Aula 24: Arquivos de texto

Introdução a Programação

Túlio Toffolo & Puca Huachi
http://www.toffolo.com.br

BCC201 – 2019/2
Departamento de Computação – UFOP

Aula de hoje

Aula de hoje

Arquivos

- Podem armazenar grande quantidade de **informação**.
- Mantém dados de forma **persistente** (gravado em disco).
- Acesso aos dados pode ser **não sequencial**.
- Acesso à informação pode ser **concorrente**.

Arquivos

Arquivo texto

- Armazena caracteres seguindo uma codificação (utf-8, por exemplo).
- Exemplo:

```
1 Este é um arquivo de texto, composto por caracteres...
2  - abc
3  - def...
```

Arquivo binário

- Sequência de bits sujeita às convenções do programa que o gerou.
- Exemplos: arquivos executáveis, compactados, de registros, etc.

Aula de hoje

Biblioteca <stdio.h>

C fornece o tipo FILE para representar um arquivo.
Na prática, usamos um ponteiro do tipo FILE.

Exemplo de declaração:

```
1  // arquivo para leitura
2  FILE *entrada;
3
4  // arquivo para gravação
5  FILE *saida;
```

Biblioteca <stdio.h>

A função `fopen` é usada para abrir um arquivo e tem o seguinte protótipo:

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

Note que a função tem 2 parâmetros:

1. `filename`: nome do arquivo a ser aberto
2. `mode`: modo de abertura do arquivo
   - `"r"` (read): **leitura**
   - `"w"` (write): **gravação** (sobrescreve o arquivo, se existir)
   - `"r+"` (read/update): **leitura** e **gravação** (arquivo tem que existir)
   - `"w+"` (write/read): **leitura** e **gravação**
   - `"a+"` (append/update): **acrescenta** dados no arquivo

Biblioteca <stdio.h>

Após abrir um arquivo, **temos que fechá-lo** com a função `fclose`.

```
1  int fclose(FILE *stream);
```

A função retorna 0 em caso de sucesso e `EOF` (-1) caso contrário.

Biblioteca <stdio.h>

Exemplos de uso de fopen e fclose:

```c
// abrindo arquivo file.txt para leitura
FILE *arquivo = fopen("file.txt", "r");
...
fclose(arquivo);
```

```c
// abrindo arquivo file.txt para gravação
FILE *arquivo = fopen("file.txt", "w");
...
fclose(arquivo);
```

```c
FILE *arquivo;
...
// abrindo arquivo file.txt no modo "append"
arquivo = fopen("file.txt", "a+");
...
fclose(arquivo);
```

Biblioteca <stdio.h>

Para impressão (gravar no arquivo), podemos utilizar a função `fprintf`, cujo funcionamento é muito parecido com a função `printf`.

```
1  int fprintf(FILE *stream, const char *format, ... );
```

Exemplo:

```
1  FILE *arquivo = fopen("texto.txt", "w");
2
3  // escrevendo texto e um número inteiro no arquivo
4  int n = 10;
5  fprintf(arquivo, "O valor de n = %d\n", n);
6
7  fclose(arquivo);
```

Biblioteca <stdio.h>

Exemplo completo de uso de fprintf:

```c
#include <stdio.h>

int main()
{
    int inteiro = 10;
    char palavra[10] = "Palavra";

    // declaração e carregamento do arquivo
    FILE *arquivo = fopen("file.txt", "w");

    // gravando um inteiro e uma palavra no arquivo
    fprintf(arquivo, "%s - %d\n", palavra, inteiro);

    // fechando (e salvando) o arquivo
    fclose(arquivo);
}
```

Biblioteca <stdio.h>

Para leitura, podemos utilizar a função `fscanf`, cujo funcionamento é muito parecido com a função `scanf`.

```
1  int fscanf(FILE *stream, const char *format, ... );
```

- A função retona o número de argumentos preenchidos ou `EOF` se o fim do arquivo for atingido.

Exemplo de uso:

```
1  FILE *arquivo = fopen("file.txt", "r");
2
3  // lendo um inteiro e um caractere separados por um espaço
4  int inteiro;
5  char caractere;
6  fscanf(arquivo, "%d %c", &inteiro, &caractere);
7
8  fclose(arquivo);
```

Biblioteca <stdio.h>

Exemplo completo de uso de `fscanf` para ler um vetor:

```c
#include <stdio.h>

int main()
{
    // declaração e carregamento do arquivo
    FILE *arquivo = fopen("file.txt", "r");

    // lendo o tamanho do vetor
    int n; // no máximo 100
    fscanf(arquivo, "%d", &n);

    // criando e lendo o vetor
    int vetor[100];
    for (int i = 0; i < n; i++)
        fscanf(arquivo, "%d", &vetor[i]);

    ...

    // fechando (e salvando) o arquivo
    fclose(arquivo);
}
```

Outra funções

A biblioteca <stdio.h> fornece outras funções úteis para **ler** dados de um arquivo texto:

```
// lê uma linha, incluindo o '\n' de um arquivo (lembram dela?)
char *fgets (char *str, int num, FILE *stream);

// lê um caractere e retorna (sim, retorna como um inteiro)
int fgetc(FILE *stream);

// retorna 0 se a posição atual não for o fim do arquivo
// e um valor diferente de 0 caso contrário
int feof(FILE *stream);
```

