Processamento de arquivo

Grupo 1: Débora da Silva Costa
Noêmia Cíntia Sales Santos da Silva
Rafael dos Santos Lima
Sabrina Araújo Cardoso

INTRODUÇÃO

Iremos apresentar como se dá a criação, a leitura, a atualização, o acesso, bem como a gravação de objetos em arquivos sequenciais e em arquivos de acesso aleatório.

Tópicos abordados

01

Arquivo sequencial: Criação, gravação e leitura de dados aleatórios.

02

Arquivo de acesso aleatório: Criação, gravação, leitura e atualização de dados aleatórios.

03

Escrita e leitura de um arquivo aleatório sequencialmente, entrada e saída de objetos e síntese.

04

Apresentação de uma aplicação.

Arquivos Sequenciais

Arquivos sequenciais são arquivos em que os registros são acessados na ordem em que estão armazenados.

Código	Nome	E-mail	Data Nasc
1	Carlos	carlos@gmail.com	13/12/1977
2	Ana	ana@gmail.com	17/04/1975
3	Sílvio	silvio@gmail.com	28/01/1969
4	Joana	joana@gmail.com	31/07/1982

Biblioteca Fstream

Esta biblioteca possui funções e métodos para trabalhar com arquivos (files).

Ofstream

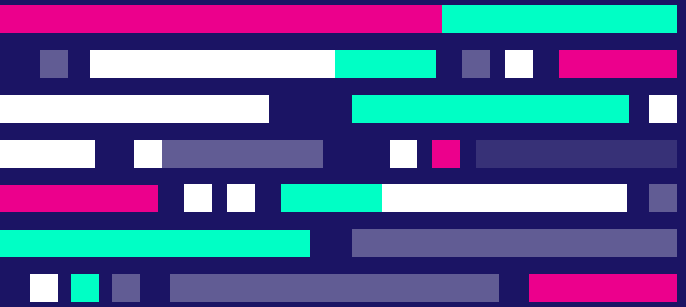
Apenas para saída

Ifstream

Apenas para entrada

Fstream

Serve para ler e escrever



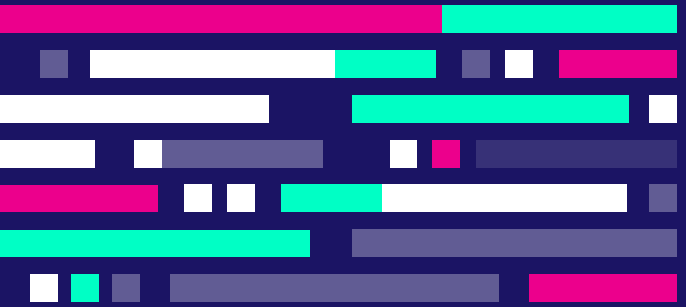
Criando um arquivo sequencial

OFSTREAM - permite acessar arquivos para gravação

```
 ofstream FuncionarioFile; //sai do sistema e entra no arquivo

FuncionarioFile.open("Funcionarios.dat", ios::out);
```

Modos de abertura	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 leitura
ios::out	Abre um arquivo para gravação
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)

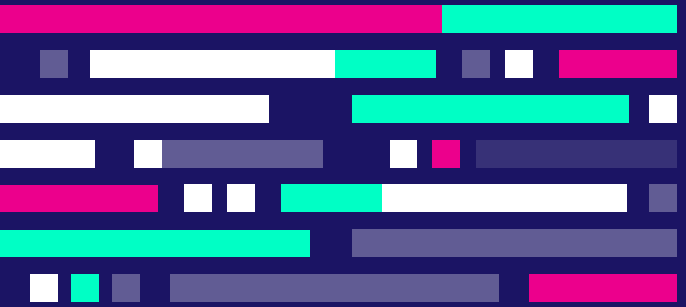


Lendo um arquivo sequencial

IFSTREAM - permite acessar arquivos para leitura

```
 ifstream FuncionarioFile; //sai do arquivo e entra no sistema

FuncionarioFile.open("Funcionarios.dat", ios::in);
```



Criando e lendo um arquivo sequencial

FSTREAM – permite acessar arquivos para criação e leitura

```
 fstream FuncionarioFile; //sai do arquivo e entra no sistema

FuncionarioFile.open("Funcionarios.dat"
,fstream::in|stream::out|stream::app);
```

