

Tema: Introdução à programação III
Atividade: Arquivos em Java

01.) Editar e salvar um esboço de programa em Java:

```
/**
 * Exemplo0121
 *
 * @author
 * @version 01
 */

// ----- dependencias

import IO.*;

// ----- definicao da classe principal

public class Exemplo0121
{
// ----- definicao de metodo auxiliar

    /**
     * guardar dados em arquivo,
     * dada a quantidade deles.
     */
    public static void metodo01 ( )
    {
        // 1. definir dados
        int n;
        int k;
        String dado;

        FILE arquivo;

        // 2. identificar
        IO.println ( "Guardar dados em arquivos" );

        // 3. ler dado do teclado
        n = IO.readint ( "Quantos nomes? " );
```

```

// 4. testar a quantidade
if ( n <= 0 )
{
    IO.println ( "ERRO: Quantidade invalida." );
}
else
{
    // criar arquivo para gravar dados
    arquivo = new FILE ( FILE.OUTPUT, "DADOS1.TXT" );
    // guardar a quantidade de dados em arquivo
    arquivo.println ( ""+ n );
    // ler os outros dados do teclado
    for ( k = 1; k <= n; k = k + 1 )
    {
        // ler um dado do teclado
        dado = IO.readString ( "Nome = " );
        // guardar dado em arquivo
        arquivo.println ( ""+dado );
    } // fim repetir
    // fechar o arquivo (INDISPENSÁVEL para a gravacao)
    arquivo.close ( );
} // fim se

// 5. encerrar
IO.println ( );
IO.pause ( "Apertar ENTER para continuar." );
} // fim metodo01( )

// ----- definicao do metodo principal

/**
 * main() – metodo principal
 */
public static void main ( String [ ] args )
{
    // identificar
    IO.println ( "EXEMPLO0121 - Programa em Java" );
    IO.println ( "Autor: _____" );
    // executar o metodo auxiliar
    metodo01 ( );
    // encerrar
    IO.pause ( "Apertar ENTER para terminar." );
} // fim main( )
} // fim class Exemplo0121

```

02.) Compilar o programa.

Se houver erros, resolvê-los e compilar novamente, até que todos tenham sido resolvidos.

Se não houver erros, seguir para o próximo passo.

Em caso de dúvidas, consultar a apostila, recorrer aos monitores ou apresentá-las ao professor.

03.) Executar o programa.

Observar as saídas.

Registrar os dados e os resultados.

Em caso de erro (ou dúvida), usar comentários para registrar a ocorrência e, posteriormente, tentar resolvê-lo (ou esclarecer a dúvida).

04.) Copiar a versão atual do programa para outra nova – Exemplo0122.java.

05.) Editar mudanças no nome do programa e versão.

Acrescentar outro método para ler e mostrar os dados gravados.

Na parte principal, editar a chamada do método para o novo.

```
/**
 * consultar dados em arquivo.
 */
public static void metodo02 ( )
{
    // 1. definir dados
    String linha;
    FILE arquivo;

    // 2. identificar
    IO.println ( "Consultar dados em arquivos" );

    // 3. abrir arquivo para a leitura
    arquivo = new FILE ( FILE.INPUT, "DADOS1.TXT" );

    // 4. ler linhas do arquivo
    // tentar ler uma linha do arquivo
    linha = arquivo.readLine ( );
    // repetir enquanto houver dado
    while ( ! arquivo.eof ( ) )
    {
        // mostrar dado na tela
        IO.println ( "" + linha );
        // tentar ler outra linha do arquivo
        linha = arquivo.readLine ( );
    } // fim repetir
    // fechar o arquivo (RECOMENDAVEL para a leitura)
    arquivo.close ( );

    // 5. encerrar
    IO.println ( );
    IO.pause ( "Apertar ENTER para continuar." );
} // fim metodo02 ( )
```

OBS.:

Todo o conteúdo será lido como texto, inclusive a quantidade, sem distinções.

06.) Compilar o programa novamente.

Se houver erros, resolvê-los e compilar novamente, até que todos tenham sido resolvidos.

Se não houver erros, seguir para o próximo passo.

07.) Executar o programa.

Observar as saídas.

Registrar os dados e os resultados.

08.) Copiar a versão atual do programa para outra nova – Exemplo0123.java.

**Abrir o arquivo DADOS1.TXT, e conferir se a quantidade de dados está na primeira linha.
Se não estiver, incluir esse valor e salvar novamente o arquivo.**

09.) Editar mudanças no nome do programa e versão.