Outra funções

A biblioteca <stdio.h> também fornece outras funções úteis para **gravar** dados em um arquivo texto:

```c
// escreve uma string no arquivo
// a função retorna EOF em caso de erro
int fputs(const char *str, FILE *stream);

// escreve um caractere no arquivo (sim, como um inteiro);
// a função retorna EOF em caso de erro
int fputc(int character, FILE *stream);

// retorna 0 se a posição atual não for o fim do arquivo
// e um valor diferente de 0 caso contrário
int feof(FILE *stream);

// atualiza o arquivo (grava todo o conteúdo que ainda não foi
// gravado); retorna 0 em caso de sucesso e EOF caso contrário
int fflush(FILE *stream);
```

Aula de hoje

Exemplo 1

Crie um programa que conta o número de espaços em branco em um arquivo passado como argumento.

```c
int main(int argc, char **argv)
{
    int nEspacos = 0;
    char c;
    FILE *arquivo;

    arquivo = fopen(argv[1], "r"); // argv[1] é o primeiro argumento
    while (!feof(arquivo)) {
        c = fgetc(arquivo);
        if (c == ' ') nEspacos++;
    }
    fclose(arquivo);

    printf("O arquivo possui %d espaços.\n", nEspacos);
    return 0;
}
```

Exemplo 2

Crie um programa que copia um arquivo texto em outro arquivo texto removendo espaços (nomes dos arquivos são passados por argumento).

```
1  int main(int argc, char **argv)
2  {
3      char c;
4      FILE *entrada, *saida;
5
6      entrada = fopen(argv[1], "r"); // primeiro argumento
7      saida = fopen(argv[2], "w");   // segundo argumento
8      while (!feof(entrada)) {
9          c = fgetc(entrada);
10         if (c != ' ' && c != EOF)
11             fputc(c, saida);
12     }
13     fclose(entrada);
14     fclose(saida);
15
16     return 0;
17 }
```

Exemplo 2 (alternativa)

Crie um programa que copia um arquivo texto em outro arquivo texto removendo espaços (nomes dos arquivos são passados por argumento).

```
1  int main(int argc, char **argv)
2  {
3      char c;
4      FILE *entrada, *saida;
5
6      entrada = fopen(argv[1], "r"); // primeiro argumento
7      saida = fopen(argv[2], "w");   // segundo argumento
8      while (fscanf(entrada, "%c", &c) != EOF) {
9          if (c != ' ')
10             fprintf(saida, "%c", c);
11     }
12     fclose(entrada);
13     fclose(saida);
14
15     return 0;
16 }
```

Exemplo 3

Crie uma função que lê uma matriz $n \times m$ de inteiros de um arquivo texto e a imprime na saída. Assuma que $n \leq 100$ e $m \leq 100$.
O arquivo tem a seguinte informação:

- Os dois primeiro números indicam as dimensões da matriz (n e m).
- Em seguida, a matriz é incluída no arquivo.
- Exemplo:

```
1  5 4
2  10 9 4 3
3  2 8 3 0
4  2 3 1 9
5  28 3 6 4
6  9 1 4 5
```

Exemplo 3

```
1  /* Esta função lê os dados de uma matriz; note que as dimensões da
2   * matriz são armazenadas nas variáveis n e m passadas por referência
3   */
4  void leMatriz(int matriz[100][100], char arquivo[], int &n, int &m)
5  {
6      FILE *entrada = fopen(arquivo, "r");
7
8      fscanf(entrada, "%d", n);
9      fscanf(entrada, "%d", m);
10
11     for (int i = 0; i < *n; i++) {
12         for (int j = 0; j < *m; j++)
13             fscanf(entrada, "%d", &matriz[i][j]);
14     }
15
16     fclose(entrada);
17 }
```

Exemplo 4

Crie uma função que escreve uma matriz de inteiros em um arquivo.

- Os dois primeiro números no arquivo indicam as dimensões da matriz.
- Em seguida, a matriz é incluída no arquivo.
- Exemplo:

```
5 4
10 9 4 3
2 8 3 0
2 3 1 9
28 3 6 4
9 1 4 5
```

- A função deve ter a seguinte assinatura:

```
void escreveMatriz(int matriz[][100], char arquivo[], int n, int m);
```

Exemplo 4

```c
/* Esta função escreve uma matriz de inteiros em um arquivo.
 */
void escreveMatriz(int matriz[][100], char arquivo[], int n, int m)
{
    FILE *saida = fopen(arquivo, "w");
    fprintf(saida, "%d %d\n", n, m);

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            if (j > 0)
                fprintf(arquivo, " ");
            fprintf(arquivo, "%d ", matriz[i][j]);
        }
        fprintf(arquivo, "\n");
    }

    fclose(saida);
}
```

Aula de hoje

Exercícios

Exercício

Elabore um programa que lê um arquivo de texto de, no máximo, 100 linhas (e 100 colunas) e cria um arquivo com as linhas em ordem inversa.

Dica: utilize um vetor de strings (`char[100][100]`) para armazenar as linhas, e use a função `fgets` para ler uma linha completa do arquivo.

Aula de hoje

Próximas aulas

- Estruturas heterogêneas
- Alocação dinâmica
- Arquivos binários
- Revisão

- **Prova 03**
- **Trabalho Prático**

Perguntas?