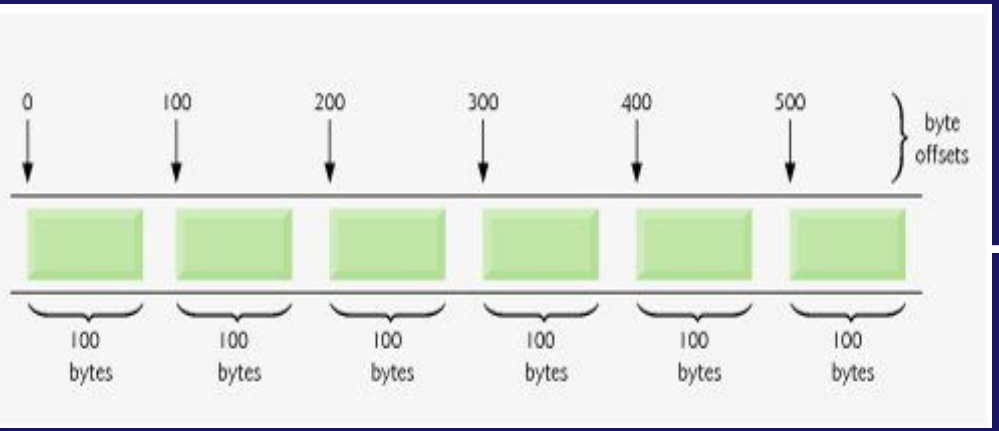
02

Criando e gravando dados aleatoriamente em um arquivo de acesso aleatório

Arquivos de acesso aleatório:

Ao contrário dos arquivos sequenciais, podemos acessar os arquivos de acesso aleatório em qualquer ordem que quisermos. Essas aplicações ditas como de acesso instantâneo. Algumas aplicações típicas de acesso instantâneo são os sistemas de ponto-de-venda, caixas automáticos e outros tipos de sistemas de processamento de transações que requerem acesso rápido a dados específicos.

Arquivo de acesso aleatório:



A técnicas mais simples requer que todos os registros no arquivo possuam um tamanho fixo. A utilização de registros de tamanho fixo facilita o cálculo da localização exata de qualquer registro relativamente ao início do arquivo.

Na figura ao lado é mostrado um esquema C++ de um arquivo de acesso aleatório, composto de registros de tamanho fixo (cada registro possui 100 bytes).

Quando nosso programa lê e grava arquivos aleatoriamente, ele trata o arquivo como um grande Array. Com Array sabemos que podemos adicionar imprimir ou remover valores em qualquer ordem.

Antes de fazermos toda essa operação,precisamos saber uma coisa importante primeiro. Essa coisa é o *ponteiro de posição do arquivo*, que se move sequencialmente, mas também podemos fazê-lo se mover aleatoriamente, através de alguma das funções que veremos a seguir.

FUNÇÃO	SINTAXE	EXPLICAÇÃO
seekg()	Arquivo.seekg(bytes, origem);	Estamos movendo o ponteiro de **entrada** para o local especificado **para leitura** usando essa função. **Arquivo** é o arquivo que queremos acessar. **Bytes** é o número de bytes que queremos pular. E a **origem** é o valor onde começamos a pular os bytes.
seekp()	Arquivo.seekp(bytes, origem);	Nesse caso, movemos o ponteiro de **saída** para o local especificado usando essa função. É igual a seekg(), mas para **escrita**.
tellg()	Arquivo.tellg();	**Retorna a posição** atual do ponteiro de **entrada**.
tellp()	Arquivo.tellp();	**Retorna a função** atual do ponteiro de **saída**.

Agora, veremos um pouco mais sobre a *função seekg()*. Função que nos permite mover o ponteiro de entrada para o local especificado, para fins de leitura dentro do arquivo.

ORIGEM	SINTAXE	EXPLICAÇÃO
ios::beg	Arquivo.seekg(0, ios::beg);	Usando essa sintaxe, o ponteiro de arquivos voltará ao **início** do arquivo.
ios::cur	Arquivo.seekg(0, ios::cur);	O ponteiro de arquivos mostrará sua **posição atual**.
ios::end	Arquivo.seekg(0, ios::end);	O ponteiro de arquivos apontará para o **final** do arquivo.

