Linguagem de Programação C

Arquivos

Cristiano Lehrer

http://www.ybadoo.com.br/

Introdução

- Em C um arquivo é apenas um conjunto de *bytes* colocados uns após os outros de forma sequencial:
- Utilização de arquivos:
  - Fonte de dados para o programa:
    - Trata-se de um arquivo de entrada de dados (input).
  - Destino de dados do programa:
    - Trata-se de um arquivo de saída de dados (output).

Tipos de Periféricos

- Periféricos de entrada:
  - Teclado, mouse, leitor de códigos de barras, …
- Periféricos de saída:
  - Tela do computador, impressora, …
- Periféricos de entrada e saída:
  - Memórias, discos rígidos, …
- Todas as entradas e saídas de dados são considerados pelo C como *streams*.

Streams

- Conjunto sequencial de caracteres, isto é, um conjunto de *bytes*, sem qualquer estrutura interna:
  - Programas e comandos veem apenas um conjunto de *bytes*;
  - São independentes do dispositivo *(device independent*):
    - Mesma sintaxe para enviar um fluxo de *bytes* para o monitor ou para a impressora, por exemplo.
  - Em C cada *stream* está ligado a um arquivo, o qual pode não corresponder fisicamente a um arquivo existente num disco, como é o caso do teclado ou da tela do computador.

Tipo Arquivo

- A declaração de variáveis para o processamento de arquivos faz-se usando o tipo `FILE` definido no arquivo `<stdio.h>`:

```
FILE *fp; /* fp - file pointer */
```

Abertura de um Arquivo

```
FILE *fopen(const char *filename, const char *mode)
```

- `filename`
  - *String* contendo o nome físico do arquivo;
  - No caso do arquivo `C:\TMP\DADOS.DAT`
    - `C:\\TMP\\DADOS.DAT`
- `mode`
  - *String* contendo o modo de abertura do arquivo

Modos de Abertura

- `"r"` – *read*: abertura do arquivo para leitura
  - Caso não possa abrir, a função devolve `NULL`.
- `"w"` – *write*: abertura do arquivo para escrita
  - É criado um novo arquivo com o nome passado à função. Notar que se já existir algum arquivo com o mesmo nome este é apagado e criado um novo arquivo, perdendo-se assim toda a informação nele contido. Caso não possa criá-lo, a função devolve `NULL`.
- `"a"` – *append*: abertura do arquivo para acrescentar
  - Se o arquivo não existir, ele é criado e funciona tal e qual o modo `"w"`. Se o arquivo já existir, coloca-se no final deste, de forma a permitir a escrita dos dados de forma sequencial a partir dos dados já existentes. Caso não possa abrir ou criá-lo, a função devolve `NULL`.

Fechamento de um Arquivo

```
int fclose(FILE *arq)
```

- A função devolve `0` em caso de sucesso ou a constante `EOF` em caso de erro:
  - Embora os arquivos sejam automaticamente fechados quando uma aplicação termina, é um bom hábito dos programadores realizarem eles mesmos o fechamento dos arquivos, evitando assim graves problemas que podem surgir motivados por uma falha de energia ou desligamento súbito do computador.
  - Nesse caso, os dados presentes em *buffers* não irão ser colocados no arquivo, e este último pode não ficar estável, podendo, em casos extremos, ocorrer mesmo a perda de setores do disco.

Exemplo – abrir e fechar um arquivo

```c
#include <stdio.h>
int main(){
  FILE *fp;
  char s[100];

  printf("Forneça o nome do arquivo: ");
  scanf("%s", s);

  fp = fopen(s, "r");

  if(fp == NULL){
    printf("Impossível abrir o arquivo %s\n", s);
  }
  else{
    printf("Arquivo %s aberto com sucesso!\n", s);
    fclose(fp);
  }
  return 0;
}
```

Exemplo – leitura de dados do arquivo

```
#include <stdio.h>
int main(){
  FILE *fp;
  char s[100];
  char ch;
  printf("Forneça o nome do arquivo: ");
  scanf("%s", s);
  fp = fopen(s, "r");
  if(fp == NULL){
    printf("Impossível abrir o arquivo %s\n", s);
  }
  else{
    while((ch = fgetc(fp)) != EOF){
      putchar(ch);
    }
    fclose(fp);
  }
  return 0;
}
```