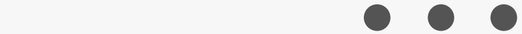
Arquivos

Aula 11 - Programação orientada a objetos



Professor Daniel Marques

INSTITUTO FEDERAL
Paraíba
Campus Campina Grande

Introdução

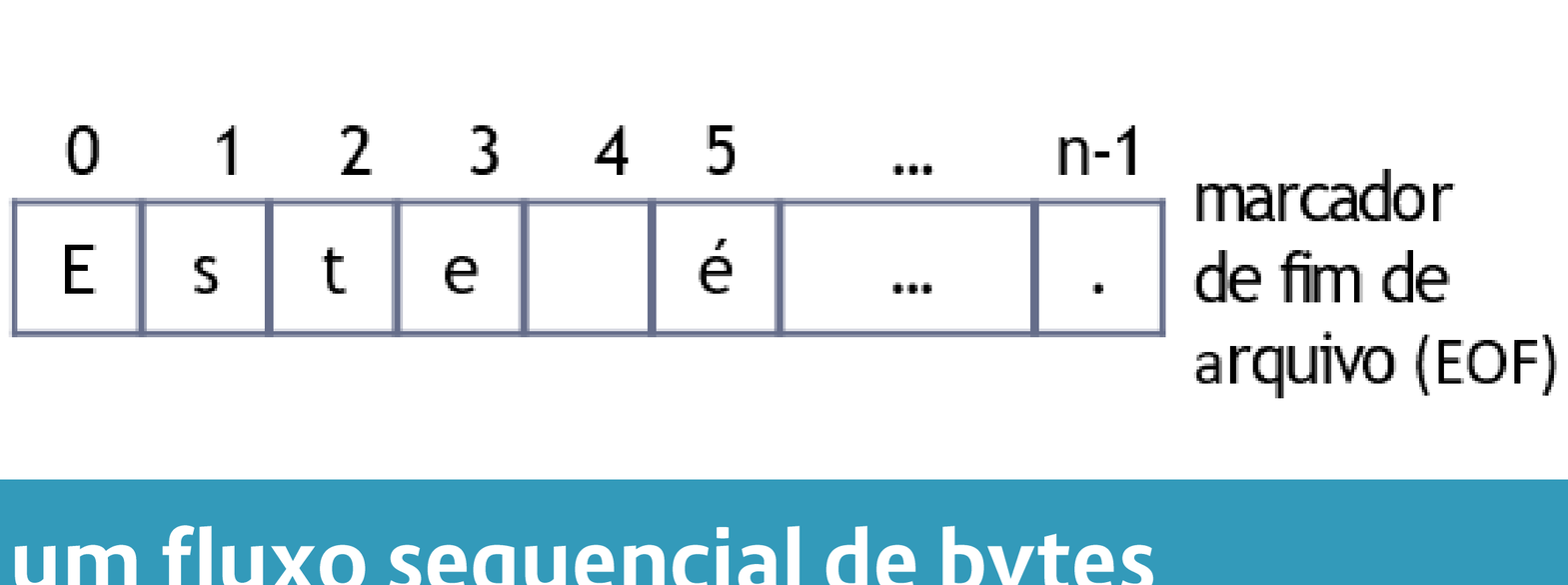
Quando não queremos perder os dados de um programa, precisamos armazená-los de forma permanente (persistente). Para isso, os computadores utilizam arquivos

Arquivos são armazenados em dispositivos de memória secundária (não-volátil): Disco rígidos, pendrives, cartões de memória

Arquivos e fluxos

Teste.txt

Este é o meu arquivo.



Cada arquivo é um fluxo sequencial de bytes

Em geral, podemos:

- Ler ou gravar arquivos de texto;
- Ler ou gravar arquivos binários (imagens, vídeos, aplicativos, etc);

Arquivos e fluxos

Manipular arquivos

Para manipular os arquivos, é necessário associá-los a um objeto de fluxo (ou stream)

Tipos de fluxos

- Entrada (input): lê informações do arquivo;
- Saída (output): grava informações no arquivo;
- Entrada/Saída: lê e grava informações no arquivo;

Arquivos e fluxos

Em C++, para manipular arquivos, deve-se incluir <iostream> e <fstream>

```
1  #include <iostream>
2  #include <fstream>
3
4
5
6
7
```

- **ofstream:** (output) objetos que escrevem dados em um arquivo.
- **ifstream:** (input) objetos que leêm dados em um arquivo.
- **fstream:** objetos que podem tanto ler como escrever em um arquivo.

