Módulo 1

Introdução à Programação 1

Apêndice C

Respostas dos exercícios

Versão 1.0 - Jan/2007

Autor
Florence Tiu Balagtas

Equipe
Joyce Avestro
Florence Balagtas
Rommel Feria
Reginald Hutcherson
Rebecca Ong
John Paul Petines
Sang Shin
Raghavan Srinivas
Matthew Thompson

Necessidades para os Exercícios

Sistemas Operacionais Suportados

NetBeans IDE 5.5 para os seguintes sistemas operacionais:

3. Microsoft Windows XP Profissional SP2 ou superior
4. Mac OS X 10.4.5 ou superior
5. Red Hat Fedora Core 3
6. Solaris™ 10 Operating System (SPARC® e x86/x64 Platform Edition)

NetBeans Enterprise Pack, poderá ser executado nas seguintes plataformas:

1. Microsoft Windows 2000 Profissional SP4
2. Solaris™ 8 OS (SPARC e x86/x64 Platform Edition) e Solaris 9 OS (SPARC e x86/x64 Platform Edition)
3. Várias outras distribuições Linux

Configuração Mínima de Hardware

Nota: IDE NetBeans com resolução de tela em 1024x768 pixel

Sistema Operacional	Processador	Memória	HD Livre
Microsoft Windows	500 MHz Intel Pentium III workstation ou equivalente	512 MB	850 MB
Linux	500 MHz Intel Pentium III workstation ou equivalente	512 MB	450 MB
Solaris OS (SPARC)	UltraSPARC II 450 MHz	512 MB	450 MB
Solaris OS (x86/x64 Platform Edition)	AMD Opteron 100 Série 1.8 GHz	512 MB	450 MB
Mac OS X	PowerPC G4	512 MB	450 MB

Configuração Recomendada de Hardware

Sistema Operacional	Processador	Memória	HD Livre
Microsoft Windows	1.4 GHz Intel Pentium III workstation ou equivalente	1 GB	1 GB
Linux	1.4 GHz Intel Pentium III workstation ou equivalente	1 GB	850 MB
Solaris OS (SPARC)	UltraSPARC IIIi 1 GHz	1 GB	850 MB
Solaris OS (x86/x64 Platform Edition)	AMD Opteron 100 Series 1.8 GHz	1 GB	850 MB
Mac OS X	PowerPC G5	1 GB	850 MB

Requerimentos de Software

NetBeans Enterprise Pack 5.5 executando sobre Java 2 Platform Standard Edition Development Kit 5.0 ou superior (JDK 5.0, versão 1.5.0_01 ou superior), contemplando a Java Runtime Environment, ferramentas de desenvolvimento para compilar, depurar, e executar aplicações escritas em linguagem Java. Sun Java System Application Server Platform Edition 9.

1. Para **Solaris**, **Windows**, e **Linux**, os arquivos da JDK podem ser obtidos para sua plataforma em http://java.sun.com/j2se/1.5.0/download.html
2. Para **Mac OS X**, Java 2 Plataform Standard Edition (J2SE) 5.0 Release 4, pode ser obtida diretamente da Apple's Developer Connection, no endereço: http://developer.apple.com/java (é necessário registrar o download da JDK).

Para mais informações:
http://www.netbeans.org/community/releases/55/relnotes.html

Colaboradores que auxiliaram no processo de tradução e revisão

Alexandre Mori
Alexis da Rocha Silva
Aline Sabbatini da Silva Alves
Allan Wojcik da Silva
André Luiz Moreira
Andro Márcio Correa Louredo
Antoniele de Assis Lima
Antonio Jose R. Alves Ramos
Aurélio Soares Neto
Bruno da Silva Bonfim
Bruno dos Santos Miranda
Bruno Ferreira Rodrigues
Carlos Alberto Vitorino de Almeida
Carlos Alexandre de Sene
Carlos André Noronha de Sousa
Carlos Eduardo Veras Neves
Cleber Ferreira de Sousa
Cleyton Artur Soares Urani
Cristiano Borges Ferreira
Cristiano de Siqueira Pires
Derlon Vandri Aliendres
Fabiano Eduardo de Oliveira
Fábio Bombonato
Fernando Antonio Mota Trinta
Flávio Alves Gomes
Francisco das Chagas
Francisco Marcio da Silva
Gilson Moreno Costa
Givailson de Souza Neves
Gustavo Henrique Castellano
Hebert Julio Gonçalves de Paula
Heraldo Conceição Domingues
Hugo Leonardo Malheiros Ferreira
Ivan Nascimento Fonseca
Jacqueline Susann Barbosa
Jader de Carvalho Belarmino
João Aurélio Telles da Rocha
João Paulo Cirino Silva de Novais
João Vianney Barrozo Costa
José Augusto Martins Nieviadonski
José Leonardo Borges de Melo
José Ricardo Carneiro
Kleberth Bezerra G. dos Santos
Lafaiete de Sá Guimarães
Leandro Silva de Morais
Leonardo Leopoldo do Nascimento
Leonardo Pereira dos Santos
Leonardo Rangel de Melo Filardi
Lucas Mauricio Castro e Martins
Luciana Rocha de Oliveira
Luís Carlos André
Luís Octávio Jorge V. Lima
Luiz Fernandes de Oliveira Junior
Luiz Victor de Andrade Lima
Manoel Cotts de Queiroz
Marcello Sandi Pinheiro
Marcelo Ortolan Pazzetto
Marco Aurélio Martins Bessa
Marcos Vinicius de Toledo
Maria Carolina Ferreira da Silva
Massimiliano Giroldi
Mauricio Azevedo Gamarra
Mauricio da Silva Marinho
Mauro Cardoso Mortoni
Mauro Regis de Sousa Lima
Namor de Sá e Silva
Néres Chaves Rebouças
Nolyanne Peixoto Brasil Vieira
Paulo Afonso Corrêa
Paulo José Lemos Costa
Paulo Oliveira Sampaio Reis
Pedro Antonio Pereira Miranda
Pedro Henrique Pereira de Andrade
Renato Alves Félix
Renato Barbosa da Silva
Reyderson Magela dos Reis
Ricardo Ferreira Rodrigues
Ricardo Ulrich Bomfim
Robson de Oliveira Cunha
Rodrigo Pereira Machado
Rodrigo Rosa Miranda Corrêa
Rodrigo Vaez
Ronie Dotzlaw
Rosely Moreira de Jesus
Seire Pareja
Sergio Pomerancblum
Silvio Sznifer
Suzana da Costa Oliveira
Tásio Vasconcelos da Silveira
Thiago Magela Rodrigues Dias
Tiago Gimenez Ribeiro
Vanderlei Carvalho Rodrigues Pinto
Vanessa dos Santos Almeida
Vastí Mendes da Silva Rocha
Wagner Eliezer Roncoletta