Acrescentar outro método para ler e mostrar os dados gravados, conhecida a quantidade.

Na parte principal, editar a chamada do método para o novo.

```
// ----- definicao de metodo auxiliar
/**
 * recuperar dados em arquivo,
 * dada a quantidade deles na primeira linha.
 */
public static void metodo03 ( )
{
    // 1. definir dados
    int    n, k;
    String linha;
    FILE   arquivo;

    // 2. identificar
    IO.println ( "Consultar dados em arquivos" );

    // 3. abrir arquivo para a leitura
    arquivo = new FILE ( FILE.INPUT, "DADOS1.TXT" );

    // 4. ler quantidade de dados do arquivo
    // 4.1 ler uma linha do arquivo
    linha = arquivo.readLine ( );
    // 4.2 converter conteudo para valor inteiro
    n = IO.getint ( linha );

    // 5. testar a quantidade
    if ( n <= 0 )
    {
        IO.println ( "ERRO: Quantidade invalida." );
    }
    else
    {
        // mostrar a quantidade de dados
        IO.println ( "Quantidade de dados = " + n );
        // ler os outros dados do arquivo
        for ( k = 1; k <= n; k = k + 1 )
        {
            // ler uma linha do arquivo
            linha = arquivo.readLine ( );
            // mostrar dado na tela
            IO.println ( "" + linha );
        } // fim repetir
        // fechar o arquivo (RECOMENDAVEL para a leitura)
        arquivo.close ( );
    } // fim se

    // 6. encerrar
    IO.println ( );
    IO.pause ( "Apertar ENTER para continuar." );
} // fim metodo03 ( )
```

OBS.:

Observar a necessidade da conversão de formato texto para valor inteiro.

- 10.) Compilar o programa novamente.
Se houver erros, resolvê-los e compilar novamente, até que todos tenham sido resolvidos.
Se não houver erros, seguir para o próximo passo.
- 11.) Executar o programa.
Observar as saídas.
Registrar os dados e os resultados.
- 12.) Copiar a versão atual do programa para outra nova – Exemplo0124.java.
- 13.) Editar mudanças no nome do programa e versão.
Acrescentar outro método para gravar valores inteiros em outro arquivo.
Na parte principal, editar a chamada do método para o novo.

// ----- definicao de metodo auxiliar

```
/**
 * guardar dados inteiros em arquivo,
 * dada a quantidade deles.
 */
public static void metodo04 ( )
{
    // 1. definir dados
    int n, k, dado;
    FILE arquivo;
    // 2. identificar
    IO.println ( "Guardar dados inteiros em arquivos" );
    // 3. ler dado do teclado
    n = IO.readint ( "Fornecer a quantidade de valores: " );
    // 4. testar a quantidade
    if ( n <= 0 )
    {
        IO.println ( "ERRO: Quantidade invalida." );
    }
    else
    {
        // criar arquivo para gravar dados
        arquivo = new FILE ( FILE.OUTPUT, "DADOS2.TXT" );
        // guardar a quantidade de dados
        arquivo.println ( ""+n );
        // ler os outros dados do teclado
        for ( k = 1; k <= n; k = k + 1 )
        {
            // ler um dado do teclado
            dado = IO.readint ( "Dado = " );
            // guardar dado em arquivo
            arquivo.println ( ""+dado );
        } // fim repetir
        // fechar o arquivo (INDISPENSÁVEL para a gravacao)
        arquivo.close ( );
    } // fim se
    // 5. encerrar
    IO.println ( );
    IO.pause ( "Apertar ENTER para continuar." );
} // fim metodo04 ( )
```

- 14.) Compilar o programa novamente.
Se houver erros, resolvê-los e compilar novamente, até que todos tenham sido resolvidos.
Se não houver erros, seguir para o próximo passo.
- 15.) Executar o programa.
Observar as saídas.
Registrar os dados e os resultados.
- 16.) Copiar a versão atual do programa para outra nova – Exemplo0125.java.
- 17.) Editar mudanças no nome do programa e versão.
Acrescentar outro método para ler valores inteiros de arquivo, conhecida a quantidade.
Na parte principal, editar a chamada do método para o novo.

