
**UNIVERSIDADE METODISTA DE SÃO PAULO**

**FACULDADE DE EXATAS E TECNOLOGIA**

**TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS**

**LUCIA HELENA APARECIDA RISSI SALVI**

# INCUBADORA DE EMPRESAS: CALCULADORA DE MATRIZES

**SÃO BERNARDO DO CAMPO**

**2015**

LUCIA HELENA APARECIDA RISSI SALVI

# INCUBADORA DE EMPRESAS: CALCULADORA DE MATRIZES

## TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

## PROJETO DE AÇÃO PROFISSIONAL

## 1º PERÍODO

Atividade realizada pelos alunos do 1º semestre do curso de TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS, modalidade EAD, da Universidade Metodista de São Paulo para atendimento ao Projeto de Ação Profissional - PAP.

**SÃO BERNARDO DO CAMPO**

**2015**

Salvi, Lúcia Helena Aparecida Rissi.
Incubadora de empresas: Calculadora de matrizes. 2015. 92 f.

Projeto de Ação Profissional - PAP. Universidade Metodista de São Paulo. 2015. Osmir Torres.

1. Matrizes. 2. Análise 3. Classes e Objetos. I. Torres, Osmir Alonso Ramires, orient. II. Título.

**LUCIA HELENA APARECIDA RISSI SALVI**

LISIANTHUS Soluções em Tecnologia Ltda
Rua Ada Augusta Byron King, 1815. Jardim Pascal
PABX: 11 LISIANTO (5474-2686) www.lisianthus.com.br

LISIANTHUS

# INCUBADORA DE EMPRESAS: CALCULADORA DE MATRIZES

Atividade realizada pelos alunos do 1º semestre do curso de TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS, modalidade EAD, da Universidade Metodista de São Paulo para atendimento ao Projeto de Ação Profissional - PAP.

**Área de Concentração: Análise e Desenvolvimento de Sistemas**

**Data da Apresentação: 24/05/2015**

**Resultado: _____**

**BANCA EXAMINADORA**

**Camila Santiago** **Prof.** _____
**Universidade Metodista de São Paulo**

**Cristiano Camilo dos Santos de Almeida** **Prof.** _____
**Universidade Metodista de São Paulo**

**Silvia Aparecida Brunini** **Prof.** _____
**Universidade Metodista de São Paulo**

LISIANTHUS

# Resumo

Este documento tem como objetivo o cumprimento da proposta do PAP – Projeto de Ação Profissional, que visa aplicar os conceitos sorvidos no presente curso à um projeto que simula a prática profissional.

O prospecto tema é o case “Incubadora de Empresas”, do Grupo Opportunity que busca potenciais parceiros de trabalho, pelo qual, deve ser apresentado um plano de negócios, dividido em duas partes:

A primeira, é a apresentação da “Lisianthus Soluções em Tecnologia LTDA”, contendo sua logomarca, organograma e histórico.

A segunda etapa é constituída de um projeto técnico de um software que implementa operações com uma ou duas matrizes, contendo a documentação tal qual originou a estrutura dos algoritmos e sua codificação em linguagem de programação (JAVA).

Palavras chave: Matrizes. Algoritmos. Classes. Objetos. Análise.

# Abstract

This document aims to fulfill the PAP 's proposal - Professional Action Project , which aims to apply the concepts sorbed in this way on a project that simulates professional practice.

The issue prospectus is the case " Incubadora de Empresas ", the Grupo Opportunity seeking potential partners to work, for which , a business plan must be submitted in two parts :

The first is the presentation of " Lisianthus Soluções em Tecnologia LTDA ," containing your logo , chart and history.

The second step consists of a technical design of a software that implements operations with one or two matrices containing such documentation which originated the structure of a coding algorithms and programming languages (Java ).

Keywords: Mothers. Algorithms. Classes. Objects. Analysis.

LISIANTHUS

# SUMÁRIO

# 1. INTRODUÇÃO

Este plano de negócios foi desenvolvido a partir da oferta do case “Incubadora de Empresas” do Gruppo Opportunity. É composto por 2(duas) partes, a saber:

Numa primeira abordagem, temos a apresentação da “Lisianthus Soluções em Tecnologia Ltda” e todos os tópicos necessários para o conhecimento e convencimento que é uma empresa séria e comprometida com este projeto.

Na abordagem seguinte, é apresentado o projeto técnico, demonstrando que o sistema produz as demandas especificadas no case, com o detalhamento do produto, seu estudo de viabilidade econômica e todas as informações pertinentes.

# 2. SOBRE A EMPRESA LISIANTHUS

## 2.1. Histórico, localização e ramo de atuação

A Lisianthus nasceu da junção de forças de seus sócios, com o espirito empreendedor e motivados por um segmento do mercado que carece de soluções criativas.

A palavra Lisianthus se originou da aglutinação de letras compositoras dos nomes dos seus sócios. Com nossas vogais e consoantes, formamos um nome; com nossas forças e virtudes, formamos uma empresa que seu maior bem é seu capital humano.

Nossa empresa oferece ao mercado soluções baseadas em tecnologia que trazem satisfação de nossos clientes com produtos inovadores, funcionais e sustentáveis.

A Lisianthus localiza-se na Rua Ada Augusta Byron King, 1815, Jardim Pascal, Cidade, Estado, CEP 09123-456, PABX: 11 LISIANTO (5474-2686).

Nossa empresa atua com o desenvolvimento e licenciamento de softwares personalizados, garantidos pelos serviços de suporte técnico e manutenção, bem como consultoria em tecnologia da informação.

## 2.2. Missão, Visão e Valores

A Lisianthus tem como visão ser referência no mercado de TI, pela excelência e inovação de seus produtos.

Nossa missão é fornecer as melhores e mais criativas soluções em tecnologia da informação.

Nossos valores são ética, excelência e perseverança.

Lisianthus vem do latim, *Lisianto*. É uma flor de beleza delicada, tolerante às condições adversas e duradoura. Assim é a Lisianthus.

# 3. ESTRUTURA DO PROJETO INCUBADORA DE EMPRESAS: CALCULADORA DE MATRIZES

Diante do prospecto apresentado, desenvolvemos uma calculadora de matrizes que o usuário escolha a operação e ordem da matriz; o sistema fará a leitura dos dados das matrizes e exibirá o resultado na janela do console.

## 3.1. Escopo

Em consonância com o case, o sistema abrange a realização de operações com uma ou duas matrizes, quadradas, de ordem máxima 5. Com uma matriz é possível obter o determinante bem como a multiplicação por escalar, ao passo que, com duas é possível realizar a adição, subtração e verificação de igualdade.

## 3.2. A quem se destina

A calculadora de matrizes é um sistema de fácil utilização e pode ser utilizado por qualquer pessoa que necessite de tais resultados.

## 3.3. Principais Funcionalidades

O usuário deve escolher uma dentre as operações disponíveis: (a) Determinante de uma matriz; (b) Multiplicação de uma matriz por escalar; (c) Adição de duas matrizes; (d) Subtração de duas matrizes; (e) Verificação de igualdade de duas matrizes.

Através de entrada (leitura da variável int opcao), o usuário poderá optar por (a) ou (b), o sistema operará com uma matriz; se (c), (d) ou (e), serão operadas duas matrizes.

A próxima interação do sistema com o usuário é requerer que este informe a ordem da matriz ou matrizes, de ordem máxima 5, que serão trabalhadas. Esta informação (int ordem) será fundamental para o chamamento do método.

Cada método tem sua peculiar formato mas, fundamentalmente, será feita a leitura de dados complementares (como o escalar, na multiplicação de matriz por escalar), dos elementos da matriz ou matrizes, o processamento dos devidos cálculos e a apresentação dos resultados na janela do console.

### 3.4. Requisitos Funcionais do Sistema

RF001 – O sistema deve estar em execução e aguardo;
RF002 – O sistema requer informações do usuário;
RF003 – O sistema deve solicitar inserção de dados;
RF004 – O sistema deve armazenar os dados;
RF005 – O sistema deve processar os dados;
RF006 – O sistema deve apresentar o resultado na tela principal.

### 3.5. Características das Matrizes a serem utilizadas

As matrizes utilizadas são de livre inserção de dados, mas devem observar os critérios de validação para cálculo: as matrizes devem ser quadradas bem como de ordem máxima 5. Não são, portanto, admitidas, matrizes linhas ou colunas.

### 3.6. Operações que serão aplicadas entre as Matrizes

São operações aplicáveis:

- Determinante de uma matriz: “Denomina-se assim um número real que se obtém por operações entre os elementos de uma matriz e que oferece informação sobre o caráter regular ou singular da matriz”. ( in Temática Barsa. – Rio de Janeiro: Barsa Planeta, 2006, 9v.:il., pág 168).

- Multiplicação de uma matriz por escalar: “Sejam r um número real e A uma matriz do tipo m x n, o produto de r por A e, por definição, a matriz m x n que

se obter ao multiplicar r por cada um dos elementos de A" (in Enciclopedia Barsa Universal. – Rio de Janeiro: Barsa Planeta, 2010, 11v.: pág 3819).

- Adição de duas matrizes: "Sejam duas matrizes A e B do tipo m x n, a soma dos elementos de A e B é, por definição, a matriz m x n, a qual se obtém somando cada elemento de A com o elemento correspondente de B" (in Enciclopedia Barsa Universal. – Rio de Janeiro: Barsa Planeta, 2010, 11v.: pág 3819).

- Subtração de duas matrizes: cada elemento da primeira matriz será subtraído ao elemento correspondente da segunda matriz.