Auxiliadores especiais

Revisão Geral do texto para os seguintes Países:

- **Brasil** – Tiago Flach
- **Guiné Bissau** – Alfredo Cá, Bunene Sisse e Buon Olossato Quebi – ONG Asas de Socorro

Coordenação do DFJUG

- **Daniel deOliveira** – JUGLeader responsável pelos acordos de parcerias
- **Luci Campos -** Idealizadora do DFJUG responsável pelo apoio social
- **Fernando Anselmo -** Coordenador responsável pelo processo de tradução e revisão, disponibilização dos materiais e inserção de novos módulos
- **Regina Mariani -** Coordenadora responsável pela parte jurídica
- **Rodrigo Nunes -** Coordenador responsável pela parte multimídia
- **Sérgio Gomes Veloso -** Coordenador responsável pelo ambiente JEDI™ (Moodle)

Agradecimento Especial

John Paul Petines – Criador da Iniciativa JEDI™
Rommel Feria – Criador da Iniciativa JEDI™

Lição 1 – Introdução à programação de computadores

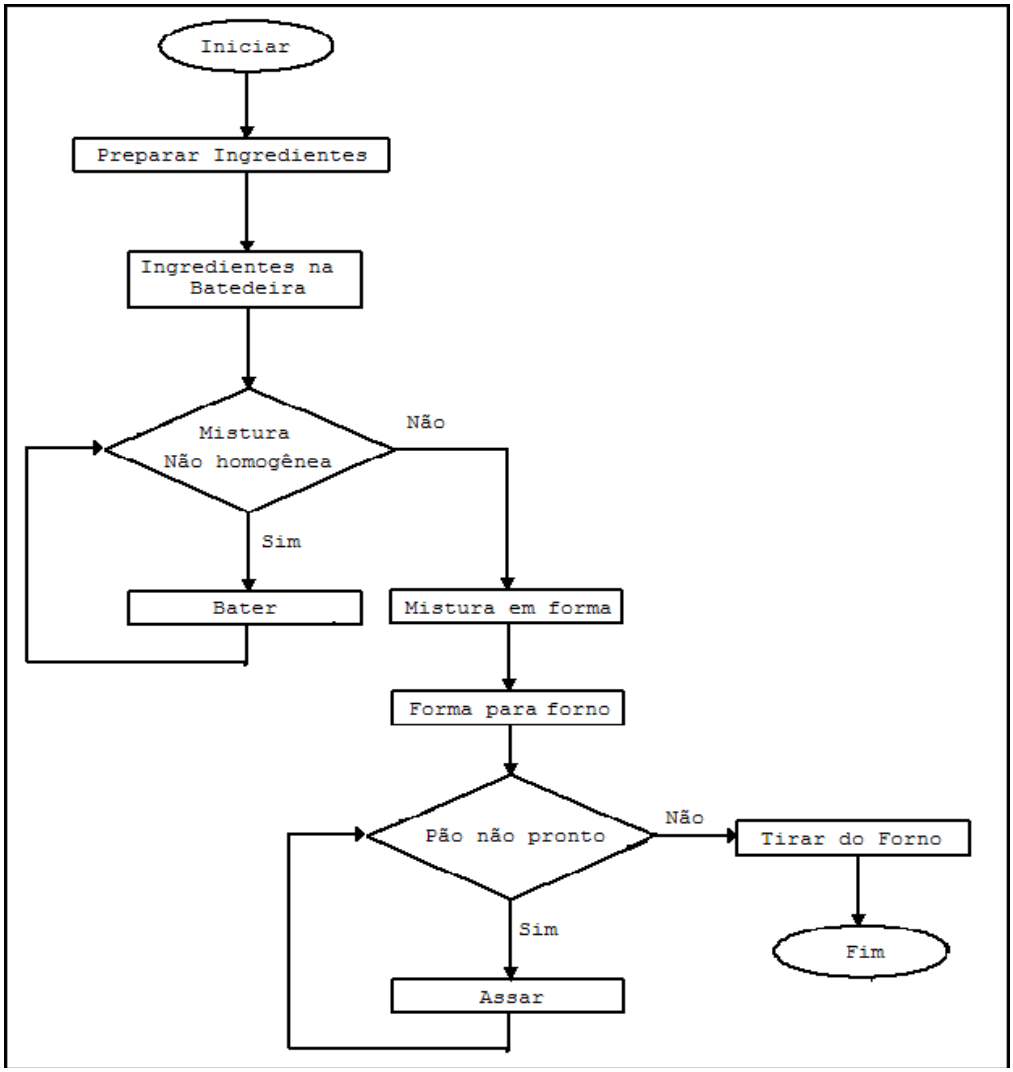
Exercício 1. Escrevendo Algoritmos

1. Assar Pão

Pseudo código:

```
Preparar todos os ingredientes
Colocar os ingredientes na batedeira
Enquanto a mistura não estiver homogênea
    Bater ingredientes
Colocar a mistura em uma forma de pão
Inserir a forma no forno
Enquanto pão não estiver pronto
    Esperar
Retirar a forma do forno
```

Fluxograma:

