Arquivos

Professor: Guilherme Corredato Guerino

Disciplina: Algoritmos e Técnicas de Programação

Ciência da Computação
Universidade Estadual do Paraná

Introdução

- Um **arquivo** é **armazenado** em um d**ispositivo de memória secundária**, tal como o **disco**.

- O arquivo é formado por uma **coleção de caracteres** (**arquivo texto**) ou **de bytes** (**arquivo binário**).

- Os arquivos podem ser **lidos** e/ou **escritos** pelos **programas**.

Introdução

- Um arquivo em C pode representar diferentes elementos, tais como:
  - **Arquivos em disco**
  - **Impressora**
  - **Teclado**
  - **Qualquer dispositivo de entrada ou saída**

Observação

- Na disciplina, **somente arquivos em disco** serão considerados, entretanto, se for necessário trabalhar com **outros dispositivos**, utiliza-se a **mesma interface**.

Introdução

- A linguagem C oferece suporte à utilização de arquivos por meio da **biblioteca <stdio.h>**.

- Essa biblioteca oferece funções para a **manipulação de arquivos** e apresenta **novos tipos de dados** a serem usados com arquivos, como o tipo **FILE**.

Tipo FILE

- Ao declarar uma **variável do tipo FILE**, é possível **identificar um arquivo no disco**, direcionando para ele **todas as operações**.

- **Declaração:**

```
FILE *arquivo, *ponteiro;
```

Arquivos

- Diferentes **operações** podem ser realizadas em arquivos.

- **Entre as principais operações estão:**
  - Abertura
  - Fechamento
  - Leitura
  - Escrita

Arquivos

- Diferentes **operações** podem ser realizadas em arquivos.

- **Entre as principais operações estão:**
  - **Abertura**
  - Fechamento
  - Escrita
  - Leitura

Abertura de um Arquivo

- A função básica para abrir um arquivo é **fopen().**

```
FILE* fopen (char* nome, char* modo);
```

Onde:
nome: nome do arquivo.
modo: modo de abertura do arquivo.

Abertura de um Arquivo

Modo de Abertura	Descrição
r	Abre um arquivo de texto para leitura
w	Cria um arquivo de texto para escrita
a	Anexa dados a um arquivo de texto
rb	Abre um arquivo binário para leitura
wb	Cria um arquivo binário para escrita
ab	Anexa dados a um arquivo binário
r+	Abre um arquivo de texto para leitura e escrita
w+	Cria um arquivo de texto para leitura e escrita
a+	Anexa novos dados ou cria um arquivo de texto para leitura e escrita
rb+	Abre um arquivo binário para leitura e escrita
wb+	Cria um arquivo binário para leitura e escrita
ab+	Anexa novos dados a um arquivo binário para leitura e escrita

Abertura de um Arquivo

```
FILE* ponteiro_arquivo;

ponteiro_arquivo = fopen("Teste.txt", "w");

if (ponteiro_arquivo == NULL){
  printf("\n Erro na abertura do arquivo \n");
}
.........
```

Abertura de um Arquivo

- No exemplo apresentado, a função fopen() **cria um novo arquivo** chamado **Teste.txt**, para **escrita**.

- Quando a função fopen() é utilizada em modo de abertura **w** (ou **wb**), as seguintes situações podem acontecer:
  - Se o arquivo **não existir**, ele será **criado**; e
  - Se o arquivo **já existir**, ele será **sobreposto por um novo arquivo vazio**.

Arquivos

- Diferentes **operações** podem ser realizadas em arquivos.

- **Entre as principais operações estão:**
  - Abertura
  - **Fechamento**
  - Escrita
  - Leitura

Fechamento de um Arquivo

- A função básica para fechar um arquivo é **fclose()**.

```
int fclose (FILE* ponteiro_arquivo);
```

Onde:
ponteiro_arquivo: referência para o arquivo.