- Igualdade de duas matrizes: verificação da igualdade dos elementos de uma matriz em comparação aos termos correspondentes da segunda matriz.

# 4. PROGRAMAÇÃO ORIENTADA A OBJETOS – POO

A questão temática deste projeto é a criação de um software que promova 5 (cinco) operações matemáticas envolvendo matrizes definidas no prospecto. A análise da questão gerou um projeto que será implementado, testado e, finalmente, implantado.

Em outras palavras, partir da descrição do problema, encontramos uma solução, e desenvolvemos um projeto a partir de sua documentação, que foi codificado em linguagem de programação.

A codificação deste projeto é em JAVA, que é uma linguagem de programação orientada à objetos. E, portanto, analisamos conceitos que gravitam ao redor deste modelo, tais como: classes, objetos, métodos, herança, polimorfismo, abstração.

De acordo com o projeto, foram criadas uma classe principal (MatrizCalc) e 5 subclasses consignadas a esta, que são as operações: determinante de uma matriz (Determinantes), Multiplicação de uma matriz por escalar (MultEscalar), Adição de duas matrizes (AdicaoMatriz), Subtração de duas matrizes (SubtracaoMatriz) e Verificação de igualdade de duas matrizes (IgualdadeMatriz), conforme “Diagrama de Classes” (fls.24/26)

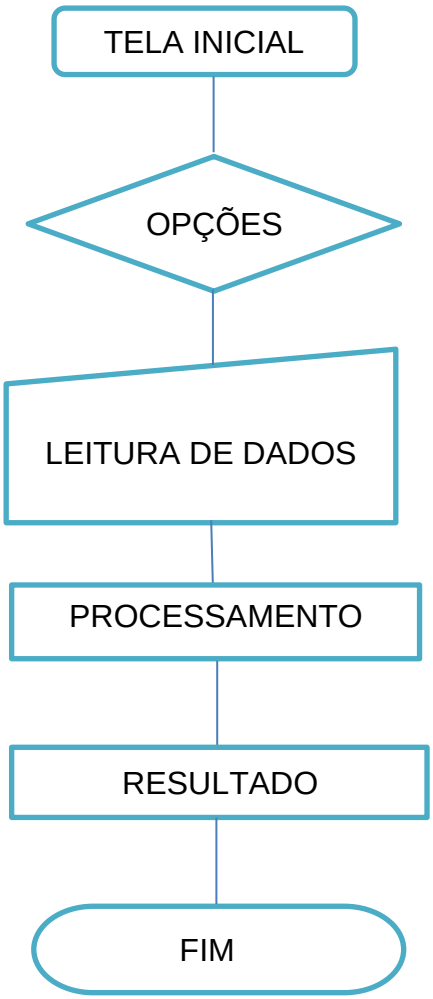
Para a justificativa da escolha, numa analogia: o software possui a estrutura de uma cômoda, onde sua base é a classe principal, que suporta as demais subclasses – as operações unitariamente consideradas. Cada subclasse é como uma gaveta que abarca uma operação.

A implicação principal: possibilita uma interface muito mais limpa e direta, mas também a manutenção compartimentada, uma vez que a função em manutenção poderá ser desabilitada e somente ela, permitindo o uso parcial.

Por fim, o funcionamento está demonstrado através do “Fluxograma” (fls.14/21) e do “Diagrama de Casos de Uso” (fls.22/23), que descrevem sinteticamente as ações do software, e no APÊNDICE E, “Relatório de Resultados”, os resultados dos testes promovidos.

### 4.1 Fluxograma

Apresentamos, neste tópico, o fluxo de dados geral de entrada, processamento e saída que embasa este software.