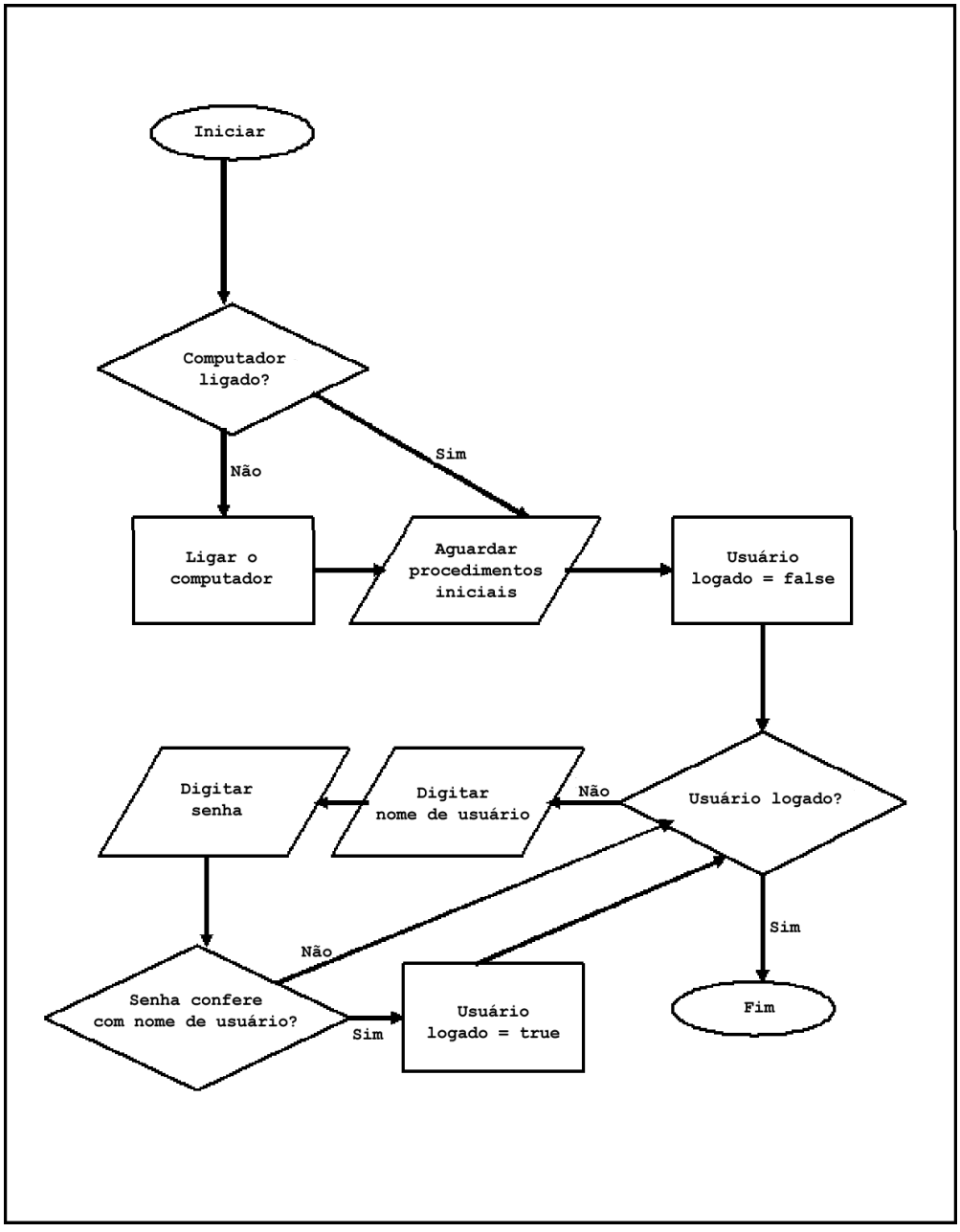


2. Acessar o computador

Pseudo código:

```
Fazer power = botão ligar do computador
Fazer in = status do usuário (inicialmente falso)
Se power == off
    Pressione o botão ligar
Aguardar o procedimento inicial
Enquanto in == falso
    Entrar com o nome do usuário
    Entrar com a senha
    Se senha e nome do usuário são corretos
        in = verdadeiro
```

Fluxograma:

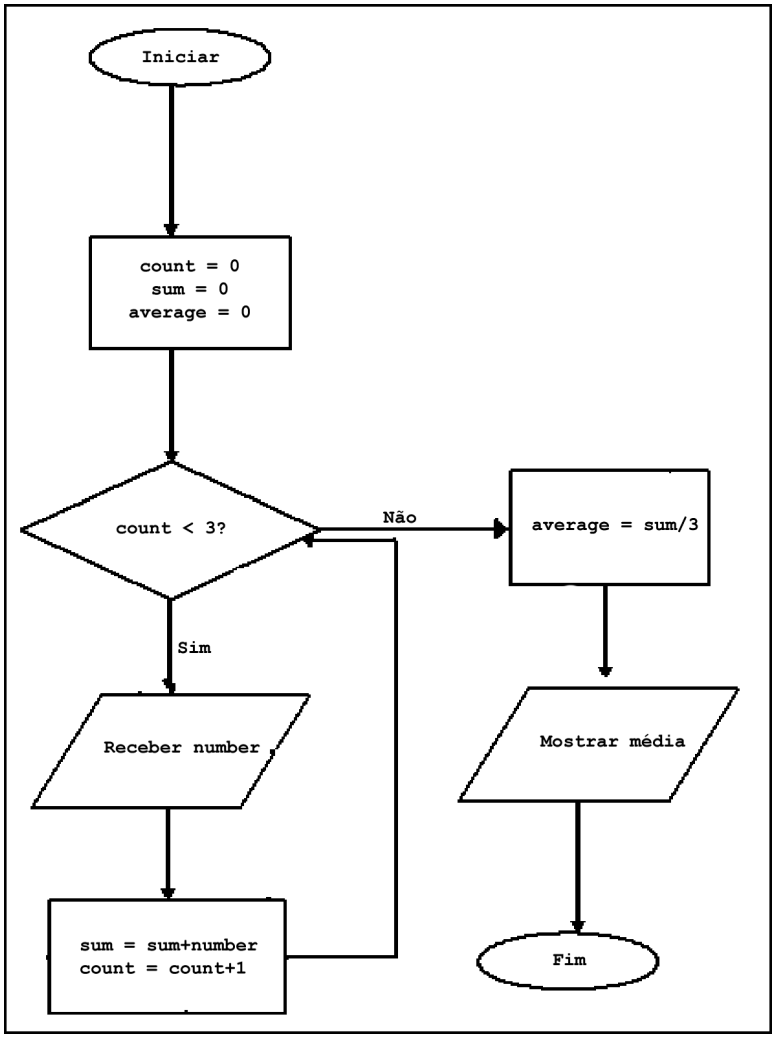


3. Obter a média de três números

Pseudo código:

```
Fazer count = 0
Fazer sum = 0
Fazer average = 0
Enquanto count < 3
   Obter number
   Fazer sum = sum + number
   Fazer count = count + 1
Fazer average = sum / 3
Mostrar average
```