Operações sobre arquivos: abrir arquivos

Através de instanciação e criação de objeto

```
1  #include <iostream>
2  #include <fstream>
3
4  int main(){
5      ofstream stream("arquivo.txt", ios::out);
6      return 0;
7  }
8
9
10
```

Operações sobre arquivos: abrir arquivos

Através do método *open( )*

```
1  #include <iostream>
2  #include <fstream>
3
4  int main(){
5      ofstream stream;
6      stream.open("arquivo.txt", ios::out);
7      return 0;
8  }
9
```

Mas não escrevemos nada dentro do arquivo. Apenas criamos um arquivo que vai ser salvo na sua máquina.

```cpp
#include <iostream>
#include <fstream>
using namespace std;

int main () {
      // ofstream - arquivo apenas para saída de dados
      ofstream arq1;

      // função open - abre o arquivo. Cria o arquivo caso ele não exista.
      arq1.open ("nomes.txt");

      // Insere nomes no arquivo (operador "<<")
      arq1 << "Bruno Gomes" << endl;
      arq1 << "Maria Dantas" << endl;

      // função close - fecha o arquivo
      arq1.close();
      return 0;
}
```

Exemplo

```cpp
#include <iostream>
#include <fstream>
using namespace std;

int main () {
    /* Outra forma: arquivo a ser aberto já é colocado diretamente
    após nome da variável arquivo */
    ofstream arq1("nomes.txt");

    //função is_open – testa se o arquivo está realmente aberto
    if (arq1.is_open()) {
        arq1 << "Bruno Gomes" << endl;
        arq1 << "Maria Dantas" << endl;
        arq1.close();
    }

    return 0;
}
```

Exemplo

Modos de abertura: função open

Modo	Descrição
`ios::app`	Acrescenta toda saída ao fim do arquivo.
`ios::ate`	Abre um arquivo para saída e move-se para o fim do arquivo (normalmente utilizado para acrescentar dados a um arquivo). Os dados podem ser gravados em qualquer lugar do arquivo.
`ios::in`	Abre um arquivo para a entrada.
`ios::out`	Abre um arquivo para a saída.
`ios::trunc`	Descarta o conteúdo do arquivo se ele existir (essa também é a ação-padrão de `ios::out`).
`ios::binary`	Abre um arquivo para entrada ou saída binária (isto é, não-texto).

- open(<nome_arquivo>, <modo>)
- <modo> é um valor ou a combinação dos valores na tabela acima

Modos de abertura: função open

ofstream arq1;

arq1.open ("nomes.txt", ios::out)

Modo padrão para ofstream (ios::out)

arq1.open ("nomes.txt", ios::out | ios::app)

Abre no final do arquivo “nomes.txt”. Ou seja, na posição depois do último caractere inserido

arq1.open ("nomes.txt", ios::app);

Mesmo que o anterior, uma vez que “out” é o padrão para ofstream

Exemplo: escrita no final do arquivo

```cpp
#include <iostream>
#include <fstream>
using namespace std;

int main () {
    ofstream arq1;

    //ios::app - abre o arquivo no final (depois do último caractere)
    arq1.open ("nomes.txt", ios::app);

    if (arq1.is_open()) {
        //tellp() - retorna a posição atual do apontador para escrita
        long pos = arq1.tellp();
        cout << "Posição atual no arquivo: " << pos << endl;
        // Insere nomes no final do arquivo (foi aberto com ios::app)
        arq1 << "Álvares de Azevedo" << endl;
        arq1 << "Machado de Assis" << endl;
        arq1.close();
    } else {
        cout << "ERRO: arquivo não foi aberto ou não existe" << endl;
    }
    return 0;
}
```

Operações sobre arquivos: leitura de arquivos

Semelhante à saída, mas usa a classe ifstream ao invés de ofstream

```
1  #include <iostream>
2  #include <fstream>
3  using namespace std;
4
5  int main () {
6      //modo 1
7      ifstream arquivo("novoArquivo.txt");
8
9      //modo 2
10     ifstream arquivo;
11     arquivo.open("novoArquivo.txt")
12     return 0;
13 }
```