// ----- definicao de metodo auxiliar

```
/**
 * recuperar dados inteiros em arquivo,
 * dada a quantidade deles na primeira linha.
 */
public static void metodo05 ( )
{
    // 1. definir dados
    int    n, k, dado;
    String linha;
    FILE  arquivo;

    // 2. identificar
    IO.println ( );
    IO.println ( "Consultar dados em arquivos" );
    IO.println ( );

    // 3. abrir arquivo para a leitura
    arquivo = new FILE ( FILE.INPUT, "DADOS2.TXT" );

    // 4. ler quantidade de dados do arquivo
    // 4.1 ler uma linha do arquivo
    linha = arquivo.readLine ( );
    // 4.2 converter conteudo para valor inteiro
    n = IO.getint ( linha );
```

```

// 5. testar a quantidade
if ( n <= 0 )
{
    IO.println ( "ERRO: Quantidade invalida." );
}
else
{
    // mostrar a quantidade de dados
    IO.println ( "Quantidade de dados = " + n );
    // ler os outros dados do arquivo
    for ( k = 1; k <= n; k = k + 1 )
    {
        // ler uma linha do arquivo
        linha = arquivo.readLine ( );
        // converter conteudo para valor inteiro
        dado = IO.getInt ( linha );
        // mostrar dado na tela
        IO.println ( "" + dado );
    } // fim repetir
    // fechar o arquivo (RECOMENDAVEL para a leitura)
    arquivo.close ( );
} // fim se

// 6. encerrar
IO.println ( );
IO.pause ( "Apertar ENTER para continuar." );
} // fim metodo05 ( )

```

OBS.:

Observar a necessidade da conversão de formato texto para valor inteiro.

18.) Compilar o programa novamente.

Se houver erros, resolvê-los e compilar novamente, até que todos tenham sido resolvidos.

Se não houver erros, seguir para o próximo passo.

19.) Executar o programa.

Observar as saídas.

Registrar os dados e os resultados.

20.) Copiar a versão atual do programa para outra nova – Exemplo0126.java.

- 21.) Editar mudanças no nome do programa e versão.
Acrescentar outro método para gravar dados reais em arquivo.
Na parte principal, incluir uma chamada para esse método.

```
// ----- definicao de metodo auxiliar

/**
 * guardar dados reais em arquivo,
 * dada a quantidade deles.
 */
public static void metodo06 ( )
{
    // 1. definir dados
    int    n, k;
    double dado;
    FILE   arquivo;
    // 2. identificar
    IO.println ( "Guardar dados reais em arquivos" );
    // 3. ler dado do teclado
    n = IO.readint ( "Fornecer a quantidade de valores: " );
    // 4. testar a quantidade
    if ( n <= 0 )
    {
        IO.println ( "ERRO: Quantidade invalida." );
    }
    else
    {
        // criar arquivo para gravar dados
        arquivo = new FILE ( FILE.OUTPUT, "DADOS3.TXT" );
        // guardar a quantidade de dados
        arquivo.println ( ""+n );
        // ler os outros dados do teclado
        for ( k = 1; k <= n; k = k + 1 )
        {
            // ler um dado do teclado
            dado = IO.readdouble ( "Dado = " );
            // guardar dado em arquivo
            arquivo.println ( ""+dado );
        } // fim repetir
        // fechar o arquivo (INDISPENSÁVEL para a gravacao)
        arquivo.close ( );
    } // fim se
    // 5. encerrar
    IO.println ( );
    IO.pause ( "Apertar ENTER para continuar." );
} // fim metodo06 ( )
```

- 22.) Compilar o programa novamente.
Se houver erros, resolvê-los e compilar novamente, até que todos tenham sido resolvidos.
Se não houver erros, seguir para o próximo passo.
- 23.) Executar o programa.
Observar as saídas.
Registrar os dados e os resultados.

24.) Copiar a versão atual do programa para outra nova – Exemplo0127.java.

25.) Editar mudanças no nome do programa e versão.

