IFSULDEMINAS, *campus* Muzambinho
Curso de Ciência da Computação

Entrada e Saída com Arquivos Utilizando a Linguagem C

Prof. Ricardo José Martins
ricardo.martins@muz.ifsuldeminas.edu.br

Curso de Bacharelado em Ciência da Computação
AED III – Algoritmo e Estruturas de Dados III

Entrada e Saída de Dados pelo Console

- As operações de Entrada e Saída (**E/S**) em **ANSI C** são efetuadas pelas funções de biblioteca da linguagem, existindo **E/S** pelo console e a E/S de arquivo.

- **Leitura e Escrita de Caracteres**
  - **int getchar();**

    Lê caracter do teclado; o byte de baixa ordem contém o caractere.

  - **int putchar(int c);**

    escreve seu argumento na tela a partir da posição atual do cursor;
    devolve o caractere escrito ou -1 (constante **EOF** definida em **stdio.h**)

Entrada e Saída de Dados pelo Console

- **Leitura e escrita de caracteres**
  - **Programa que lê caracteres da tela e inverte a caixa deles**

```
main()
{
  char ch;
  printf("Entre com algum texto (digite um . para sair)");
  do
  {
    ch = getchar();
    if (islower(ch))
       ch = toupper(ch);
    else
        ch = tolower(ch);
    putchar(ch);
  } while (ch != '.');
}
```

Entrada e Saída de Dados pelo Console

- **Leitura e Escrita de Strings**
  - **`char * gets(char *str);`**

    `str é uma matriz de caracteres que recebe os caracteres enviados pelo usuário.`

    `Ex: char str[];`

    `gets(str);`
  - **`int puts(char *s);`**

    `escreve seu argumento string na tela seguido por uma nova linha.`

    `Ex: puts(“alo”);`

Entrada e Saída de Dados pelo Console

- Leitura e escrita formatada
  - As funções printf() e scanf() podem ler e escrever dados em vários formatos que estão sob seu controle.

  - **`int printf(char *string_de_controle, lista_de_args);`**

    `Ex: printf(“Eu gosto %s de %c”,”muito”, ‘c’);`

  - **`int scanf(char *string_de_controle, lista_de_args);`**

    ```
    Ex: int i;
        scanf(“%d”,&i);
    ```

Entrada e Saída com Arquivos

- Introdução
  - Em C não existem comandos de Entrada e Saída, sendo estas tarefas executadas por funções especialmente criadas para esta finalidade e armazenadas em bibliotecas específicas.

- Fluxos de Dados
  - Para isolar os programadores dos problemas de manipular os vários tipos de dispositivos de armazenamento e seus diferentes formatos a linguagem C utiliza o conceito de fluxo (*stream*) de dados. Todos os diferentes sistemas de arquivos se comportam da mesma maneira quando manipulados como um fluxo contínuo de dados.
  - Dados podem ser manipulados em dois diferentes tipos de fluxos:
    - **fluxos de texto e fluxos binários.**

Entrada e Saída com Arquivos

- **Fluxos de Texto**
  - Um fluxo de texto é composto por uma seqüência de caracteres, que pode ou não ser dividida em linhas, terminadas por um caracter de final de linha.

- **Fluxo Binário**
  - Um fluxo binário é composto por uma seqüência de bytes lidos, sem tradução, diretamente do dispositivo externo. Não ocorre nenhuma tradução e existe uma correspondência um para um entre os dados do dispositivo e os que estão no fluxo.

Entrada e Saída com Arquivos

- *Streams* e Arquivos
  - o sistema de E/S de C fornece uma interface ao programador C, independente do dispositivo real que é acessado. Essa abstração é chamada de *stream*.
- *Streams*
  - trabalha com uma variedade de dispositivos : terminais, discos, fitas, dentre outros
- *Streams* de Texto
  - É uma seqüência de caracteres. Permite que seja organizado por linhas. O caracter de nova linha é opcional na última linha.
- *Streams* Binárias
  - É uma seqüência de bytes com uma correspondência de um para um com aqueles encontrado no dispositivo.
- Arquivo
  - Pode ser qualquer dispositivo: disco, terminal ou impressora.
  - Deve-se associar um arquivo com uma *stream*.

Entrada e Saída com Arquivos

Funções de manipulação de arquivos

- **fopen()** : abre um arquivo
- **fclose()** : fecha um arquivo
- **putc()** : escreve um caracter num arquivo
- **fputc()** : o mesmo que **putc**
- **getc()** : lê um caracter do arquivo
- **fgetc()** : o mesmo que **getc**
- **fprintf()** : o mesmo que **printf** para o arquivo
- **fscanf()** : o mesmo que **scanf** para o arquivo
- **feof()** : devolve **true** se fim de arquivo
- **ferror()** : devolve **true** se ocorreu erro
- **rewind()** : volta para o inicio do arquivo
- **remove()** : apaga um arquivo
- **fflush()** : descarrega buffer de um arquivo.