Exemplo de leitura

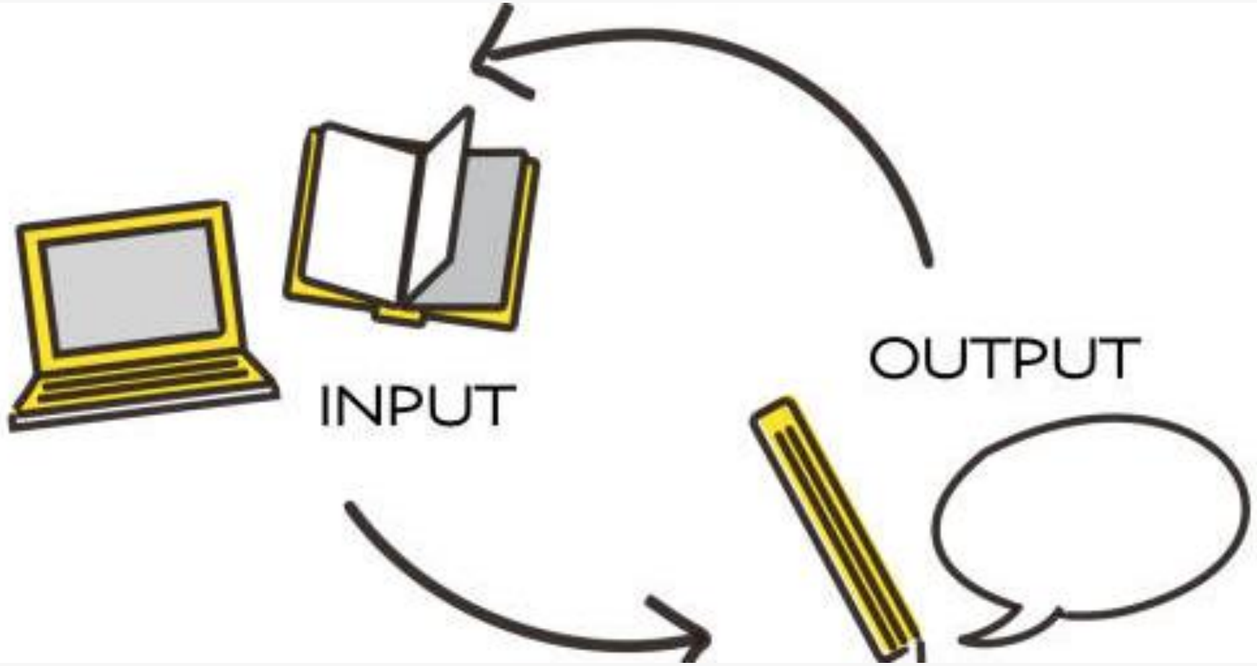
```cpp
#include <iostream>
#include <fstream>
#include <string>
using namespace std;

int main () {
    string linha;
    //ifstream - abre o arquivo apenas para leitura
    ifstream arq_in ("nomes.txt");

    if (arq_in.is_open()) {
        //eof() - retorna true ao atingir o fim do arquivo
        while (! arq_in.eof() ) {
            getline (arq_in, linha);
            cout << linha << endl;
        }
        arq_in.close();
    } else {
        cout << "ERRO: arquivo não foi aberto ou não existe" << endl;
    }
    return 0;
}
```

fstream: leitura e escrita

Caso se necessite tanto de operações de leitura quanto de escrita em um mesmo arquivo



Na abertura do arquivo deve-se indicar qual operação deve ser feita

- ios::in - leitura
- ios::out - escrita

fstream: exemplo

```cpp
#include <iostream>
#include <fstream>
#include <string>
using namespace std;

int main () {
    string linha;
    fstream arq; //fstream - leitura e escrita
    arq.open ("nomes.txt", ios::out | ios::app); //abre para escrita (ios::out)

    if (arq.is_open()) {
        //realiza uma escritra
        arq << "Roberto Carlos" << endl;
        arq.close();

        arq.open("nomes.txt", ios::in); //abre no modo de leitura (ios::in)
        while (! arq.eof() ) {
            getline (arq, linha);
            cout << linha << endl;
        }
        arq.close();
    } else {
        cout << "ERRO: arquivo não foi aberto ou não existe" << endl;
    }
    return 0;
}
```

Operações com arquivos: fechar arquivo

Fechar arquivo

- Através do método close
- stream.close();

Por quê fechar um arquivo?

- Importante fechar um arquivo, sempre que você não precise mais usá-lo
- Quando um arquivo é fechado explicitamente, todos os dados que estão no buffer são escritos no arquivo.
- Quando ele não é fechado, alguns dados podem ser perdidos ou o arquivo pode ser corrompido!

Alguma dúvida?

Não guardem dúvidas, perguntem

• • •

Referências

1. DA COSTA, Anderson Fabiano F. **Fundamentos de C++**. Instituto Federal da Paraíba. 2022.

2. Materiais de aula dos professores Guillermo Camara-Chavez, Tiago Maritan, Fred Guedes Pereira e Danielle Chaves.

3. DEITEL, **C++ Como Programar**, 5ª edição, Editora Prentice Hall, 2006

4. GOMES, Bruno E. G.; **Linguagem C++: Entrada e saída com arquivos**. Instituto Federal do Rio Grande do Norte. 2012.

5.