Fluxograma:



Exercício 2. Conversões Numéricas

1. 1980_{10} para binário, hexadecimal e octal

Para Binário:

```
1980/2 = 990          0
990/2  = 495          0
495/2  = 247          1
```

```
1980/2 = 990          0
247/2  = 123          1
123/2  = 61           1
61/2   = 30           1
30/2   = 15           0
15/2   = 7            1
7/2    = 3            1
3/2    = 1            1
1/2    = 0            1
```

Binário = `11110111100`

Para Hexadecimal:

```
0111        1011         1100
  7           B            C
```

Hexadecimal = `7BC`

Para Octal:

```
011       110       111      100
 3         6         7        4
```

Octal = `3674`

2. 1001001101_2 para decimal, hexadecimal e octal

Para Decimal:

```
1 * 1   =   1
0 * 2   =   0
1 * 4   =   4
1 * 8   =   8
0 * 16  =   0
0 * 32  =   0
1 * 64  =  64
0 * 128 =   0
0 * 256 =   0
1 * 512 = 512
TOTAL   = 589
```

Decimal = `589`

Para Hexadecimal:

```
0010         0100         1101
 2            4            D
```

Hexadecimal = `24D`

Para Octal:

```
001       001       001      101
 1         1         1        5
```

Octal = `1115`

3. 76_8 para binário, hexadecimal e decimal

Para Binário:

```
111                 110
 7                   6
```

Binário = 111110

Para Hexadecimal:

```
  0011        1110
    3           E
```

Hexadecimal = 3E

Para Decimal:

```
6 * 1 =  6
7 * 8 = 56
TOTAL = 62
```

Decimal = 62

4. $43F_{16}$ para binário, decimal e octal

Para Binário:

```
   4          3          F
 0100       0011       1111
```

Binário = 010000111111

Para Decimal:

```
F * 1   =   15
3 * 16  =   48
4 * 256 = 1024
TOTAL   = 1087
```

Decimal = 1087

Para Octal:

```
 010       000       111       111
   2         0         7         7
```

Octal = 2077

Lição 2 – Histórico de Java

Não houve exercícios nesta lição

Lição 3 – Primeiros passos no ambiente de programação

Exercício 1. Melhorando o Hello World

```
/**
 * Esta classe mostra "Welcome to Java Programming [SeuNome]!!!"
 */
public class HelloWorld {
    public static void main(String[] args){
        System.out.println("Welcome to Java Programming [YourName]!!!");
    }
}
```

Exercício 2. A árvore

```
/**
 * O programa mostra o poema "The Tree"
 */
public class TheTree {
    public static void main(String[] args){
        System.out.println("I think I shall never see,");
        System.out.println("[Eu acho que nunca verei,]");
        System.out.println("a poem as lovely as a tree.");
        System.out.println("[um poema tão adorável quanto uma árvore.]");
        System.out.println("A tree whose hungry mouth is pressed");
        System.out.println("[Uma árvore cuja boca faminta é pressionada]");
        System.out.println("Against the Earth's flowing breast.");
        System.out.println("[Contra a Terra fluindo em seu seio docemente.]");
    }
}
```

Lição 4 – Fundamentos da programação

Exercício 1. Declarar e mostrar variáveis

```
/**
 * Um programa que declara diferentes variáveis
 * e mostra os valores dessas variáveis
 */
public class VariableSample {
    public static void main(String[] args){
        // Declarar variável number do tipo integer
        // com valor inicial igual a 10
        int number = 10;

        // Declarar variável letter do tipo character
        // com valor inicial igual a 'a'
        char letter = 'a';

        // Declara variável result do tipo boolean
        // com valor inicial igual a true
        boolean result = true;

        // Declarar variável str do tipo String
        // com valor inicial igual a "hello"
        String str = "hello";

        // Mostrar os valores das variáveis
        System.out.println("Number = " + number);
        System.out.println("letter = " + letter);
        System.out.println("result = " + result);
        System.out.println("str    = " + str);
    }
}
```

Exercício 2. Obter a média de três números

```
/**
 * Um programa que calcula a média de três
 * números: 10,20, e 45
 * e imprime o resultado na tela
 */
public class AverageNumber {
    public static void main(String[] args){
        // Declarar três números
        int num1 = 10;
        int num2 = 20;
        int num3 = 45;
        // Obter a média dos números e guardar na variável ave
        int ave = (num1+num2+num3)/3;
        // Mostrar
        System.out.println("number 1 = " + num1);
        System.out.println("number 2 = " + num2);
        System.out.println("number 3 = " + num3);
        System.out.println("Average is = " + ave);
    }
}
```

Exercício 3. Exibir o maior valor

```
/**
 * Um programa que imprime o número com
 * maior valor dados três números
 */
public class GreatestValue {
    public static void main(String[] args){
```

```
        // Declarar os números
        int num1 = 10;
        int num2 = 23;
        int num3 = 5;
        int max = 0;
        // Determinar qual é o maior
        max = (num1>num2)?num1:num2;
        max = (max>num3)?max:num3;
        // Mostrar
        System.out.println("número 1 = " + num1);
        System.out.println("número 2 = " + num2);
        System.out.println("número 3 = " + num3);
        System.out.println("O maior número é = " + max);
    }
}
```

Exercício 4. Precedência de Operadores

1. (((a/b)^c)^((d-e+f-(g*h))+i))
2. ((((((3*10)*2)/15)-2+4)^2)^2)
3. ((r^((((s*t)/u)-v)+w))^(x-(y++)))

Lição 5 – Capturando entrada de dados através do teclado

Exercício 1. As 3 palavras (versão Console - BufferedReader)

```
import java.io.*;
/**
 * Um programa que solicita três palavras ao usuário
 * e então as imprime na tela como uma frase
 */
public class LastThreeWords {
    public static void main(String[] args){
        // Declarar a variável reader como
        BufferedReader reader = new BufferedReader(
            new InputStreamReader(System.in));
        // Declarar variáveis String para as 3 palavras
        String firstWord = "";
        String secondWord = "";
        String thirdWord = "";
        try{
            System.out.print("Palavra 1: ");
            // Obter a primeira palavra
            firstWord = reader.readLine();
            // Obter a segunda palavra
            System.out.print("Palavra 2: ");
            secondWord = reader.readLine();
            // Obter a terceira palavra
            System.out.print("Palavra 3: ");
            thirdWord = reader.readLine();
        } catch(IOException e) {
            System.out.println("Erro na obtenção dos dados");
        }
        // Mostrar a frase
        System.out.println(firstWord + " " + secondWord + " " + thirdWord);
     }
}
```