Entrada e Saída com Arquivos

- Ponteiro de arquivo
  - O ponteiro é o meio comum que une o sistema de E/S com o buffer.
  - Ponteiro para informações :
    - nome
    - status
    - posição atual do arquivo.
    - **Exemplo em C: FILE *fp;**

- Abrir um arquivo

```
FILE *fopen(const *char nomearq, const *char modo);
```

Modo	Significado
r	Abre arquivo texto para leitura
w	Cria arquivo texto para escrita
rb	Abre arquivo binário para leitura
wb	Abre arquivo binário para escrita
r+	Abre arquivo texto para leitura/escrita
w+	Cria arquivo texto para leitura/escrita

Entrada e Saída com Arquivos

- Fechar um arquivo
  - `int fclose(FILE *fp);`

    `retorna zero se bem-sucedida.`
- Escrever um caracter no arquivo
  - `int putc(int ch, FILE *fp);`

    `retorna caracter se bem sucedido; senão retorna EOF.`
- Ler um caracter um caracter do arquivo
  - `int getc(FILE *fp);`
  - `devolve caracter lido.`
- Escrever uma string no arquivo
  - `int fputs(const char *str, int length, FILE *fp);`
- Ler uma string do arquivo
  - `char *fgets(char *s, int length, FILE *fp);`

Entrada e Saída com Arquivos

Exemplo 1

Programa que lê strings do teclado e as escreve em um arquivo em disco

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

main()
{
  FILE *fp;
  char str[80];

  if((fp=fopen("teste.txt", "w"))==NULL)
  {
    printf("arquivo não pode ser aberto\n");
    exit(1);
  }

  do
  {
    printf("Entre uma string (CR para sair) : \n");
    gets(str);
    strcat(str, "\n");
    fputs(str, fp);
  } while(*str != '\n' );
}
```

Entrada e Saída com Arquivos

Exemplo 2

Programa que simula o comando TYPE do DOS

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

main(int argc, char *argv[])
{
  FILE *fp;
  char ch;
  if (argc!=2)
  {
    printf("entrar com o nome do arquivo");
    exit(1);
  }

  if((fp=fopen(argv[1],"r"))==NULL)
  {
    printf("arquivo não pode ser aberto\n");
    exit(1);
  }
  ch = getc(fp);
  while(ch != EOF)
  {
     putchar(ch);
     ch = getc(fp);
  }
  fclose(fp);
}
```

Entrada e Saída com Arquivos

EXERCÍCIOS DE FIXAÇÃO

- **EXERCÍCIO 1**
  - Construa um programa para ler caracteres do teclado e imprimir em um arquivo em disco. O nome do arquivo deve ser passado por linha de comando.

- **EXERCÍCIO 2**
  - Construa um programa para copiar o conteúdo de um arquivo texto para outro. Os nomes dos arquivos fonte e destino devem ser passados por linha de comando (**obs:** usar **`feof()`** para verificar fim de arquivo.

Entrada e Saída com Arquivos

- **fread() e fwrite()**
  - Usados para ler e escrever dados maiores que um byte.

  Protótipos :

```
size_t fread(void *buffer, size_t num_bytes, size_t count, FILE *fp);
size_t fwrite(const void *buffer, size_t num_bytes, size_t  count, FILE *fp);
```

Lembre-se que size_t é definido em STDIO.H.

Exemplo

Escreve alguns dados não-caracteres em um arquivo em disco e lê de volta.

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

main()
{
  FILE *fp;
  double d = 12.23;
```

Entrada e Saída com Arquivos

```
  int i = 101;
  long l = 123023L;

  if ((fp=fopen(“dados.dat”, “wb+”)) == NULL)
  {
   printf(“arquivo não pode ser aberto\n”);
   exit(1);
  }

  fwrite(&d, sizeof(double), 1, fp);
  fwrite(&i, sizeof(int), 1, fp);
  fwrite(&l, sizeof(long), 1, fp);

  rewind(fp);

  fread(&d, sizeof(double), 1, fp);
  fread(&i, sizeof(int), 1, fp);
  fread(&l, sizeof(long), 1, fp);
  printf(“%f %d %ld, d, i, l);
  fclose(fp);
}
```

Entrada e Saída com Arquivos

EXERCÍCIOS DE FIXAÇÃO

- **EXERCÍCIO 3**
  - Construa um programa para cadastrar alunos em um arquivo binário. O programa deve permitir gerar uma lista de todos os alunos já cadastrados.
    - Informação dos alunos
      - matricula : inteiro
      - nome : string[30]
      - endereco : string[50]
      - curso : string[15]