

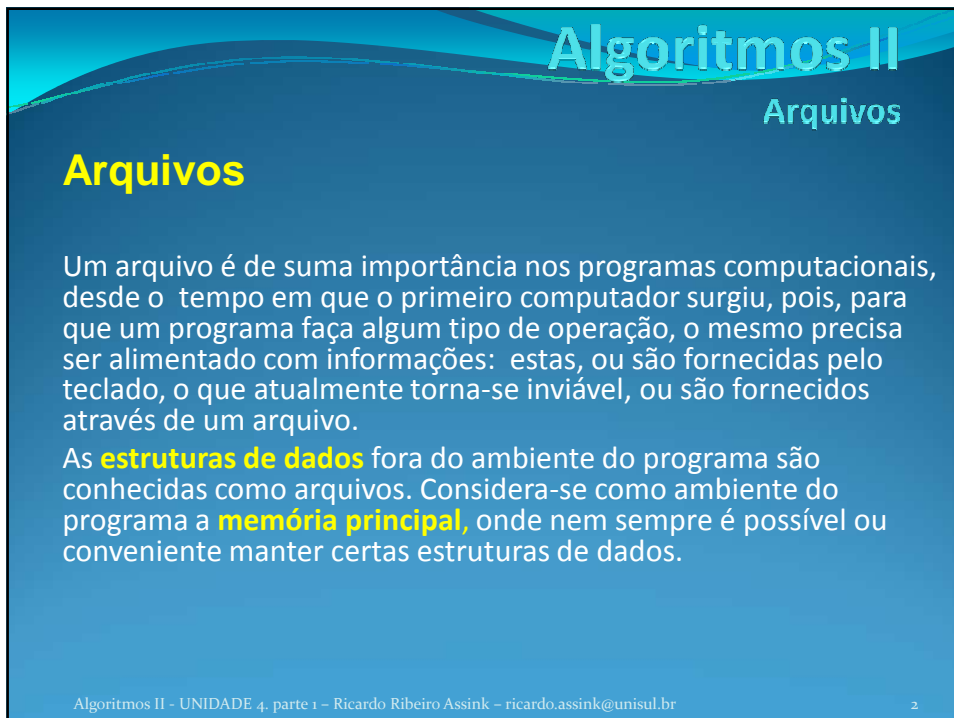


**Algoritmos II**

**Ricardo Ribeiro Assink**  
ricardo.assink@unisul.br  
ricardo@equipedigital.com

<http://www.ricardoassink.com.br/>

Algoritmos II - UNIDADE 4. parte 1 - Ricardo Ribeiro Assink - ricardo.assink@unisul.br 1



**Algoritmos II**  
**Arquivos**

**Arquivos**

Um arquivo é de suma importância nos programas computacionais, desde o tempo em que o primeiro computador surgiu, pois, para que um programa faça algum tipo de operação, o mesmo precisa ser alimentado com informações: estas, ou são fornecidas pelo teclado, o que atualmente torna-se inviável, ou são fornecidos através de um arquivo.

As **estruturas de dados** fora do ambiente do programa são conhecidas como arquivos. Considera-se como ambiente do programa a **memória principal**, onde nem sempre é possível ou conveniente manter certas estruturas de dados.

Algoritmos II - UNIDADE 4. parte 1 - Ricardo Ribeiro Assink - ricardo.assink@unisul.br 2

# Algoritmos II

## Arquivos

### Arquivos

Um arquivo é armazenado em um **dispositivo de memória secundária**, como discos, por exemplo, pode ser lido ou escrito por um programa.

Um arquivo é formado por uma coleção de registros, cada registro é composto por campos e cada campo possui suas características específicas. Um ou mais campos desse registro é considerado **campo-chave**, que é o campo que diferencia um registro dos demais, evitando duplicidade de informações.

Um sistema de banco de dados é composto por um ou vários arquivos, onde cada arquivo possui programas de manutenção que são: inclusão, exclusão lógica ou exclusão física, alteração, consulta geral, consulta específica e relatórios.

Algoritmos II - UNIDADE 4, parte 1 - Ricardo Ribeiro Assink - ricardo.assink@unisul.br 3

# Algoritmos II

## Arquivos

### Arquivos

As **operações básicas** que podem ser feitas em um arquivo através de um algoritmo são:

- obtenção de um registro do arquivo(consulta),
- inserção de um novo registro,
- modificação de um registro.
- exclusão de um registro.