Exercício 1. As 3 palavras (versão Console - Scanner)

import java.util.Scanner;

```
/**
 * Um programa que solicita três palavras ao usuário
 * e então as imprime na tela como uma frase
 */
public class LastThreeWords {
    public static void main(String[] args){
        // Declarar a variável  Scanner
        Scanner sc = new Scanner(System.in));
        // Declarar variáveis String para as 3 palavras
        String firstWord = "";
        String secondWord = "";
        String thirdWord = "";
        System.out.print("Palavra 1: ");
        // Obter a primeira palavra
        firstWord = sc.nextLine();
        // Obter a segunda palavra
        System.out.print("Palavra 2: ");
        secondWord = sc.nextLine();
        // Obter a terceira palavra
        System.out.print("Palavra 3: ");
        thirdWord = sc.nextLine();
        // Mostrar a frase
        System.out.println(
```

```
            firstWord + " " + secondWord + " " + thirdWord);
    }
}
```

Exercício 2. Últimas 3 palavras (versão Gráfica)

```
import javax.swing.JOptionPane;
/**
 * Um programa que solicita do usuário três palavras usando
 * o  JOptionPane e mostra na tela essas três palavras como
 * uma frase
 */
public class LastThreeWords {
    public static void main(String[] args){
        // Obter a primeira palavra do usuário
        String firstWord = JOptionPane.showInputDialog("Palavra 1");
        // Obter a segunda palavra do usuário
        String secondWord = JOptionPane.showInputDialog("Palavra 2");
        // Obter a terceira palavra do usuário
        String thirdWord = JOptionPane.showInputDialog("Palavra 3");
        // mostra a mensagem
        JOptionPane.showMessageDialog(null,
            firstWord + " " + secondWord + " " + thirdWord);
    }
}
```

Lição 6 – Estruturas de controle

Exercício 1. Notas

1. Utilizando BufferedReader:

```
import java.io.*;
/**
 * Obtém três entradas numéricas do usuário
 * e mostra a média na tela
 */
public class Grades {
    public static void main(String[] args) {
        // Declarar a variável reader como entrada
        BufferedReader reader = new BufferedReader
            (new InputStreamReader(System.in));
        int firstGrade = 0;
        int secondGrade = 0;
        int thirdGrade = 0;
        double average = 0;
        try {
            System.out.print("Primeira nota: ");
            firstGrade = Integer.parseInt(reader.readLine());
            System.out.print("Segunda nota: ");
            secondGrade = Integer.parseInt(reader.readLine());
            System.out.print("Terceira nota: ");
            thirdGrade = Integer.parseInt(reader.readLine());
        } catch(Exception e) {
            System.out.println("Entrada inválida");
            System.exit(0);
        }
        // Calcular a média
        average = (firstGrade+secondGrade+thirdGrade)/3;
        // Mostrar a média dos três exames
        System.out.print("Média: "+average);
        if (average >= 60)
            System.out.print(" :-)");
        else
            System.out.print(" :-(");
    }
}
```

2. Utilizando JOptionPane:

```
import javax.swing.JOptionPane;
/**
 * Obtém três entradas numéricas do usuário
 * e mostra a média na tela
 */
public class Grades {
    public static void main(String[] args) {
        double firstGrade = 0;
        double secondGrade = 0;
        double thirdGrade = 0;
        double average = 0;
        try {
            firstGrade = Double.parseDouble(JOptionPane.showInputDialog
                ("Primeira nota"));
            secondGrade = Double.parseDouble(JOptionPane.showInputDialog
                ("Segunda nota"));
            thirdGrade = Double.parseDouble(JOptionPane.showInputDialog
                ("Terceira nota"));
        } catch( Exception e) {
            JOptionPane.showMessageDialog(null, "Entrada inválida");
            System.exit(0);
```

```
        }
        // Calcular a média
        average = (firstGrade+secondGrade+thirdGrade)/3;
        if (average>=60)
            JOptionPane.showMessageDialog(null,"Média :"+average+" :-)");
        else
            JOptionPane.showMessageDialog(null,"Média :"+average+" :-(");
    }
}
```

Exercício 2. Números por Extenso

Classe Básica

```
import javax.swing.JOptionPane;
/**
 * Transforma uma entrada numérica entre 1-10 para palavras
 * utilizando if-else
 */
public class NumWords {
    public static void main(String[] args) {
        String msg = "";
        int input = 0;
        // Obter a entrada
        input = Integer.parseInt(JOptionPane.showInputDialog("Digite número"));
        // -------------------------------------
        // Substitua aqui as declarações
        // -------------------------------------
        // Mostrar o número em palavras se dentro da faixa
        JOptionPane.showMessageDialog(null,msg);
    }
}
```

1. Declaração if-else:

```
// Declaração if para atribuir em msg o extenso do número digitado
if (input == 1) msg = "um";
else if(input == 2) msg = "dois";
else if(input == 3) msg = "três";
else if(input == 4) msg = "quatro";
else if(input == 5) msg = "cinco";
else if(input == 6) msg = "seis";
else if(input == 7) msg = "sete";
else if(input == 8) msg = "oito";
else if(input == 9) msg = "nove";
else if(input == 10) msg = "dez";
else msg = "número inválido";
```

2. Declaração switch:

```
// Declaração switch para atribuir em msg o extenso do número digitado
switch (input) {
    case 1: msg = "um"; break;
    case 2: msg = "dois"; break;
    case 3: msg = "três"; break;
    case 4: msg = "quatro"; break;
    case 5: msg = "cinco"; break;
    case 6: msg = "seis"; break;
    case 7: msg = "sete"; break;
    case 8: msg = "oito"; break;
    case 9: msg = "nove"; break;
    case 10: msg = "dez"; break;
    default: msg = "número inválido";
}
```