Acrescentar um método para ler dados reais de arquivo, conhecida a quantidade.
Na parte principal, incluir uma chamada para esse método.

```
// ----- definicao de metodo auxiliar
```

```
/**
 * recuperar dados reais em arquivo,
 * dada a quantidade deles na primeira linha.
 */
public static void metodo07 ( )
{
    // 1. definir dados
    int    n, k;
    double dado;
    String linha;
    FILE    arquivo;
    // 2. identificar
    IO.println ( "Consultar dados em arquivos" );
    // 3. abrir arquivo para a leitura
    arquivo = new FILE ( FILE.INPUT, "DADOS3.TXT" );
    // 4. ler quantidade de dados do arquivo
    // 4.1 ler uma linha do arquivo
    linha = arquivo.readLine ( );
    // 4.2 converter conteudo para valor inteiro
    n = IO.getint ( linha );
    // 5. testar a quantidade
    if ( n <= 0 )
    {
        IO.println ( "ERRO: Quantidade invalida." );
    }
    else
    {
        // mostrar a quantidade de dados
        IO.println ( "Quantidade de dados = " + n );
        // ler os outros dados do arquivo
        for ( k = 1; k <= n; k = k + 1 )
        {
            // ler uma linha do arquivo
            linha = arquivo.readLine ( );
            // converter conteudo para valor inteiro
            dado = IO.getdouble ( linha );
            // mostrar dado na tela
            IO.println ( "" + dado );
        } // fim repetir
        // fechar o arquivo (RECOMENDAVEL para a leitura)
        arquivo.close ( );
    } // fim se
    // 6. encerrar
    IO.println ( );
    IO.pause ( "Apertar ENTER para continuar." );
} // fim metodo07 ( )
```

OBS.: Observar a necessidade da conversão de formato texto para valor real.

- 26.) Compilar o programa novamente.
Se houver erros, resolvê-los e compilar novamente, até que todos tenham sido resolvidos.
Se não houver erros, seguir para o próximo passo.
- 27.) Executar o programa.
Observar as saídas.
Registrar os dados e os resultados.
- 28.) Copiar a versão atual do programa para outra nova – Exemplo0128.java.
- 29.) Editar mudanças no nome do programa e versão.
Acrescentar um método para gravar linhas de texto em um arquivo até dizer para parar.
Na parte principal, incluir chamadas para essa função.

```
// ----- definicao de metodo auxiliar

/**
 * guardar dados em arquivo.
 */
public static void metodo08 ( )
{
    // 1. definir dados
    String linha;
    FILE arquivo;
    // 2. identificar
    IO.println ( );
    IO.println ( "Guardar dados em arquivos" );
    IO.println ( );
    // 3. criar arquivo para gravar dados
    arquivo = new FILE ( FILE.OUTPUT, "TEXT0.TXT" );
    // 4. ler uma linha do teclado
    linha = IO.readLine ( "" );
    // 5. repetir enquanto quiser guardar dados
    while ( ! linha.equals ( "PARAR" ) )
    {
        // guardar linha em arquivo
        arquivo.println ( ""+linha );
        // ler outra linha do teclado
        linha = IO.readLine ( "" );
    } // fim repetir
    // 6. fechar o arquivo (INDISPENSÁVEL para a gravacao)
    arquivo.close ( );
    // 7. encerrar
    IO.println ( );
    IO.pause ( "Apertar ENTER para continuar." );
} // fim metodo08 ( )
```

- 30.) Compilar o programa novamente.
Se houver erros, resolvê-los e compilar novamente, até que todos tenham sido resolvidos.
Se não houver erros, seguir para o próximo passo.
- 31.) Executar o programa. Observar as saídas. Registrar os dados e os resultados.
- 32.) Copiar a versão atual do programa para outra nova – Exemplo0129.java.

33.) Editar mudanças no nome do programa e versão.

Acrescentar um método para ler linhas de texto de um arquivo, palavra por palavra.

Na parte principal, incluir chamadas para essa função.

```
// ----- definicao de metodo auxiliar

/**
 * ler dados de arquivo.
 */
public static void metodo09 ( )
{
    // 1. definir dados
    String dado;

    FILE arquivo;

    // 2. identificar
    IO.println ( );
    IO.println ( "Ler dados de arquivos" );
    IO.println ( );

    // 3. abrir arquivo para ler dados
    arquivo = new FILE ( FILE.INPUT, "TEXT0.TXT" );

    // 4. tentar ler um dado do teclado
    dado = arquivo.read ( );

    // 5. repetir enquanto houver dados
    while ( ! arquivo.eof ( ) )
    {
        // mostrar dado lido de arquivo
        IO.println ( ""+dado );
        // ler outro dado do teclado
        dado = arquivo.read ( );
    } // fim repetir

    // 6. fechar o arquivo (RECOMENDAVEL para a leitura)
    arquivo.close ( );

    // 7. encerrar
    IO.println ( );
    IO.pause ( "Apertar ENTER para continuar." );
} // fim metodo09 ( )
```

34.) Compilar o programa novamente.