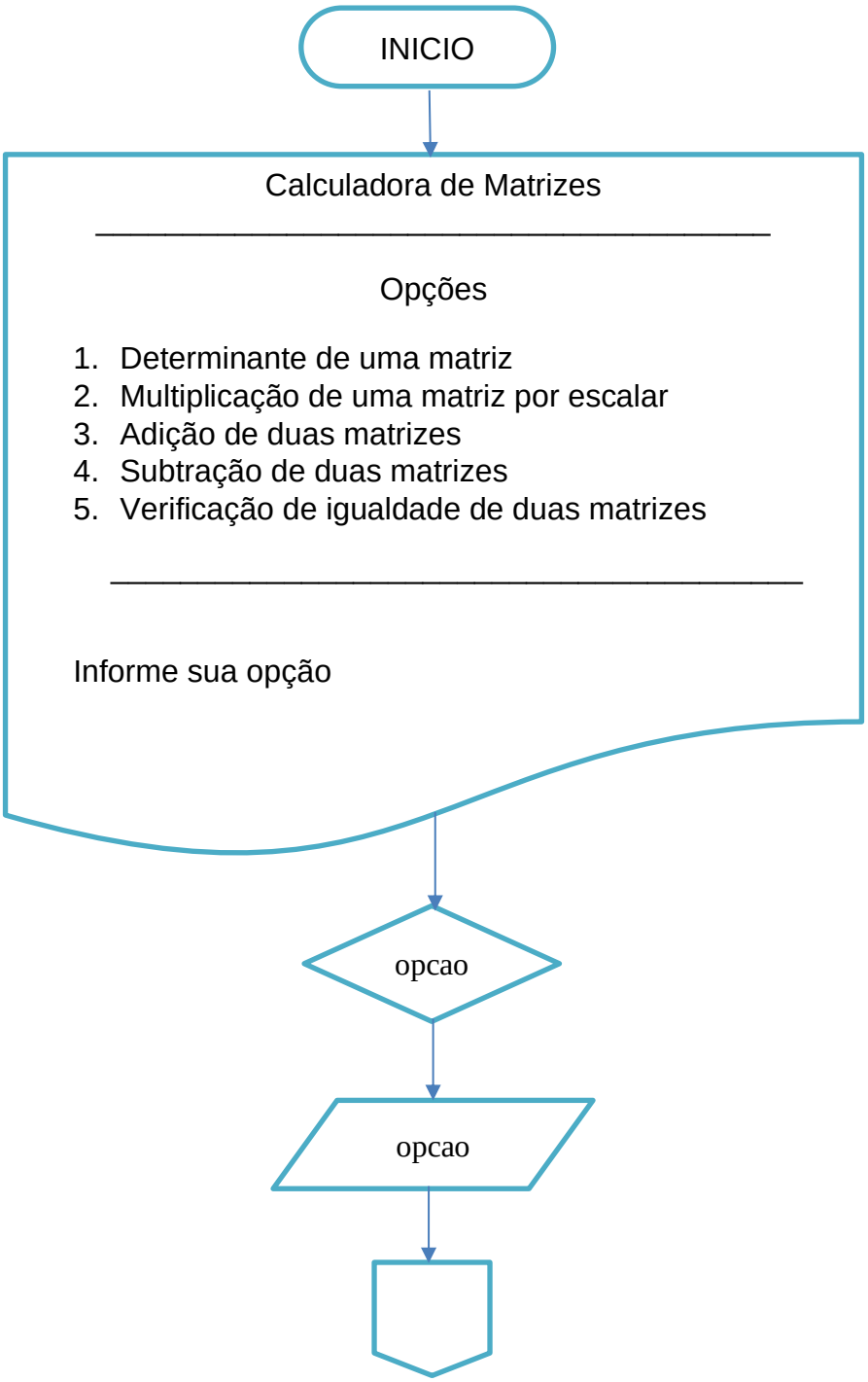


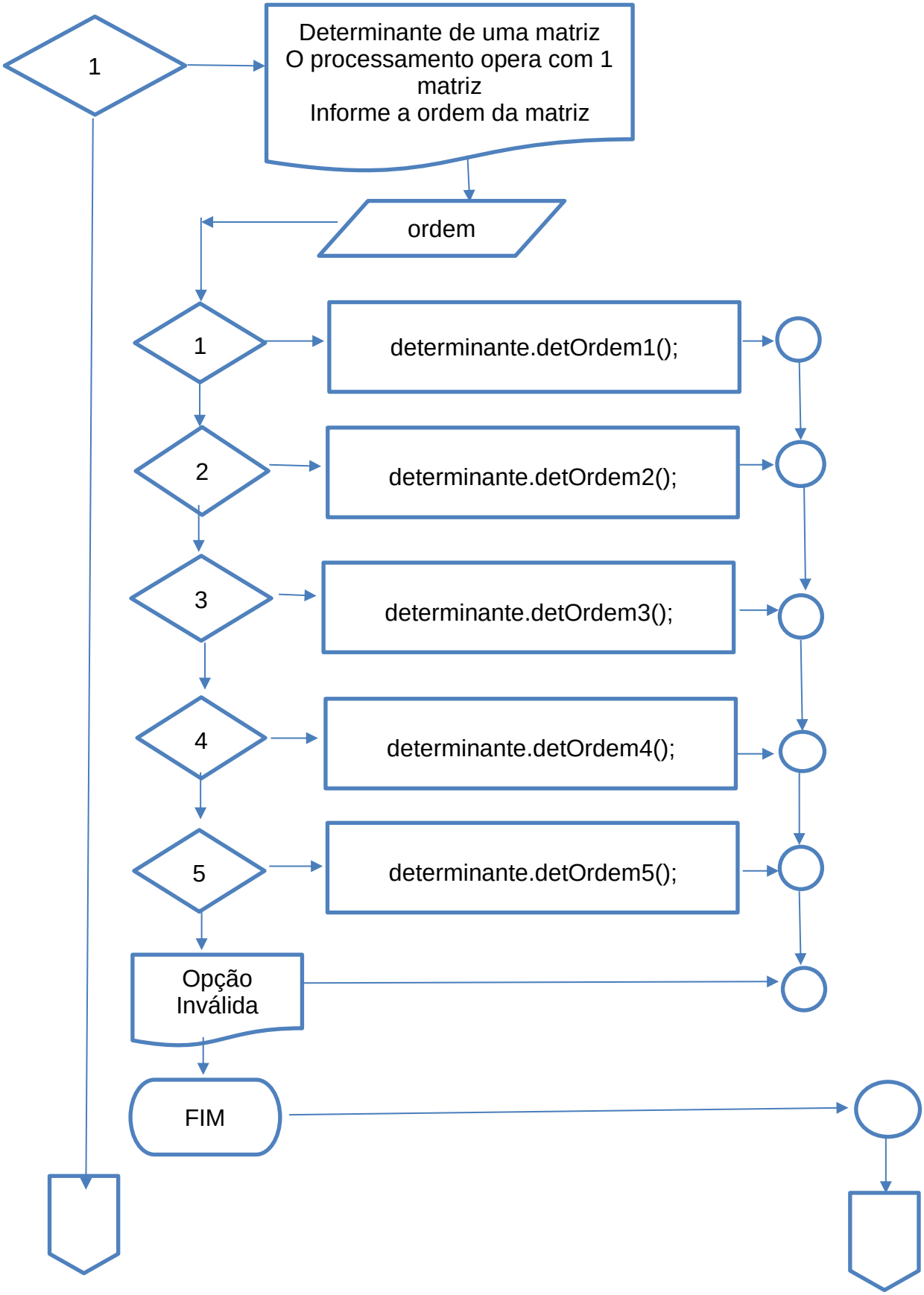

Menu

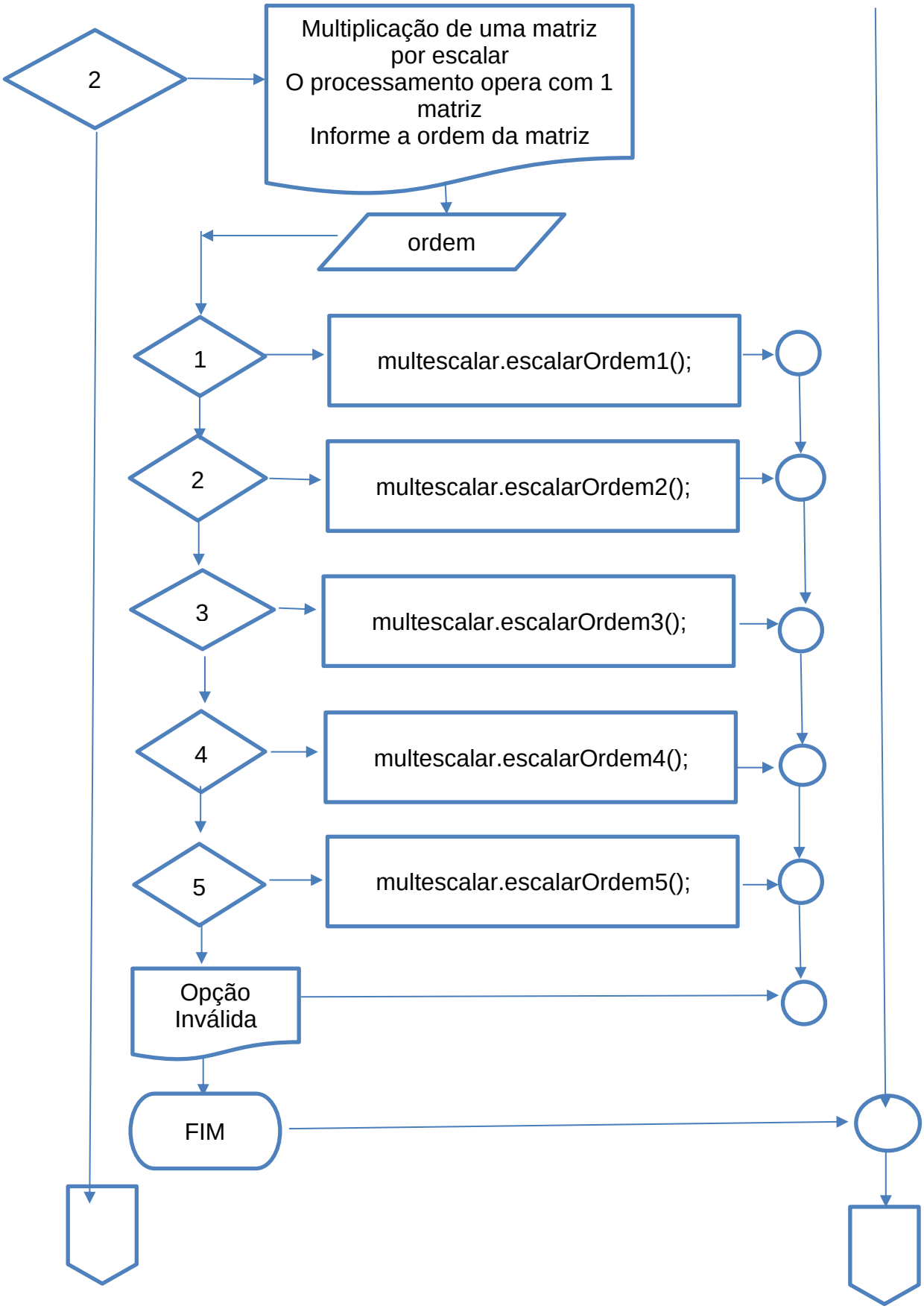
Opções: Operação e ordem da matriz.

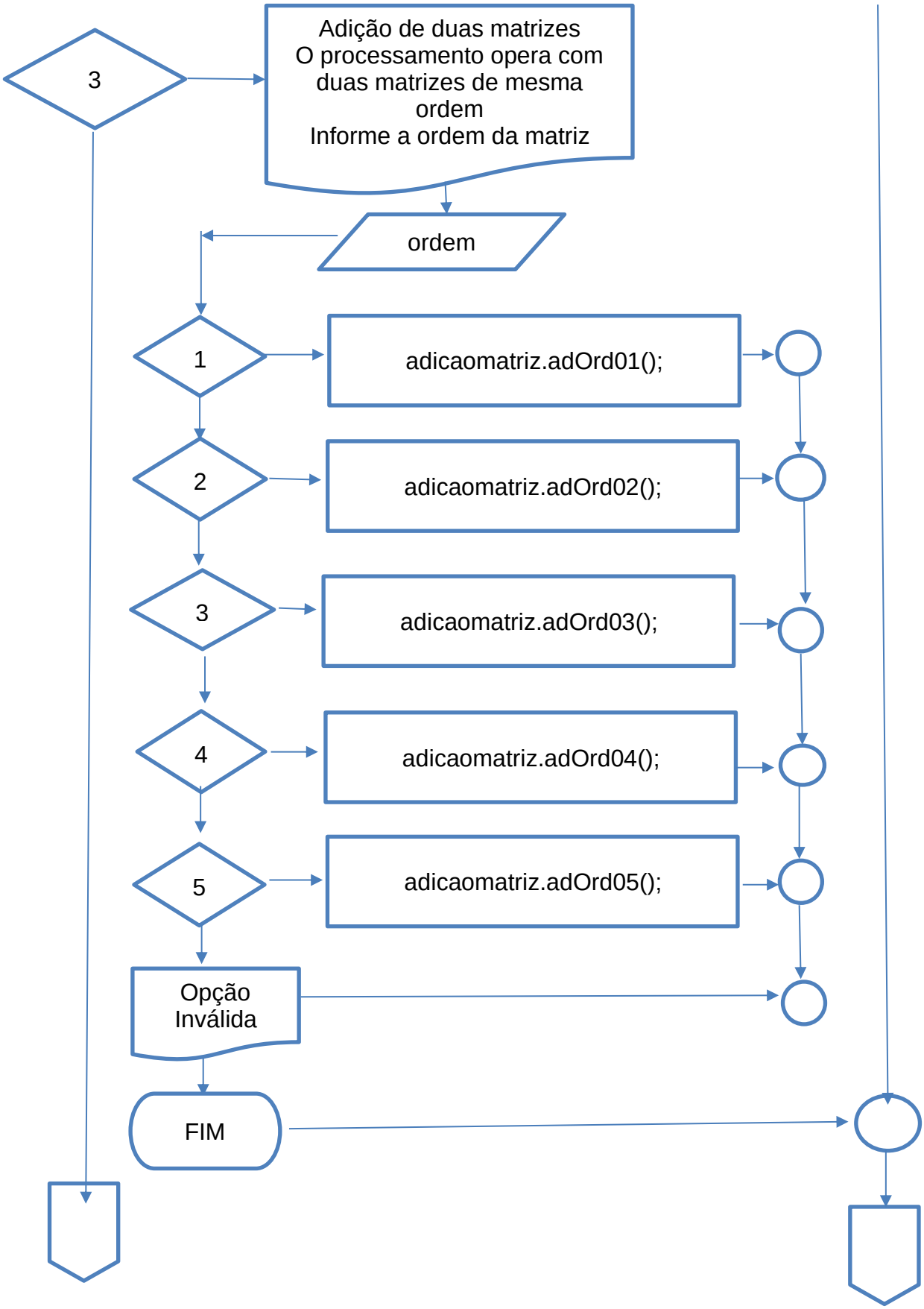
Leitura dos dados da matriz.