Dependendo do tipo de problema, estas operações poderão ocorrer em maior ou menor número.

Algoritmos II - UNIDADE 4, parte 1 - Ricardo Ribeiro Assink - ricardo.assink@unisul.br 4

# Algoritmos II

## Arquivos

### Arquivos

A disposição dos registros no arquivo pode favorecer determinadas operações em detrimento de outras. O conhecimento das possibilidades de **organização dos registros** no arquivos permite ao programador escolher aquela que seja mais adequada à solução do seu problema em termos de eficácia e eficiência.

Basicamente, existem duas possibilidades de organização de arquivo: a **organização texto**, na qual os registros são obtidos ou inseridos no arquivo em ordem seqüencial, e a **organização binária**, em que o acesso do registro é feito em ordem aleatória. Outros tipos de organização são, na verdade, variações destas e, ainda que possam ser importantes em algumas aplicações, não serão tratadas aqui.

# Algoritmos II

## Arquivos

### Arquivos

A principal característica de um arquivo texto é a de que os dados são **armazenados contiguamente**, isto é um após o outro.

Como consequência, o acesso aos registros do arquivo, tanto na leitura quanto na escrita são feitos **seqüencialmente**, ou seja, a leitura de um registro só é possível após a leitura de todos os registros anteriores e a escrita de um registro só é feita após o ultimo registro.

Um arquivo do tipo texto, também conhecido por arquivo seqüencial, é um tipo especial de arquivo que, ao contrário do arquivo binário, pode ser editado normalmente através de um editor de textos qualquer. Ele é dito seqüencial porque a leitura tem que ser feita seqüencialmente do início ao fim do arquivo, não podendo desta forma, como é feito no arquivo binário através do comando **posicionarArquivo**, que posiciona de forma direta, o ponteiro de arquivo em um registro em particular.

# Algoritmos II

## Arquivos

### Arquivos

Nos arquivos do tipo texto, todas as informações lá armazenadas são texto (Literais), mesmo assim, é possível escrever no arquivo informações de qualquer tipo de dado simples (Inteiro, Real) as quais, ao serem fisicamente armazenadas no arquivo, serão automaticamente convertidas do seu tipo original para o tipo Literal.

Na processo de leitura quando é lida uma informação em um arquivo texto, a mesma precisa ser convertida para o tipo da variável necessário para o restante do processamento.

# Algoritmos II

## Arquivos

O acesso a um arquivo dentro do algoritmo deve ser feito de forma semelhante a uma variável. No algoritmo, o arquivo deve ser declarado e aberto, antes que tal acesso possa ser a leitura ou escrita. No final do algoritmo, ou quando houver necessidade, o arquivo deve ser fechado.

A declaração de um arquivo é feita através da especificação a seguir:

```

Algoritmo DeclaraçãoArquivoTexto
Variáveis
    <Nome_arquivo> : Arquivo Texto
Início
Fim.
  
```

**Onde:**

<Nome\_arquivo> é a referência ao arquivo dentro do algoritmo.  
Arquivo Texto são palavras reservadas

# Algoritmos II

## Arquivos

A **abertura de um arquivo** é feita através da especificação a seguir:

```

Algoritmo AberturaArquivoTexto
Variáveis
    <Nome_arquivo> : Arquivo Texto
Início
    abrirArquivo( <Nome_arquivo>, <Nome_arquivo_SO>,
                  <Utilização>, <Conteúdo> )
Fim.
  
```

**Onde:**

<Nome\_arquivo> é o arquivo que será aberto pelo comando abrir.

**abrirArquivo** é uma palavra reservada. Alguns autores pode utilizar a palavra Abra, OpenFile, etc...

<Nome\_arquivo\_SO> é o nome do arquivo no sistema operacional.

<Utilização> especifica se o arquivo será usado somente para **leitura** ou somente para **escrita**.

<Conteúdo> especifica se o comando de abertura deve continuar (.V.) a completar o arquivo existente ou esvaziá-lo (.F.) na hora de abrí-lo. Se não especificar nada o arquivo é esvaziado.

# Algoritmos II

## Arquivos

O **fechamento de um arquivo** é feito através da especificação a seguir:

```

Algoritmo FechamentoArquivoTexto
Variáveis
    <Nome_arquivo> : Arquivo Texto
Início
    fecharArquivo ( <Nome_arquivo> )
Fim.
  
```

**Onde:**

<Nome\_arquivo> é o arquivo que será fechando pelo comando fechar.