Observações

- Quando a função fclose() é executada, um **número inteiro é retornado como resultado**.

- Se o **número inteiro é zero**, significa que o arquivo foi **fechado corretamente**. Caso contrário, significa que **um erro aconteceu**.

Observações

- Qualquer **problema** que ocorra com a **execução de um programa** pode **corromper** e/ou **danificar** os **arquivos abertos**.

- É aconselhável que os arquivos sejam **mantidos em aberto** o **menor tempo possível**.

Fechamento de um Arquivo

```
FILE* ponteiro_arquivo;

ponteiro_arquivo = fopen("Teste.txt", "a+");

if (ponteiro_arquivo == NULL){
  printf("\n Erro na abertura do arquivo \n");
  exit(1);
}
.........

fclose(ponteiro_arquivo);
```

Arquivos

- Diferentes **operações** podem ser realizadas em arquivos.

- **Entre as principais operações estão:**
  - Abertura
  - Fechamento
  - **Escrita**
  - Leitura

Escrita de Dados em um Arquivo

- A escrita de arquivos em **modo texto** pode ser realizada pelas seguintes funções:
  - **int** fprintf (**um ou mais parâmetros**)
  - **int** fputc (**um ou mais parâmetros**)
  - **char*** fputs (**um ou mais parâmetros**)

Escrita de Dados em um Arquivo

- A função fprintf() apresenta a seguinte sintaxe:

```
int fprintf(FILE* p, char* formato, lista);
```

Onde:
p: referência para o arquivo.
formato: formato do dado que será salvo.
lista: lista de variáveis cujo conteúdo será salvo no arquivo.

Observação

- O **valor de retorno** da função **fprintf()** é:
  - O **número de bytes escritos** no arquivo.

```c
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
  FILE *pont_arq; // cria variável ponteiro para o arquivo
  char palavra[20]; // variável do tipo string

  pont_arq = fopen("arquivo_palavra.txt", "w");

  if(pont_arq == NULL)  {
      printf("Erro na abertura do arquivo!");
      return 1;
  }

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

  fprintf(pont_arq, "%s", palavra);

  fclose(pont_arq);

  printf("Dados gravados com sucesso!");

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

Escrita de Dados em um Arquivo

- A função fputc() apresenta a seguinte sintaxe:

```
int fputc(int c, FILE* p);
```

Onde:
c: código do caractere a ser escrito no arquivo.
p: referência para o arquivo.

Observação

- O **valor de retorno** da função **fputc()** é o próprio **caractere escrito** ou **EOF**, se ocorrer **erro na escrita**.

```c
#include <stdio.h>
#include <stdlib.h>
int main(void){
  FILE *pont_arq;
  char frase[50];
  int i;
  int tamanho;

  pont_arq = fopen("arquivo1.txt","w");
  if (pont_arq == NULL)
  {
    printf("Erro ao tentar abrir o arquivo!");
    exit(1);
  }

  printf("Digite a frase a ser gravada no arquivo:");
  gets(frase);

  tamanho=strlen(frase);

  for(i=0; i < tamanho; i++)
  {
    fputc(frase[i], pont_arq);
  }

  fclose(pont_arq);
  return 0;
}
```

Escrita de Dados em um Arquivo

- A função fputs() apresenta a seguinte sintaxe:

```
char* fputs (char* s, FILE* p);
```

Onde:
s: cadeia de caracteres que será escrita no arquivo.
p: referência para o arquivo.

Observação

- O **valor de retorno** da função **fputs()** é:
  - Um **ponteiro** da cadeia de caracteres ou **NULL** em caso de **erro de escrita** e/ou **final de arquivo**.