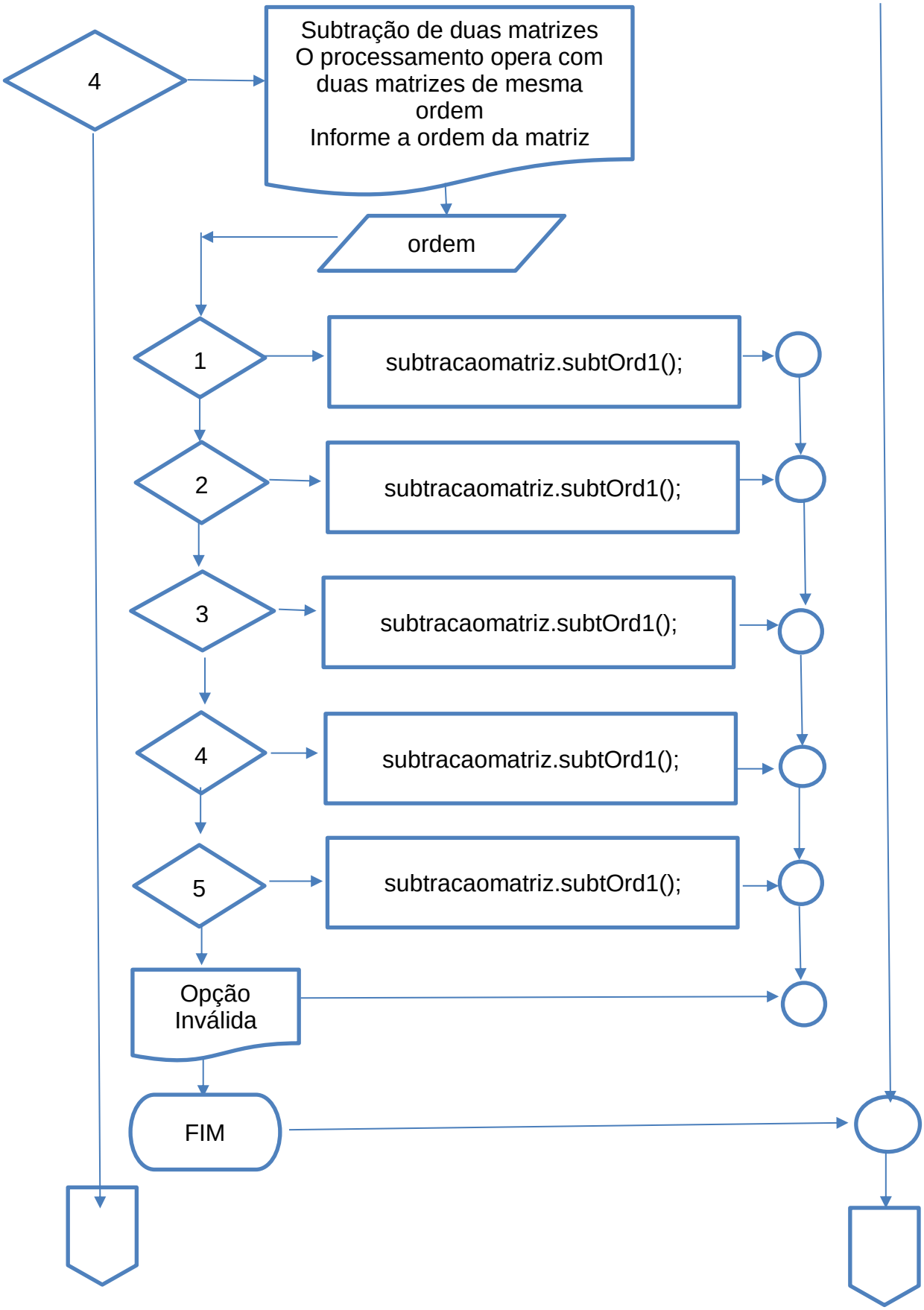
Impressão dos resultados.

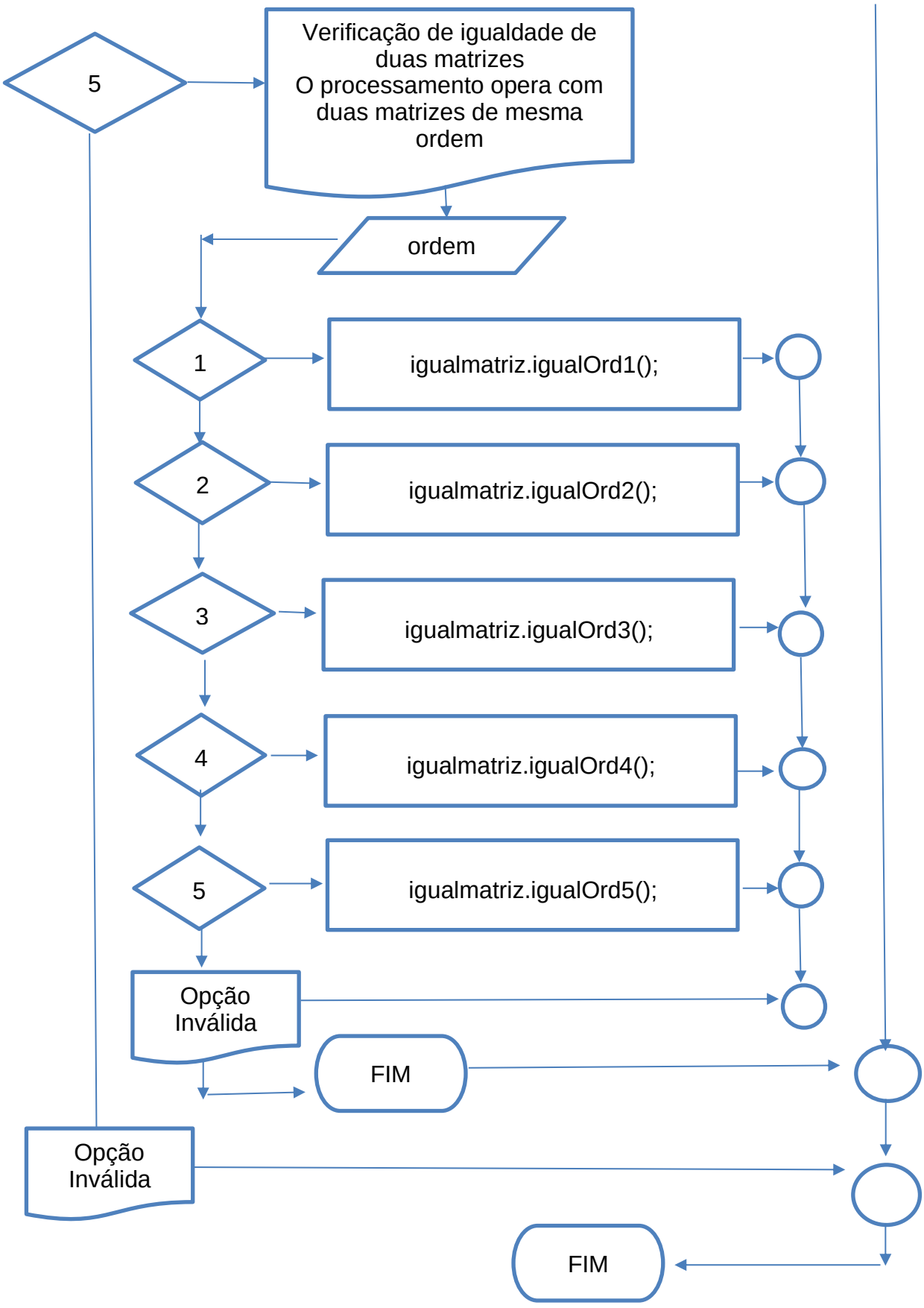

INICIO
Calculadora de Matrizes
Opções
1. Determinante de uma matriz
2. Multiplicação de uma matriz por escalar
3. Adição de duas matrizes
4. Subtração de duas matrizes
5. Verificação de igualdade de duas matrizes
Informe sua opção
opcao
opcao


1
Determinante de uma matriz
O processamento opera com 1 matriz
Informe a ordem da matriz
ordem
1
determinante.detOrdem1();
2
determinante.detOrdem2();
3
determinante.detOrdem3();
4
determinante.detOrdem4();
5
determinante.detOrdem5();
Opção Inválida
FIM