Exercício 3. Cem Vezes

Classe Básica

```
import java.io.*;
/**
 * Um programa que imprime 100 vezes um número digitado
 */
public class HundredNames{
    public static void main(String[] args){
        BufferedReader reader = new BufferedReader(
            new InputStreamReader(System.in));
        String name = "";
        // Obter o nome do usuário
        try {
            System.out.print("Digite o nome: ");
            name = reader.readLine();
        } catch(Exception e) {
            System.out.println("entrada inválida");
            System.exit(0);
        }
        // ------------------------------------
        // Substitua aqui as declarações
        // ------------------------------------
    }
}
```

1. Declaração while:

```
// Declaração while para exibir 100 vezes o nome digitado
int counter = 0;
while (counter < 100) {
    System.out.println(name);
    counter++;
}
```

2. Declaração do-while:

```
// Declaração do-while para exibir 100 vezes o nome digitado
int counter = 0;
do {
    System.out.println(name);
    counter++;
} while(counter < 100);
```

3. Declaração for:

```
// Declaração for para exibir 100 vezes o nome digitado
for (int counter = 0; counter < 100; counter++) {
    System.out.println(name);
}
```

Exercício 4. Potências

Classe Básica

```
import javax.swing.JOptionPane;
/**
 * Calcula a potência de um número dados a base e expoente.
 * O expoente está limitado a números positivos.
 */
public class Powers {
    public static void main(String[] args){
        int base = 0;
        int exp = 0;
        int power = 1;
```

```
        // Obter entrada do usuário para base e expoente
        base = Integer.parseInt(JOptionPane.showInputDialog("Base"));
        exp = Integer.parseInt(JOptionPane.showInputDialog("Expoente"));
        // Limitar variável exp a somente número positivos e maiores que 0
        if (exp <= 0) {
            JOptionPane.showMessageDialog(null,
                "Somente números positivos e maiores que 0, por favor");
            System.exit(0);
        }
        // ------------------------------------
        // Substitua aqui as declarações
        // ------------------------------------
        // Mostrar o resultado
        JOptionPane.showMessageDialog(null,
            base + " elevado a " + exp + " é igual a " + power);
    }
}
```

1. Declaração while:

```
// Declaração while para calcular a potência
int counter = 0;
while (counter++ < exp)
    power = power*base;
```

2. Declaração do-while:

```
// Declaração do-while para calcular a potência
int counter = 0;
do
    power = power*base;
while(++counter < exp);
```

3. Declaração for:

```
// Declaração for para calcular a potência
for (int counter = 0; counter < exp; counter++)
    power = power*base;
```

Lição 7 – Array em Java

Exercício 1. Dias da Semana

```
/**
 * Utilizar um array de string para salvar os dias da semana
 * e imprime na tela.
 */
public class DaysOfTheWeek {
    public static void main(String[] args){
        // Declarar o array de String dos dias da semana
        String[] days = {"Sunday", "Monday", "Tuesday",
                         "Wednesday", "Thursday", "Friday", "Saturday"};
        // ------------------------------------
        // Substitua aqui as declarações
        // ------------------------------------
    }
}
```

1. Declaração while:

```
        // Declaração while para mostrar os dias da semana
        int counter = 0;
        while (counter++ < days.length)
            System.out.println(days[counter]);
```

2. Declaração do-while:

```
        // Declaração do-while para mostrar os dias da semana
        int counter = 0;
        do
            System.out.println(days[counter++]);
        while(counter < days.length);
```

3. Declaração for:

```
        // Declaração for para mostrar os dias da semana
        for (int counter = 0; counter < days.length; counter++)
            System.out.println(days[counter]);
```

Exercício 2. Maior número

```
import javax.swing.JOptionPane;
/**
 * Um programa que utiliza JOptionPane para obter dez números do usuário
 * e exibir o maior número.
 */
public class GreatestNumber {
    public static void main(String[] args){
        int[] num = new int[10];
        int counter;
        int max = 0;
        // Declaração for para obter 10 números do usuário
        for (counter = 0; counter < 10; counter++) {
            num[counter] = Integer.parseInt(
                JOptionPane.showInputDialog("Digite o número "+(counter+1)));
            // Obter o número máximo
            if ((counter == 0)||(num[counter] > max))
                max = num[counter];
        }
        // Mostrar o maior número
        JOptionPane.showMessageDialog(null, "O número com o maior valor é " + max);
    }
}
```

Exercício 3. Entradas de Agenda Telefônica

```
import javax.swing.JOptionPane;
/**
 * Um programa que mostra os dados de um array multidimensional.
 */
public class GreatestNumber {
    public static void main(String[] args){
        String entry [][] = {
            {"Florence", "735-1234", "Manila"},
            {"Joyce", "983-3333", "Quezon City"},
            {"Becca", "456-3322", "Manila"}};
        // Declaração for para percorrer nas linhas
        for (int line = 0; line < entry.length; line++) {
            System.out.println("Name   : " + entry[line][0]);
            System.out.println("Tel. # : " + entry[line][1]);
            System.out.println("Address: " + entry[line][2]);
            System.out.println();
        }
    }
}
```

Lição 8 – Argumentos de linha de comando

Exercício 1. Argumentos de Exibição

```
/**
 * Um programa que imprime a string da linha de comando, se
 * houver.
 */
public class CommandLineSample {
    public static void main(String[] args){
        // Declaração for para exibir os argumentos da linha de comando
        for (int counter=0; counter < args.length; counter++)
            System.out.println(args[counter]);
    }
}
```

Exercício 2. Operações Aritméticas

```
/**
 *
 */
public class ArithmeticOperation {
    public static void main(String[] args){
        // Obter os argumentos em int
        int num1 = Integer.parseInt(args[0]);
        int num2 = Integer.parseInt(args[1]);
        // Mostrando valores
        System.out.println("sum = " + (num1 + num2));
        System.out.println("subtraction = " + (num1 - num2));
        System.out.println("multiplication = " + (num1 * num2));
        System.out.println("division = " + (num1 / num2));
    }
}
```

Lição 9 – Trabalhando com bibliotecas de classes

Exercício 1. Argumentos de Exibição

Não existe uma única resposta.