Se houver erros, resolvê-los e compilar novamente, até que todos tenham sido resolvidos.

Se não houver erros, seguir para o próximo passo.

35.) Executar o programa. Observar as saídas. Registrar os dados e os resultados.

36.) Copiar a versão atual do programa para outra nova – Exemplo0130.java.

37.) Editar mudanças no nome do programa e versão.

Acrescentar um método para ler valores inteiros de um arquivo, mais de um valor por linha.

Na parte principal, incluir chamadas para esse método.

```
// ----- definicao de metodo auxiliar
```

```
/**
 * ler dados de arquivo.
 */
public static void metodo10 ( )
{
    // 1. definir dados
    String dado;
    int valor;
    FILE arquivo;
    // 2. identificar
    IO.println ( );
    IO.println ( "Ler dados de arquivos" );
    IO.println ( );
    // 3. abrir arquivo para ler dados
    arquivo = new FILE ( FILE.INPUT, "TEXT0.TXT" );
    // 4. tentar ler um dado do teclado
    dado = arquivo.read ( );
    // 5. repetir enquanto houver dados
    while ( ! arquivo.eof ( ) )
    {
        // mostrar dado lido de arquivo
        if ( dado != null &&
            dado.length( ) > 0 )
        {
            IO.println ( ""+IO.getint(dado) );
        } // fim se
        // ler outro dado do teclado
        dado = arquivo.read ( );
    } // fim repetir
    // 6. fechar o arquivo (RECOMENDAVEL para a leitura)
    arquivo.close ( );
    // 7. encerrar
    IO.println ( );
    IO.pause ( "Apertar ENTER para continuar." );
} // fim metodo10 ( )
```

OBS.:

Observar a necessidade de verificar a existência de dado antes de tentar fazer a conversão.

38.) Compilar o programa novamente.

Se houver erros, resolvê-los e compilar novamente, até que todos tenham sido resolvidos.

Se não houver erros, seguir para o próximo passo.

39.) Executar o programa.

Observar as saídas.

Registrar os dados e os resultados.

Exercícios:

DICAS GERAIS: Consultar o Anexo Java 02 na apostila para outros exemplos.

Prever, realizar e registrar todos os testes efetuados.

- 01.) Fazer um programa (Exemplo0131) com método para ler um valor inteiro do teclado e gravar essa quantidade em valores ímpares em ordem decrescente terminando em 3.
- 02.) Fazer um programa (Exemplo0132) com método para ler um valor inteiro do teclado e gravar essa quantidade em múltiplos de 3 em ordem crescente começando em 3.
- 03.) Fazer um programa (Exemplo0133) com método para ler um valor inteiro do teclado e gravar essa quantidade em valores da sequência: 1 3 6 9 12 15 ...
- 04.) Fazer um programa (Exemplo0134) com método para ler um valor inteiro do teclado e gravar essa quantidade em valores decrescentes da sequência: ... 1/81 1/27 1/9 1/3 1.
- 05.) Fazer um programa (Exemplo0135) com método para ler uma cadeia de caracteres e para gravar cada símbolo separadamente, e na ordem inversa, um por linha.
- 06.) Fazer um programa (Exemplo0136) com função para calcular a soma dos primeiros valores pares positivos começando em 2. Testar essa função para quantidades diferentes. Gravar cada termo e o resultado.
- 07.) Fazer um programa (Exemplo0137) com função para calcular a soma dos inversos dos primeiros valores ímpares positivos começando em 5. Testar essa função para quantidades diferentes. Gravar cada termo e o resultado.
- 08.) Fazer um programa (Exemplo0138) com função para calcular certo termo par da série de Fibonacci. Testar essa função para quantidades diferentes. Gravar cada termo e o resultado.
- 09.) Fazer um programa (Exemplo0139) com função para calcular a quantidade de maiúsculas em uma cadeia de caracteres. Testar essa função para quantidades diferentes. Gravar cada maiúscula e o resultado.
- 10.) Fazer um programa (Exemplo0140) com função para contar dígitos em uma cadeia de caracteres. Testar essa função para quantidades diferentes. Gravar cada dígito e o resultado.

Tarefas extras:

- E1.) Fazer um programa ler um valor inteiro do teclado e gravar em arquivo os seus divisores em ordem decrescente.
- E2.) Fazer um programa ler palavras de um arquivo, uma por linha, e contar quantas começam com a letra 'a' (ou 'A').