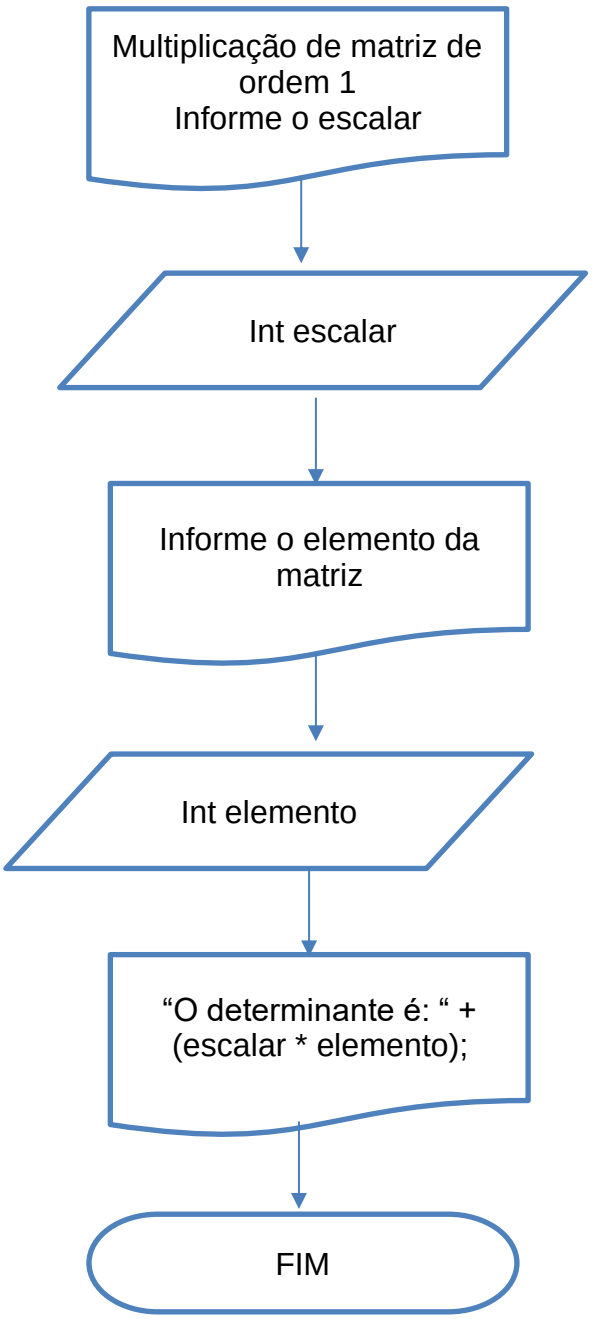

2
Multiplicação de uma matriz
por escalar
O processamento opera com 1
matriz
Informe a ordem da matriz
ordem
1
multescalar.escalarOrdem1();
2
multescalar.escalarOrdem2();
3
multescalar.escalarOrdem3();
4
multescalar.escalarOrdem4();
5
multescalar.escalarOrdem5();
Opção
Inválida
FIM


3
Adição de duas matrizes
O processamento opera com duas matrizes de mesma ordem
Informe a ordem da matriz
ordem
1
adicaomatriz.adOrd01();
2
adicaomatriz.adOrd02();
3
adicaomatriz.adOrd03();
4
adicaomatriz.adOrd04();
5
adicaomatriz.adOrd05();
Opção Inválida
FIM


4
Subtração de duas matrizes
O processamento opera com duas matrizes de mesma ordem
Informe a ordem da matriz
ordem
1
subtracaomatriz.subtOrd1();
2
subtracaomatriz.subtOrd1();
3
subtracaomatriz.subtOrd1();
4
subtracaomatriz.subtOrd1();
5
subtracaomatriz.subtOrd1();
Opção Inválida
FIM


5
Verificação de igualdade de duas matrizes
O processamento opera com duas matrizes de mesma ordem
ordem
1
igualmatriz.igualOrd1();
2
igualmatriz.igualOrd2();
3
igualmatriz.igualOrd3();
4
igualmatriz.igualOrd4();
5
igualmatriz.igualOrd5();
Opção Inválida
FIM
Opção Inválida
FIM

Fluxograma de ilustração do método multiplicação de matriz de ordem 1 por escalar (escalarOrdem1();).




**4.2 Diagrama de Casos de Uso**

| | |
|---|---|
| Nome do caso de uso | Efetuar cálculos aritméticos com uma ou duas matrizes |
| Descrição | Permite solucionar o cálculo de determinante e a multiplicação por escalar (número) de uma matriz, bem como a soma, subtração e verificação de igualdade entre duas matrizes; matrizes quadradas, até ordem 5 (5 linhas e 5 colunas). |
| Ator envolvido | Usuário |
| Pré condições | O sistema deve estar em execução aguardando pelos comandos do usuário. |
| Pós condições | Resolução do cálculo que o usuário optar ou o abandono do seu processamento. |
| FLUXO BÁSICO | |
| Usuário | Sistema |
| | {Solicita tomada de decisão do usuário}<br>Será solicitada qual a opção, a operação a ser realizada |
| Usuário toma decisão | |
| | {Solicita dados}<br>Serão solicitados os elementos da matriz ou das matrizes |
| Fornece dados e expressões | |
| | {Calcula o valor}<br>Processamento dos dados apresentados de acordo com a operação escolhida |
| | {Exibe o resultado}<br>Informa o resultado na janela do console |
| | {Fim}<br>Fim do caso de uso |
| FLUXOS ALTERNATIVOS | |

| | |
|---|---|
| {Solicita dados e expressões} | O usuário pode abandonar o cálculo e retornar à tomada de decisão para novo cálculo ou encerrar o caso de uso, o que encerra a aplicação {Fim} |

### 4.3 Diagrama de Classes

Classe principal

| MatrizCalc |
| --- |
| int opcao, ordem, escalar,<br>coluna, linha, matriz, a, b, c,<br>cofatorA, cofatorB, cofatorC,<br>cofatorD, matrizcalc; |
| public void MatrizCalc();<br>public void Determinante();<br>public void MultEscalar();<br>public void AdicaoMatriz();<br>public void SubtracaoMatriz();<br>public void IgualdadeMatriz(); |

Subclasses

| AdicaoMatriz |
| --- |
| int ordem, matriz1, matriz2, elemento1, elemento2, resultado,<br>resultado11, resultado12,resultado13, resultado14, resultado15,<br>resultado21, resultado22, resultado23, resultado24, resultado25,<br>resultado31, resultado32, resultado33, resultado34, resultado35,<br>resultado41, resultado42, resultado43, resultado44, resultado45,<br>resultado51, resultado52, resultado53, resultado54, resultado55; |
| public void adOrd01()<br>public void adOrd02()<br>public void adOrd03()<br>public void adOrd04()<br>public void adOrd05() |

| MultEscalar |
| --- |

| |
|---|
| int ordem, escalar, linha, coluna, matriz, resultado11, resultado12, resultado13, resultado14, resultado15, resultado21, resultado22, resultado23,resultado24, resultado25, resultado31, resultado32, resultado33, resultado34, resultado35, resultado41, resultado42, resultado43, resultado44, resultado45, resultado51, resultado52, resultado53, resultado54, resultado55; |
| public void escalarOrdem1()<br>public void escalarOrdem2()<br>public void escalarOrdem3()<br>public void escalarOrdem4()<br>public void escalarOrdem5() |

| SubtracaoMatriz |
|---|
| Int ordem, matriz1, matriz2, elemento1, elemento2, resultado, resultado11, resultado12, resultado13, resultado14, resultado15, resultado21, resultado22, resultado23, resultado24, resultado25, resultado31, resultado32, resultado33, resultado34, resultado35, resultado41, resultado42, resultado43, resultado44, resultado45, resultado51, resultado52, resultado53, resultado54, resultado55; |
| public void subtOrd1()<br>public void subtOrd2()<br>public void subtOrd3()<br>public void subtOrd4()<br>public void subtOrd5() |

| Determinante |
|---|

| int elemento, linha, coluna, matriz, a, b, c, cofatorA, cofatorB, cofatorC, cofatorD, determinante; |
| --- |
| public void detOrdem1()<br>public void detOrdem2()<br>public void detOrdem3()<br>public void detOrdem4()<br>public void detOrdem5() |

| IgualdadeMatriz |
| --- |
| int elemento1, elemento2;<br>boolean diferença; |
| public void igualOrd1()<br>public void igualOrd2()<br>public void igualOrd3()<br>public void igualOrd4()<br>public void igualOrd5() |

## 5. CRONOGRAMA DE ATIVIDADES

Para melhor acompanhamento do projeto, segue o cronograma das atividades e os prazos até a implantação do sistema.

| Nome da tarefa | Duração | Início | Término |
|---|---|---|---|
| Recebimento do case | 7 dias | Sáb 07/02/15 | Seg 16/02/15 |
| Análise de Requisitos | 6 dias | Seg 16/02/15 | Seg 23/02/15 |
| Avaliação | 6 dias | Seg 23/02/15 | Seg 02/03/15 |
| Parâmetros | 21 dias | Seg 02/03/15 | Seg 30/03/15 |
| Customização | 1 dia | Sex 13/03/15 | Sex 13/03/15 |
| Aprovação | 6 dias | Seg 13/04/15 | Seg 20/04/15 |
| Habilitação | 16 dias | Seg 20/04/15 | Seg 11/05/15 |
| Testes | 8 dias | Seg 11/05/15 | Qua 20/05/15 |
| Entrega do projeto | | Dom 24/05/15 | |
| Treinamento | | | |
| Implantação | | Ter 02/02/16 | |
| | | | |

### 5.1. Definição das Principais tarefas do Ciclo de desenvolvimento.

**Fase de concepção:** é feita uma coleta das necessidades do cliente. Tais necessidades são analisadas e podem precisar de detalhamento, o que mantém o projeto em análise. Feita a análise de requisitos, o conteúdo será avaliado.

Novamente, poderá ser preciso mais dados ou informações, e o projeto volta para análise até que seja avaliado satisfatoriamente.

**Fase de desenvolvimento:** Após a avaliação, podem emergir novos pontos que não foram pensados na sua concepção, o que fará o projeto retornar à fase anterior. Senão, é verificado se o projeto já atende às necessidades do cliente através de uma parametrização ou será preciso desenvolver algo novo. Se aprovado pelo cliente, poderá ser habilitado.

Caso não exista parametrização, ou esta não atenda às necessidades do cliente, deverá o projeto ser customizado. Novamente temos a figura da aprovação pelo cliente. Após aprovado, ele entra na fila de desenvolvimento para concluir esta fase com os testes e a criação de um release.