Exercício 2. Java Scavenger Hunt

Nota: Esses são apenas **alguns exemplos de métodos** na API Java que é possível utilizar. Verifique a API Java para mais respostas.

```
public class Homework1 {
    public static void main(String []args) {
        // 1. endsWith
        String str = "Hello";
        System.out.println( str.endsWith( "slo" ) );

        //2. forDigit
        System.out.println( Character.forDigit(13, 16) );

        //3. floor
        System.out.println( Math.floor(3.14));

        //4. isDigit
        System.out.println( "0=" + Character.isDigit('0'));
        System.out.println( "A=" +Character.isDigit('A'));

        //5. exit
        System.exit(1);
        System.out.println("if this is executed, exit was not called");
    }
}
```

Declaração de Classe e Método:

```
1. Classe: String
Método: public boolean endsWith(String suffix)

2. Classe: Character
Método: public static char forDigit(int digit, int radix)

3. Classe: Math
Método: public static double floor(double a)

4. Classe: Math
Método: public static boolean isDigit(char ch)

5. Classe: System
Método: public static void exit(int status)
```

Lição 10 – Criando nossas classes

Exercício 1. Registro de agenda

```
/**
 * Uma classe de registro de agenda que armazena o nome da
 * pessoa, endereço, número de telefone e endereço de email
 */
public class AddressBookEntry {
    private String name;
    private String address;
    private String tel;
    private String email;

    /**
     * construtor padrão
     */
    public AddressBookEntry(){
        this.setName("");
        this.setAddress("");
        this.setTel("");
        this.setEmail("");
    }
    /**
     * Cria um objeto AddressBookEntry com o nome, endereço,
     * número de telefone e endereço de email fornecidos.
     */
    public AddressBookEntry(String name, String address, String tel, String email) {
        this.setName(name);
        this.setAddress(address);
        this.setTel(tel);
        this.setEmail(email);
    }
    /**
     * retorna a variável name
     */
    public String getName(){
        return name;
    }
    /**
     * altera a variável name
     */
    public void setName(String name){
        this.name = name;
    }
    /**
     * retorna a variável address
     */
    public String getAddress(){
        return address;
    }
    /**
     * altera a variável address
     */
    public void setAddress(String address){
        this.address = address;
    }
    /**
     * retorna a variável tel
     */
    public String getTel(){
        return tel;
    }
    /**
     * altera a variável tel
```

```
     */
    public void setTel(String tel){
        this.tel = tel;
    }
    /**
     * retorna a variável email
     */
    public String getEmail(){
        return email;
    }
    /**
     * altera a variável email
     */
    public void setEmail(String email){
        this.email = email;
    }
}
```

Exercício 2. Agenda

```
import java.io.*;
/**
 * Cria uma agenda com 100 registros AddressBookEntries
 */
public class AddressBook {
    // Índice do último registro
    private int top = 0;
    // Número constante que indica o número máximo de registros da agenda
    private static final int MAXENTRIES = 100;
    // Array de registros da agenda
    private AddressBookEntry[] list;

    /**
     * Método principal
     */
    public static void main(String[] args) {
        BufferedReader keyIn = new BufferedReader
            (new InputStreamReader(System.in));
        AddressBook addBook = new AddressBook();
        String act = "";
        while(true) {
            // Mostrar as opções
            System.out.println("\n[A] Adicionar registro");
            System.out.println("[E] Excluir registro");
            System.out.println("[V] Visualizar registros");
            System.out.println("[U] atUalizar registro");
            System.out.println("[S] Sair do projeto");
            System.out.print("Digite a ação desejada: ");
            try {
                // Obter a escolha
                act = keyIn.readLine();
            } catch(Exception e) {
                System.out.println("Erro");
            }
            // Verificar a ação apropiada para a escolha do usuário
            switch (act.charAt(0)) {
                case 'A': case 'a': addBook.addEntry(); break;
                case 'E': case 'e': addBook.delEntry(); break;
                case 'V': case 'v': addBook.viewEntries(); break;
                case 'U': case 'u': addBook.updateEntry(); break;
                case 'S': case 's': System.exit(0);
                default: System.out.println("Comando Desconhecido");
            }
        }
    }
    /**
```

```
     * cria a agenda
     */
    public AddressBook(){
        list = new AddressBookEntry[MAXENTRIES];
    }
    /**
     * método para adicionar um registro AdressBookEntry
     * na agenda
     */
    public void addEntry(){
        BufferedReader keyIn = new BufferedReader(new InputStreamReader(System.in));
        String name = "";
        String add = "";
        String tel = "";
        String email = "";
        if (top == MAXENTRIES) {
            System.out.println("Agenda está cheia");
            return;
        }
        //Pede ao usuário a digitação dos dados
        try {
            System.out.print("Nome: ");
            name = keyIn.readLine();
            System.out.print("Endereço: ");
            add = keyIn.readLine();
            System.out.print("Telefone: ");
            tel = keyIn.readLine();
            System.out.print("Email: ");
            email = keyIn.readLine();
        } catch(Exception e) {
            System.out.println(e);
            System.exit(0);
        }
        AddressBookEntry entry = new AddressBookEntry(name, add, tel, email);
        list[top] = entry;
        top++;
    }
    /**
     * método que deleta um registro AddressBookEntry
     * da agenda com o índice
     */
    public void delEntry(){
        BufferedReader keyIn = new BufferedReader(new InputStreamReader(System.in));
        int index = 0;
        // Verificar se a agenda está vazia
        if (top == 0) {
            System.out.println("Agenda está vazia");
            return;
        }
        // Solicitar o registro a ser deletado
        try {
            // Exibir os registros existentes na agenda
            viewEntries();
            System.out.print("\nDigite número do registro: ");
            index = Integer.parseInt(keyIn.readLine())-1;
        }catch(Exception e){
        }
        //verifica se o índice está dentro do limites
        if (index < 0 || index >= top) {
            System.out.println("Índice fora dos limites");
            return;
        } else {
            for (int i=index; i<top; i++)
                list[i] = list[i+1];
            list[top] = null;
            top--;
```

```
        }
    }
    /**
     * método que imprime todos os registros da agenda
     */
    public void viewEntries(){
        for (int index = 0; index < top; index++) {
            System.out.println((index+1)+" Nome:"+list[index].getName());
            System.out.println("Endereço:"+list[index].getAddress());
            System.out.println("Telefone:"+list[index].getTel());
            System.out.println("Email:"+list[index].getEmail());
        }
    }
    /**
     * método que atualiza um registro
     */
    public void updateEntry(){
        BufferedReader keyIn = new BufferedReader(new InputStreamReader(System.in));
        int index = 0;
        String name = "";
        String add = "";
        String tel = "";
        String email = "";
        // Solicitar a digitação dos dados
        try {
            System.out.print("Número do registro: ");
            index =Integer.parseInt(keyIn.readLine())-1;
            System.out.print("Nome: ");
            name = keyIn.readLine();
            System.out.print("Endereço: ");
            add = keyIn.readLine();
            System.out.print("Telefone: ");
            tel = keyIn.readLine();
            System.out.print("Email: ");
            email = keyIn.readLine();
        } catch(Exception e) {
            System.out.println(e);
            System.exit(0);
        }
        // Atualizar o registro
        AddressBookEntry entry = new AddressBookEntry(name, add, tel, email);
        list[index] = entry;
    }
}
```