**fecharArquivo** é uma palavra reservada. Alguns autores pode utilizar a palavra Feche, CloseFile, etc..

# Algoritmos II

## Arquivos

### Exemplo de abertura e fechamento de Arquivo.

```

Algoritmo LeituraArquivoTexto
Constante
    NOME_ARQUIVO = "AGENDA.TXT"
Variáveis
    agenda : Arquivo Texto
Início
    abrirArquivo(agenda,NOME_ARQUIVO,"Leitura")
    fecharArquivo(agenda)
Fim.

```

# Algoritmos II

## Arquivos

A **escrita em um arquivo** é feito através da especificação a seguir:

```

Algoritmo EscrevendoArquivoTexto
Variáveis
    <Nome_arquivo> : Arquivo Texto
    <Nome_variável> : Literal
Início
    Escreva (<Nome_arquivo>,<Nome_variável>)
Fim.

```

Onde:

**Escreva** é palavra reservada.

<Nome\_arquivo> é o arquivo que será lido pelo comando escreva.

<Nome\_variável> é o nome da variável que contém o valor a ser escrito no arquivo. A variável será do tipo literal pois o arquivo é do tipo texto.

# Algoritmos II

## Arquivos

### Exemplo ESCRITA

```

import java.io.*;
import javax.swing.*;

public class Unidade4_01{

    static final String NOME_ARQUIVO = "AGENDA.TXT";

    public static void main(String args[]) {

        try{
            BufferedWriter agenda = null;

            String adicionar = JOptionPane.
showInputDialog("Adicionar novos dados no arquivo existente? (S/N)");

            if (adicionar.equalsIgnoreCase("S")) {

                //Abre o arquivo completando o que ja existe nele
                agenda = new BufferedWriter(new FileWriter(new File(NOME_ARQUIVO),true));

            } else {

                //Abre o arquivo zerando o seu conteudo
                agenda = new BufferedWriter(new FileWriter(new File(NOME_ARQUIVO)));

            }

        }
    }
}

```

**CONTINUA ->**

Algoritmos II - UNIDADE 4. parte 1 - Ricardo Ribeiro Assink - ricardo.assink@unisul.br

13

# Algoritmos II

## Arquivos

### Exemplo ESCRITA

```

String nome = JOptionPane.showInputDialog("Digite um nome ou ENTER para sair");

while (!nome.equals("")){

    agenda.write(nome);
    agenda.newLine();
    nome = JOptionPane.showInputDialog("Digite um nome ou ENTER para sair");

} // fim do while

agenda.close(); // fecha o arquivo

} catch (IOException e){

    JOptionPane.showMessageDialog(null, "Não abriu arquivo para escrita!");

} // fim do try / catch

} // fim do main

} // fim da class

```

Algoritmos II - UNIDADE 4. parte 1 - Ricardo Ribeiro Assink - ricardo.assink@unisul.br

14

# Algoritmos II

## Arquivos

A **leitura de um arquivo** é feito através da especificação a seguir:

```

Algoritmo LeituraArquivoTexto
Variáveis
    <Nome_arquivo> : Arquivo Texto
    <Nome_variável> : Literal

Início
    Leia (<Nome_arquivo>, <Nome_variável>)
Fim.
  
```

**Onde:**

**Leia** é palavra reservada.

**<Nome\_arquivo>** é o arquivo que será lido pelo comando **leia**.

**<Nome\_variável>** é o nome da variável que irá conter o valor lido do arquivo. A variável será do tipo literal.

# Algoritmos II

## Arquivos

### Exemplo LEITURA

```

import java.io.*;
import javax.swing.*;

public class Unidade4_02 {

    static final String NOME_ARQUIVO = "AGENDA.TXT";

    public static void main(String args[]){
        try{

            BufferedReader agenda = new BufferedReader(new FileReader(new File(NOME_ARQUIVO)));
            String nome = agenda.readLine();

            while (nome != null) {
                JOptionPane.showMessageDialog(null, "Nome: " + nome);
                nome = agenda.readLine();
            }

            agenda.close();
        }catch (IOException e) {
            JOptionPane.showMessageDialog(null, "Não abriu arquivo para leitura!");
        }
    }
}
  
```





Algoritmos II

Arquivos

**FIM**

Material Original: Osmar de Oliveira Braz Junior

Algoritmos II - UNIDADE 4, parte 1 - Ricardo Ribeiro Assink - [ricardo.assink@unisul.br](mailto:ricardo.assink@unisul.br)

17