**Fase de entrega:** O projeto será homologado e, caso seja necessário, entra em fila de desenvolvimento para aperfeiçoamento ou correção de erros. Assim, será implantando, juntamente com o devido treinamento que o software impõe ao seu usuário.

Para melhor visualização das fases, vide Apêndice B deste documento.

## 5.2. Cronograma de Atividades

As atividades descritas abaixo foram desenvolvidas no período compreendido entre fevereiro e maio de 2015. Foram iniciadas com a disponibilização do case e tem seu encerramento até 24 de maio de 2015, prazo final de entrega. Com o case, foi possível fazer um estudo geral do projeto.

A partir deste estudo, foram definidas as seguintes atividades:

- Definição dos membros do grupo de trabalho;
- Projeto geral para definição das atividades;
- Definição dos dados da empresa: a partir da escolha do nome foram definidas as informações relacionadas à empresa fictícia;

- Início da confecção do documento;
- Análise dos requisitos gerais do projeto técnico;
- Estudo do aporte teórico e elenco de bibliografia;
- Formulação de documentação: fluxograma e algoritmo não computacional. A continuidade se deu com o estabelecimento de classes e objetos, que originaram o diagrama de classes, e a codificação da classe principal.
- Revisão do documento;
- Entrega da primeira prévia;
- Conclusão da documentação;
- Codificação das subclasses;
- Formulação e realização de testes;
- Revisão e finalização do documento;
- Entrega;

# 6. INTERFACES (TELAS) DO SISTEMA E CODIFICAÇÃO

## 6.1 Telas da Codificação e do Sistema

Tela 1 – Declaração dos atributos da Classe Principal

```
package matrizcalc;

import java.util.Scanner;
public class MatrizCalc {
    int opcao, ordem, escalar, coluna, linha, matriz;
    int a, b, c, cofatorA, cofatorB, cofatorC, cofatorD;
    int matrizcalc;

    public static void main(String[] args) {
    MatrizCalc matrizcalc = new MatrizCalc();
    Determinante determinante = new Determinante();
    MultEscalar multescalar = new MultEscalar();
    AdicaoMatriz adicaomatriz = new AdicaoMatriz();
    SubtracaoMatriz subtracaomatriz = new SubtracaoMatriz();
    IgualdadeMatriz igualmatriz = new IgualdadeMatriz();
```

Tela 2 – Apresentação do menu e a leitura de informação pelo usuário que iniciará a estrutura Switch Case

```
Scanner entrada = new Scanner(System.in);
//Menu de apresentação das opções;
    System.out.println("               Calculadora de Matrizes             ");
    System.out.println("_____________________________________________________");
    System.out.println();
    System.out.println("                        Opções                     ");
    System.out.println("_____________________________________________________");
    System.out.println(" 1. Determinante de uma matriz");
    System.out.println(" 2. Multiplicação de uma matriz por escalar");
    System.out.println(" 3. Adição de duas matrizes");
    System.out.println(" 4. Subtração de duas matrizes");
    System.out.println(" 5. Verificação de igualdade de duas matrizes");
    System.out.println("_____________________________________________________");
    System.out.println();


    //Leitura da opção do usuário: operações;
    int opcao;

    System.out.println("Informe sua opção");
    opcao = entrada.nextInt();
```

Tela 3 – A primeira estrutura define qual operação será realizada. Novamente, o usuário será questionado para informar qual a ordem da matriz que deseja operar.

```
//Estrutura de escolha das operações;
switch(opcao){
    case 1:
    System.out.println();
    System.out.println("Determinante de uma matriz");
    System.out.println();
    System.out.println("O processamento opera com uma matriz");

    //Leitura sobre a ordem da matriz;
    System.out.println();
    System.out.println("Informe a ordem da matriz: de 1 à 5");
    int ordem = entrada.nextInt();
```

Tela 4 – Lida a informação sobre a ordem, iniciará uma nova estrutura Switch Case que fará o chamamento dos métodos.

```
            switch (ordem){
                case 1:
                determinante.detOrdem1();
                break;
                case 2:
                determinante.detOrdem2();
                break;
                case 3:
                determinante.detOrdem3();
                break;
                case 4:
                determinante.detOrdem4();
                break;
                case 5:
                determinante.detOrdem5();
                break;
                default:
                System.out.println();
                System.out.println("Opção inválida");
                break;
            }
        break;
```

Para esta demonstração, vamos estabelecer que o usuário opta por calcular o determinante de uma matriz de ordem 2.

Tela 5 – Classe Determinante.

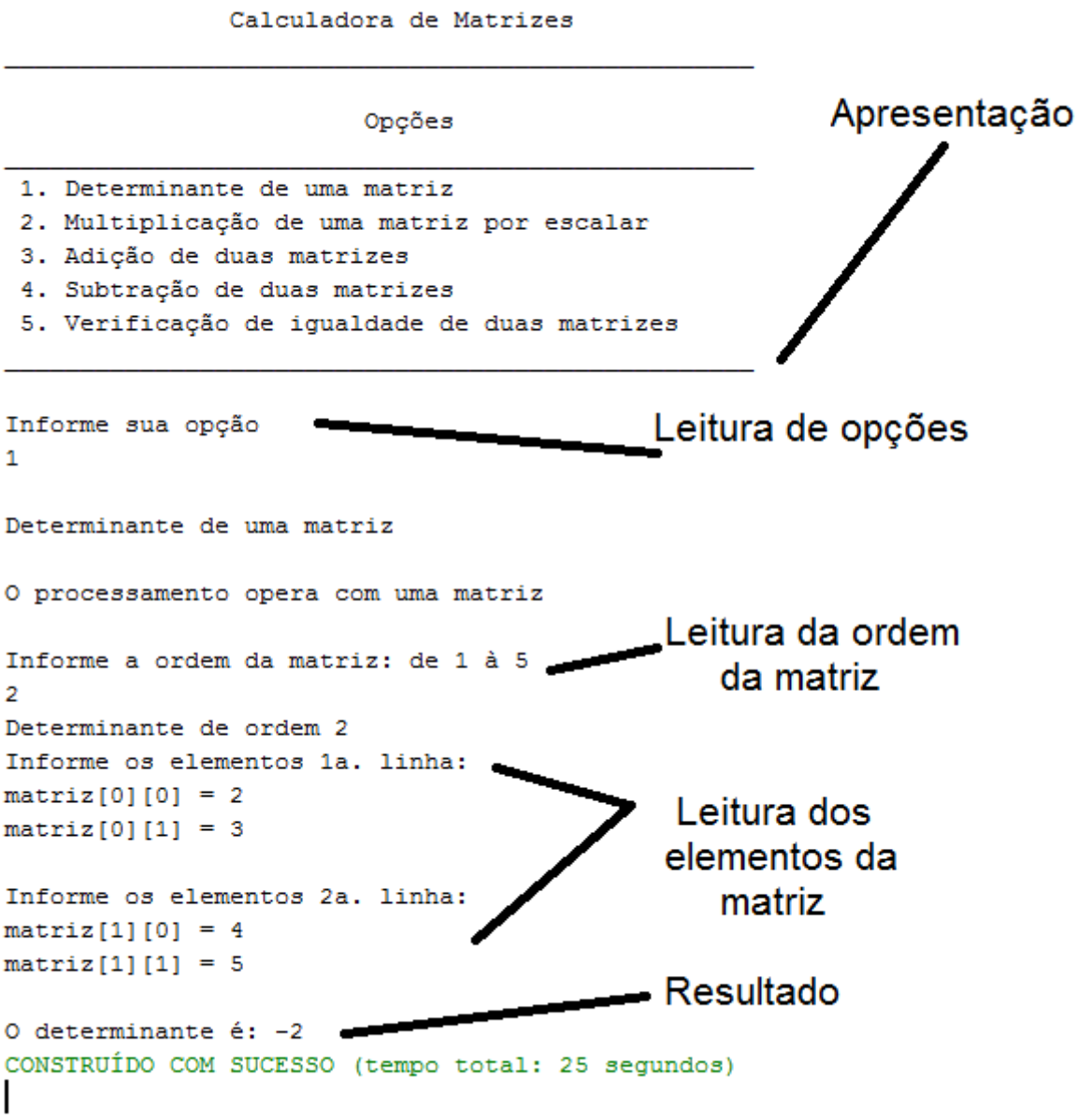
```
package matrizcalc;

import java.util.Scanner;

public class Determinante {
    Scanner entrada = new Scanner(System.in);
    //Declaração de variáveis;
    int elemento;
    int linha, coluna, matriz;
    int a, b, c;
    int cofatorA, cofatorB, cofatorC, cofatorD;
    int determinante;

    public static void main(String[] args) {

    }
```

Tela 6 – Método detOrdem2()

```
public void detOrdem2(){
int linha, coluna;
int matriz[][] = new int[2][2];
int ordem = 2;

System.out.println("Determinante de ordem 2");

//Leitura da matriz de ordem 2;
    for (linha=0; linha<ordem; linha++) {
        System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
    for (coluna=0; coluna<ordem; coluna++) {
        System.out.printf("matriz[%d][%d] = ", linha, coluna);
        matriz[linha][coluna] = entrada.nextInt();
        }
        System.out.printf("\n");
        }

determinante = ((matriz[0][0])*(matriz[1][1]))- ((matriz[1][0])* (matriz[0][1]));
System.out.println("O determinante é: " + determinante);
    }
```

Percorrida a codificação, a próxima tela se refere à saída de dados.

Tela 7 – Tela de saída




### 6.2 Especificações

O sistema é codificado em linguagem de programação JAVA com sua implementação na plataforma Net Beans IDE 8.0.2.

