais necessário fechar e reabrir o arquivo para retornar ao início dele;
- Muito indicado para modos de abertura em que pode-se ler e escrever dados.

Apagando um Arquivo

```
int remove(nome_arquivo)
```

- A função remove é usado para esse propósito;
- Diferente das funções anteriores, essa aqui recebe o caminho (absoluto ou relativo) e nome do arquivo a ser excluído, e não o seu ponteiro;
- Um valor inteiro é retornado igual a zero se o arquivo for excluído com sucesso;

```
int status;  
status = remove("teste.txt");  
if(status != 0){  
    printf("Erro na remoção do arquivo");  
    system("pause");  
    exit(1);  
}else  
    printf("Arquivo removido com sucesso");
```

Renomeando Arquivo

```
int rename(nome_antigo, nome_novo)
```

- A função rename renomeia um determinado arquivo;
- O nome_antigo é substituído pelo nome_novo;

```
int status;  
status = rename("teste.txt", "arquivo.txt");  
if(status != 0){  
    perror("Erro");  
    system("pause");  
    exit(1);  
}else  
    printf("Arquivo removido com sucesso");
```

Impressão do erro

`void perror(string)`

- Fornece a mensagem do último erro encontrado durante a execução de uma chamada de sistema ou de uma função da biblioteca;
- A *string* (opcional) é uma mensagem que pode ser impressa antes da especificação do erro.

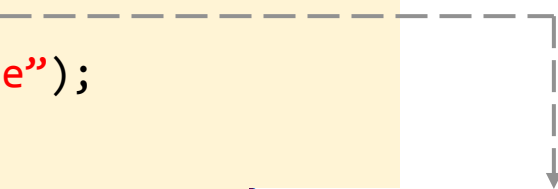
```
int status;  
status = rename("teste.txt", "arquivo.txt");  
if(status != 0){  
    perror("Erro");  
    system("pause");  
    exit(1);  
}else  
    printf("Arquivo removido com sucesso");
```

```
Erro: No such file or directory  
Pressione qualquer tecla para continuar. . . █
```

Impressão do erro

`void perror(string)`

```
int main(){  
    FILE *arq;  
    arq = fopen("arquivo.txt", "r");  
  
    fputs("Tentando escrever", arq);  
    if(ferror(arq)){  
        perror("");  
        system("pause");  
        exit(1);  
    }  
  
    fclose(arq);  
  
    return 0;  
}
```



```
Bad file descriptor  
Pressione qualquer tecla para continuar. . .
```

Função	O que faz
fopen	Abre o arquivo
fclose	Fecha o arquivo
fscanf	Leitura formatada
fprintf	Escrita formatada
fgets	Leitura de string
fgetc	Leitura de caractere
fputs	Impressão de string
fputc	Impressão de caractere
fread	Leitura de um bloco de dados do arquivo
fwrite	Escrita de um bloco de dados no arquivo
fseek	Reposiciona o ponteiro
rewind	Reposiciona o ponteiro para o início do arquivo
remove	Excluir um arquivo
rename	Renomeia um arquivo
perror	Imprime o erro