Lição 11 – Herança, polimorfismo e interfaces

Exercício 1. Estendendo StudentRecord

```
/**
 * Um objeto que armazena os dados de um estudante
 */
public class StudentRecord {
    private String name;
    private String address;
    private int    age;
    private double mathGrade;
    private double englishGrade;
    private double scienceGrade;

    protected static int studentCount;

    /**
     * Retorna o nome do estudante
     */
    public String getName() {
        return name;
    }
    /**
     * Altera o nome do estudante
     */
    public void setName(String temp) {
        name = temp;
    }
    /**
     * Retorna o endereço do estudante
     */
    public String getAddress() {
        return address;
    }
    /**
     * Altera o endereço do estudante
     */
    public void setAddress(String temp) {
        address = temp;
    }
    /**
     * Retorna a idade do estudante
     */
    public int getAge() {
        return age;
    }
    /**
     * Altera a idade do estudante
     */
    public void setAge(int temp) {
        age = temp;
    }
    /**
     * Retorna a nota de inglês do estudante
     */
    public double getEnglishGrade() {
        return englishGrade;
    }
    /**
     * Altera a nota de inglês do estudante
     */
    public void setEnglishGrade(double temp) {
        englishGrade = temp;
```

```
    }
    /**
     * Retorna a nota de matemática do estudante
     */
    public double getMathGrade() {
        return mathGrade;
    }
    /**
     * Altera a nota de matemática do estudante
     */
    public void setMathGrade(double temp) {
        mathGrade = temp;
    }
    /**
     * Retorna a nota de ciências do estudante
     */
    public double getScienceGrade() {
        return scienceGrade;
    }
    /**
     * Altera a nota de ciências do estudante
     */
    public void setScienceGrade(double temp) {
        scienceGrade = temp;
    }
    /**
     * Calcula a média das notas de inglês, matemática e
     * ciências do estudante
     */
    public double getAverage() {
        return (mathGrade+englishGrade+scienceGrade)/3;
    }
    /**
     * Retorna o número de instâncias de StudentRecords
     */
    public static int getStudentCount() {
        return studentCount;
    }
}

/**
 * Um registro de estudante de um estudante de Ciência de
 * Computação
 */
public class ComputerScienceStudentRecord extends StudentRecord {
    private String studentNumber;
    private double comSciGrade;

    /**
     * Retorna a matrícula do estudante
     */
    public String getStudentNumber() {
        return studentNumber;
    }
    /**
     * Altera a matrícula do estudante
     */
    public void setStudentNumber(String temp) {
        studentNumber = temp;
    }
    /**
     * Retorna a nota de ciência de computação do estudante
     */
    public double getComSciGrade() {
        return comSciGrade;
    }
```

```
    /**
     * Altera a nota de ciência de computação do estudante
     */
    public void setComSciGrade(double temp) {
        comSciGrade = temp;
    }
}
```

Exercício 2. Classes Abstratas

```
/**
 * Definição da classe abstrata forma
 */
public abstract class Shape {
    /**
     * retorna a área de determinada forma
     */
    public abstract double getArea();

    /**
     * retorna o nome da forma
     */
    public abstract String getName();
}

/**
 * Definição de Classe para objeto círculo
 */
public class Circle extends Shape {
    private static final double pi = 3.1416;
    private double radius = 0;
    /**
     * Construtor
     */
    public Circle(double r) {
        setRadius(r);
    }
    /**
     * retorna area
     */
    public double getArea() {
        return pi*radius*radius;
    }
    /**
     * retorna o nome da forma
     */
    public String getName() {
        return "circle";
    }
    /**
     * Atribui raio
     */
    public void setRadius(double r) {
        radius = r;
    }
    /**
     * retorna raio
     */
    public double getRadius() {
        return radius;
    }
}
/**
 * Definição de Classe para objeto quadrado
 */
public class Square extends Shape {
```

```
    private double side = 0;

    /**
     * Construtor
     */
    public Square(double s) {
        setSide( s );
    }
    /**
     * retorna area
     */
    public double getArea() {
        return side*side;
    }
    /**
     * retorna o nome da forma
     */
    public String getName() {
        return "square";
    }
    /**
     * atribui tamanho do lado
     */
    public void setSide(double s) {
        side = s;
    }
    /**
     * retorna o tamanho de um lado
     */
    public double getSide(){
        return side;
    }
}
```

Lição 12 – Tratamento básico de exceções

Exercício 1. Capturando exceções 1

```
public class TestExceptions {
    public static void main(String[] args) {
        try {
            for (int i=0; true; i++)
                System.out.println("args["+i+"]="+args[i]);
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Exception caught: " + e);
        }
        System.out.println("Quiting...");
    }
}
```

Exercício 2. Capturando exceções 2

Não existe uma única resposta.

Parceiros que tornaram JEDI™ possível

Instituto CTS
Patrocinador do DFJUG.

Sun Microsystems
Fornecimento de servidor de dados para o armazenamento dos vídeo-aulas.

Java Research and Development Center da Universidade das Filipinas
Criador da Iniciativa JEDI™.

DFJUG
Detentor dos direitos do JEDI™ nos países de língua portuguesa.

Banco do Brasil
Disponibilização de seus *telecentros* para abrigar e difundir a Iniciativa JEDI™.

Politec
Suporte e apoio financeiro e logístico a todo o processo.

Borland
Apoio internacional para que possamos alcançar os outros países de língua portuguesa.

Instituto Gaudium/CNBB
Fornecimento da sua infra-estrutura de hardware de seus servidores para que os milhares de alunos possam acessar o material do curso simultaneamente.