A codificação foi planejada a partir da criação de uma classe principal (MatrizCalc), que é a reunião das operações que a calculadora realiza, a saber: determinante, multiplicação por escalar, adição, subtração e igualdade de matrizes.

Conforme o diagrama de classes, tais operações são as subclasses, que apresentam os cálculos de cada operação.

Ato contínuo, a construção da classe principal se inicia pelo menu de apresentação, que aparecerá toda vez que o usuário iniciar o sistema. Este menu consubstancia as opções de operações disponíveis, opção que será informada pelo usuário e iniciará a estrutura switch case.

Escolhida a operação, deverá ser informado pelo usuário a ordem da matriz, de 1(um) à 5(cinco) que iniciará um novo switch case. Esta estrutura fará o chamamento do método escolhido.

Dentro de cada método, alocados nas respectivas subclasses a que pertencem, é feita a leitura dos elementos da matriz a partir do laço for, único ou duplo, a depender do método.

Finalmente, é feito o processamento de dados armazenados às fórmulas de cálculo e a apresentação do resultado na janela do console.

# 7. CONCLUSÃO

“Combati o bom combate, terminei a minha carreira, guardei a fé”.

II Carta de São Paulo à Timóteo

Este projeto não exaure a possibilidade de outras soluções uma vez que um problema pode ter inúmeras soluções, mas certamente é um resultado possível e adequado ao proposto: um produto construído com o atendimento aos padrões solicitados bem como testado para sua implementação.

Diante de todo o exposto, podemos concluir que o projeto técnico é plenamente viável e o aplicativo gera satisfação ao seu usuário. É possível ao usuário promover qualquer uma das funcionalidades com pronto e eficaz resultado.

# 8. REFERÊNCIAS BIBLIOGRÁFICAS

Algoritmos: Lógica para desenvolvimento de programação /Joé Augusto N. G. Manzano, Jayr Figueiredo de Oliveira – São Paulo: Érica, 1996.

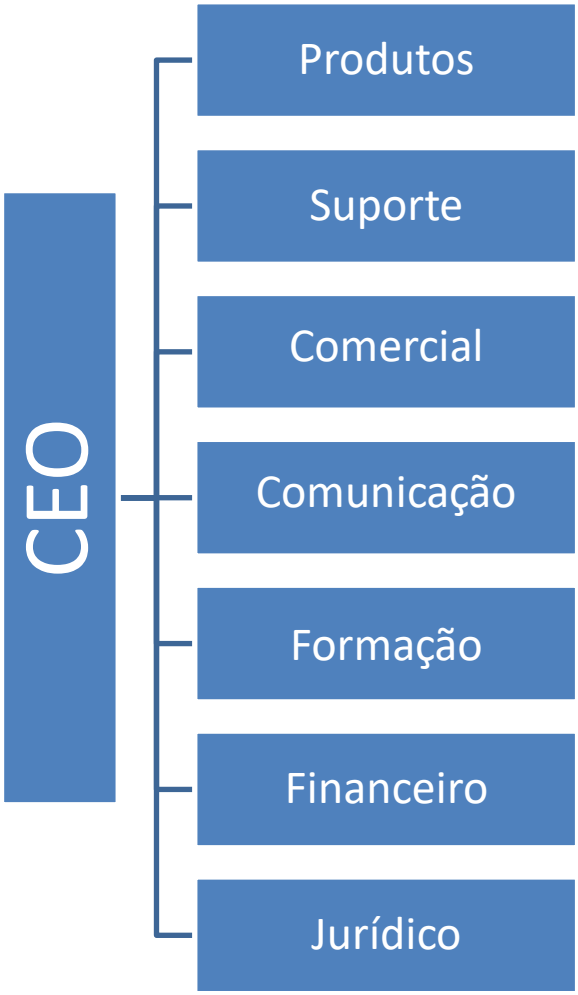
Análise e Projeto OO & UML 2.0. César Augusto Tacla. Departamento Acadêminco de Informática. Universidade Tecnológica Federal do Paraná. Disponível em: http://pt.slideshare.net/DanielMarins1/livro-anlise-e-projeto-oo-e-uml - Acesso em 23/03/2015.

Core Java, volume I: Fundamentos / Cay S. Hortsmann, Gary Cornell; tradução Carlos Schafranski e Edson Furmankiewicz; revisão técnica de Nivaldo Forest – 8.ed. – São Paulo: Pearson Prentice Hall, 2010.

Enciclopedia Barsa Universal. – Rio de Janeiro: Barsa Planeta, 2010, 11v.

Temática Barsa. – Rio de Janeiro: Barsa Planeta, 2006, 9v.:il.

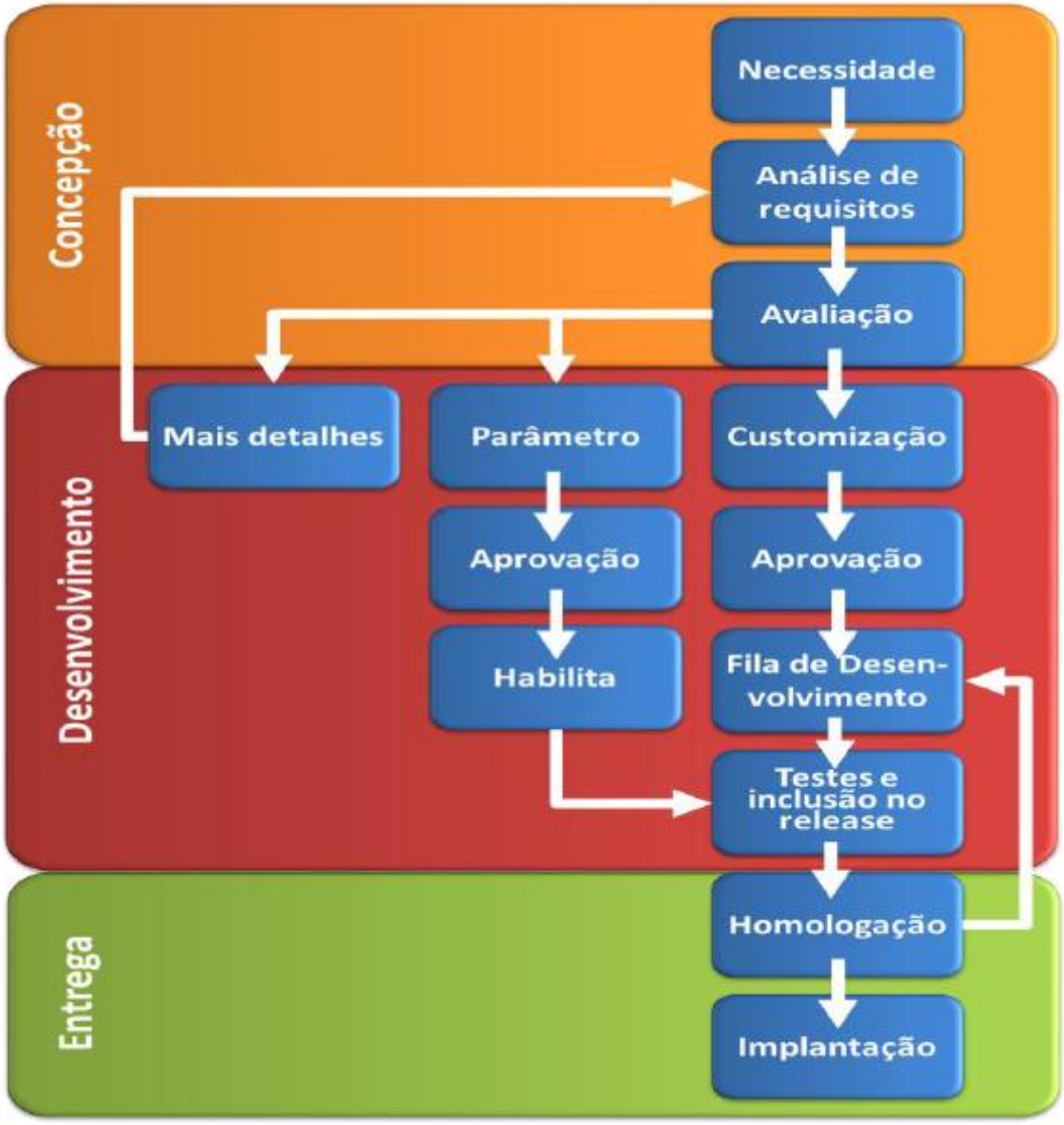
## APÊNDICE A – Organograma da Empresa




A Lisianthus é formada pelos CEO´s, seus sócios fundadores. Para uma organização total, ela é organizada em 7 gerências:

a) Gerência de Produtos, onde estão alocados analistas programadores, ou seja, o centro de criação e desenvolvimento dos sistemas; b) Gerência de suporte é o nosso primeiro canal com o cliente, que tem sua dúvida acolhida e triada. Caso seja possível, será solucionada de plano. Caso contrário, será classificado e encaminhado ao setor competente; c) Comercial: campo de atuação dos consultores junto aos clientes; d) Comunicação: nosso canal de contato com a sociedade civil; e) Formação: Atualização e treinamento dos funcionários da empresa; f) Financeiro: apoio às atividades da empresa e suporte de contabilidade para os projetos; g) Jurídico, aporte teórico para contratos e verificação de viabilidade para projetos.