```c
#include <stdio.h>
#include <stdlib.h>
int main(void){
   FILE *pont_arq;
   int r;
   pont_arq = fopen("arquivo2.txt", "w");

  if (pont_arq == NULL){
    printf("Erro ao tentar abrir o arquivo!");
    exit(1);
  }

   r = fputs("Programando em Linguagem C.", pont_arq);

   if( r == EOF){
       printf("Erro ao tentar gravar os dados! \n");
   }
   else{
        printf("Dados gravados com sucesso. \n");
     }

   fclose(pont_arq);
   return 0;
}
```

Arquivos

- Diferentes **operações** podem ser realizadas em arquivos.

- **Entre as principais operações estão:**
  - Abertura
  - Fechamento
  - Escrita
  - **Leitura**

Leitura de Dados de um Arquivo

- A leitura de arquivos em **modo texto** pode ser realizada pelas seguintes funções:
  - **int** fgetc (**um ou mais parâmetros**)
  - **char*** fgets (**um ou mais parâmetros**)

Leitura de Dados de um Arquivo

- A função fgetc() apresenta a seguinte sintaxe:

```
int fgetc (FILE* p);
```

Onde:
p: referência para o arquivo.

Observações

- O **valor de retorno** de **fgetc()** é o **código numérico do caractere lido** em caso de sucesso.

- Se o **fim do arquivo** for alcançado, a constante **EOF** (***End of File***) é retornada.

```c
#include <stdio.h>
#include <stdlib.h>
int main(void){
  FILE *pont_arq;
  char c;

  pont_arq = fopen("arquivo1.txt","r");
  if (pont_arq == NULL)  {
    printf("Erro ao tentar abrir o arquivo!");
    exit(1);
  }
  printf("Lendo e exibindo os dados do arquivo \n\n");
  //Faça
  do
  {
      //faz a leitura do caracter no arquivo apontado por pont_arq
      c = fgetc(pont_arq);

      //exibe o caracter lido na tela
      printf("%c" , c);
  }while (c != EOF);//enquanto não for final de arquivo

       fclose(pont_arq);//fechando o arquivo
  system("pause");//pausa na tela, somente para Windows
  return(0);
}
```

Leitura de Dados de um Arquivo

- A função fgets() apresenta a seguinte sintaxe:

```
char* fgets (char* s, int n, FILE* p);
```

Onde:
s: cadeia de caracteres que armazenará o conteúdo lido.
n: número máximo de caracteres que deve ser lido.
p: referência para o arquivo.

Observação

- O **valor de retorno** da função **fgets()** é:
  - O ponteiro da **cadeia de caracteres** passada como parâmetro.
  - NULL em **caso de erro de leitura** e/ou **final de arquivo**.

```
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
  FILE *pont_arq;
  char texto_str[20];

  //abrindo o arquivo_frase em modo "somente leitura"
  pont_arq = fopen("arquivo_palavra.txt", "r");

  //enquanto não for fim de arquivo o looping será executado
  //e será impresso o texto
  while(fgets(texto_str, 20, pont_arq) != NULL)
  printf("%s", texto_str);

  //fechando o arquivo
  fclose(pont_arq);

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

Leitura de Dados de um Arquivo

- A função fscanf() apresenta a seguinte sintaxe:

```
int fscanf (FILE* p, char* formato, lista);
```

Onde:
p: referência para o arquivo.
formato: formato do dado a ser lido.
lista: lista de variáveis que receberão os dados lidos.

Leitura de Dados de um Arquivo

```
.....
  while (fscanf(fp, "%c", &c) == 1){
    if (c == '\n'){
      nlinhas++;
      printf("\n");
    }
    printf("%c",c);
  }
.....
```

```c
#include <stdio.h>

int main(void)
{
 char url[]="arquivo.txt",
      ch1, ch2, ch3;
 FILE *arq;

 arq = fopen(url, "r");
 if(arq == NULL)
   printf("Erro, nao foi possivel abrir o arquivo\n");
 else
  while( (fscanf(arq,"%c %c %c\n", &ch1, &ch2, &ch3))!=EOF )
   printf("%c %c %c\n", ch1, ch2, ch3);

 fclose(arq);

 return 0;
}
```