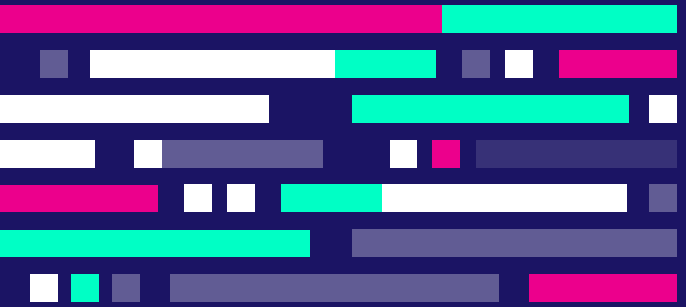
Agora, vamos mostrar algumas funções mais úteis.

FUNÇÃO	SINTAXE	EXPLICAÇÃO
Write ()	write((char *) &Obj, sizeof (Obj));	Nela, cada valor é **armazenado em seu formato binário** e cada registro é criado para ter um comprimento fixo. A função leva três argumentos: o primeiro é o **ponteiro para caractere**, o segundo é o **endereço** e o último é **tamanho do objeto.**
Read()	read((char *) &Obj, sizeof(Obj);	É o mesmo que Write(), mas no sentido de **leitura**.
Remove()	remove(NomeDoArquivo);	Usada para **remover** qualquer arquivo do registro.
Rename()	rename(nomeAntigo, novoNome);	Usada para **renomear** qualquer arquivo.

03

Escrita e leitura de um arquivo aleatório sequencialmente, entrada e saída de objeto.

3.1 Escrevendo e lendo um arquivo de acesso aleatório sequencialmente



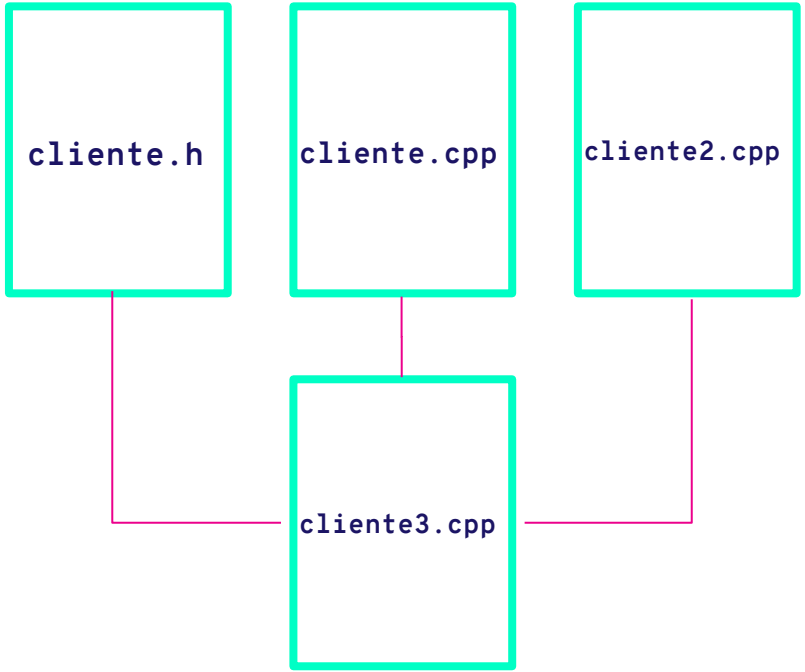
São escritos através do comando:

```
nomeDoObjeto.write((char *)(&objeto), sizeof(objeto));
```

São lidos através do comando:

```
nomeDoObjeto.read((char *)(&objeto), sizeof(objeto));
```

Exemplo



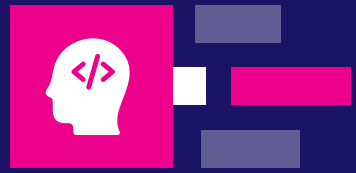
3.2 Entrada e saída de objetos

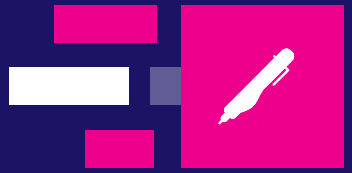
Sobrecarga de operadores

Função Friend

Ifstream

Leitura



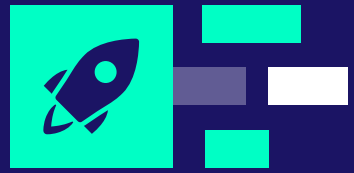


Osftream

Escrita

Exemplo

Programa



Obrigada!