# APÊNDICE B – Cronograma de Atividades

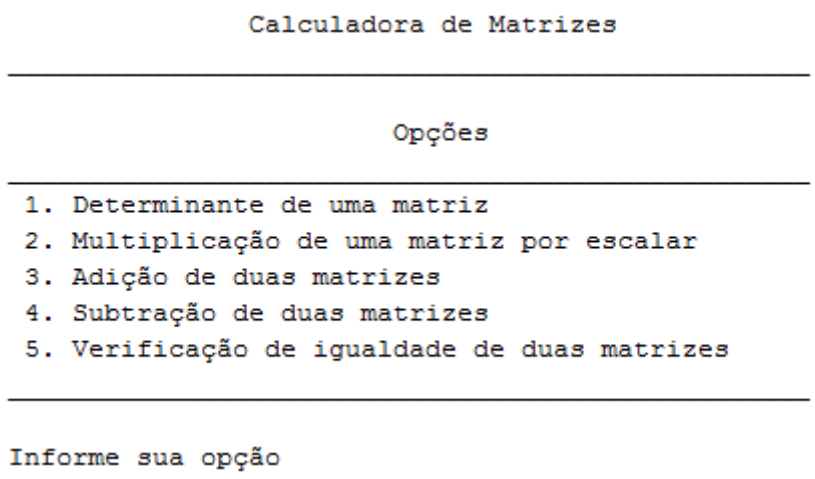


## APÊNDICE C – Interfaces do Sistema

Neste apêndice, são apresentadas as telas de saída do sistema que representam seu adequado funcionamento. Constam desta seção, os exemplos de operações disponíveis.

Observe-se que, por questão de não redundância, o exemplo de determinante pode ser consultado no item 6.1, às fls. 24 deste documento.

Tela 1 – Tela inicial: Menu.




Tela 2 – Exemplo para a operação 2, com a matriz M{[2,3],[4,5]} e escalar 5:

```
Informe sua opção
2

Multiplicação de uma matriz por escalar

O processamento opera com uma matriz

Informe a ordem da matriz: de 1 à 5
2
Multiplicação de matriz de ordem 2

Informe o escalar
5
Informe os elementos 1a. linha:
matriz[0][0] = 2
matriz[0][1] = 3

Informe os elementos 2a. linha:
matriz[1][0] = 4
matriz[1][1] = 5

Resultado:
10____15
20____25
CONSTRUÍDO COM SUCESSO (tempo total: 20 segundos)
```

Tela 3 e 4 – Exemplo de operação 3, com as matrizes A{[1,2],[3,4]} e B{[2,3],[4,5]}

```
Informe sua opção
3

Adição de duas matrizes

O processamento opera com duas matrizes
de mesma ordem

Informe a ordem da matriz
2
Adição de matrizes de ordem 2

Para esta operação, são necessárias duas matrizes

Primeira matriz
Informe os elementos 1a. linha:
matriz[0][0] = 1
matriz[0][1] = 2

Informe os elementos 2a. linha:
matriz[1][0] = 3
matriz[1][1] = 4
```

```
Segunda matriz
Informe os elementos 1a. linha:
matriz[0][0] = 2
matriz[0][1] = 3

Informe os elementos 2a. linha:
matriz[1][0] = 4
matriz[1][1] = 5

3__5
7__9
CONSTRUÍDO COM SUCESSO (tempo total: 22 segundos)
```

Tela 5 – Exemplo de operação 4, com as matrizes A{[4,9], [8,7]} e B{[2,5],[8,1]}

```
Informe sua opção
4

Subtração de duas matrizes

O processamento opera com duas matrizes
de mesma ordem

Informe a ordem da matriz
2
Subtração de matrizes de ordem 2

Primeira matriz
Informe os elementos 1a. linha:
matriz[0][0] = 4
matriz[0][1] = 9

Informe os elementos 2a. linha:
matriz[1][0] = 8
matriz[1][1] = 7

Segunda matriz
Informe os elementos 1a. linha:
matriz[0][0] = 2
matriz[0][1] = 5

Informe os elementos 2a. linha:
matriz[1][0] = 8
matriz[1][1] = 1

2__4
0__6
CONSTRUÍDO COM SUCESSO (tempo total: 23 segundos)
```

Tela 6 – Exemplo de operação 5, com as matrizes A{[1,2],[3,4]} e B{[2,3],[4,5]}

```
Informe sua opção
5

Verificação de igualdade de duas matrizes

O processamento opera com duas matrizes
de mesma ordem

Informe a ordem da matriz
2
Igualdade de matrizes de ordem 2

Primeira matriz
Informe os elementos 1a. linha:
matriz[0][0] = 1
matriz[0][1] = 2

Informe os elementos 2a. linha:
matriz[1][0] = 3
matriz[1][1] = 4


Segunda matriz
Informe os elementos 1a. linha:
matriz[0][0] = 2
matriz[0][1] = 3

Informe os elementos 2a. linha:
matriz[1][0] = 4
matriz[1][1] = 5

Matrizes desiguais
CONSTRUÍDO COM SUCESSO (tempo total: 15 segundos)
```

## APÊNDICE D – Codificação

Nesta seção, estão alocadas as codificações da classe principal (MatrizCalc) e subclasses.

1. **Código-Fonte da Classe principal MatrizCalc;**

```
/*
LISIANTHUS Soluções em Tecnologia Ltda;
*/
package matrizcalc;

import java.util.Scanner;
public class MatrizCalc {
   int opcao, ordem, escalar, coluna, linha, matriz;
   int a, b, c, cofatorA, cofatorB, cofatorC, cofatorD;
   int matrizcalc;

   public static void main(String[] args) {
   MatrizCalc matrizcalc = new MatrizCalc();
   Determinante determinante = new Determinante();
   MultEscalar multescalar = new MultEscalar();
   AdicaoMatriz adicaomatriz = new AdicaoMatriz();
   SubtracaoMatriz subtracaomatriz = new SubtracaoMatriz();
   IgualdadeMatriz igualmatriz = new IgualdadeMatriz();

   Scanner entrada = new Scanner(System.in);
   //Menu de apresentação das opções;
      System.out.println("            Calculadora de Matrizes          ");

System.out.println("_____
____");
      System.out.println();
      System.out.println("                  Opções               ");
```

```
System.out.println("_____
____");
        System.out.println(" 1. Determinante de uma matriz");
        System.out.println(" 2. Multiplicação de uma matriz por escalar");
        System.out.println(" 3. Adição de duas matrizes");
        System.out.println(" 4. Subtração de duas matrizes");
        System.out.println(" 5. Verificação de igualdade de duas matrizes");

System.out.println("_____
____");
        System.out.println();


        //Leitura da opção do usuário: operações;
        int opcao;

        System.out.println("Informe sua opção");
        opcao = entrada.nextInt();

        //Estrutura de escolha das operações;
        switch(opcao){
           case 1:
           System.out.println();
           System.out.println("Determinante de uma matriz");
           System.out.println();
           System.out.println("O processamento opera com uma matriz");

           //Leitura sobre a ordem da matriz;
           System.out.println();
           System.out.println("Informe a ordem da matriz: de 1 à 5");
           int ordem = entrada.nextInt();

              switch (ordem){
```

```
            case 1:
            determinante.detOrdem1();
            break;
            case 2:
            determinante.detOrdem2();
            break;
            case 3:
            determinante.detOrdem3();
            break;
            case 4:
            determinante.detOrdem4();
            break;
            case 5:
            determinante.detOrdem5();
            break;
            default:
            System.out.println();
            System.out.println("Opção inválida");
            break;
        }
    break;

    case 2:
    System.out.println();
    System.out.println("Multiplicação de uma matriz por escalar");
    System.out.println();
    System.out.println("O processamento opera com uma matriz");

    //Leitura da ordem das matrizes;
    System.out.println();
    System.out.println("Informe a ordem da matriz: de 1 à 5");
    ordem = entrada.nextInt();

        switch (ordem){
```

```
            case 1:
            multescalar.escalarOrdem1();
            break;
            case 2:
            multescalar.escalarOrdem2();
            break;
            case 3:
            multescalar.escalarOrdem3();
            break;
            case 4:
            multescalar.escalarOrdem4();
            break;
            case 5:
            multescalar.escalarOrdem5();
            break;
            default:
            System.out.println("Opção inválida");
            break;
        }
    break;

    case 3:
    System.out.println();
    System.out.println("Adição de duas matrizes");
    System.out.println();
    System.out.println("O processamento opera com duas matrizes \n"
                + "de mesma ordem");
    //Leitura sobre a ordem das matrizes;
    System.out.println();
    System.out.println("Informe a ordem da matriz");
    ordem = entrada.nextInt();

        switch (ordem){
            case 1:
```

```
            adicaomatriz.adOrd01();
            break;
            case 2:
            adicaomatriz.adOrd02();
            break;
            case 3:
            adicaomatriz.adOrd03();
            break;
            case 4:
            adicaomatriz.adOrd04();
            break;
            case 5:
            adicaomatriz.adOrd05();
            break;
            default:
            System.out.println("Opção inválida");
            break;
        }
    break;
    case 4:
    System.out.println();
    System.out.println("Subtração de duas matrizes");
    System.out.println();
    System.out.println("O processamento opera com duas matrizes \n"
                + "de mesma ordem");

    //Leitura sobre a ordem das matrizes;
    System.out.println();
    System.out.println("Informe a ordem da matriz");
    ordem = entrada.nextInt();

        switch (ordem){
            case 1:
            subtracaomatriz.subtOrd1();
```

```
            break;
            case 2:
            subtracaomatriz.subtOrd2();
            break;
            case 3:
            subtracaomatriz.subtOrd3();
            break;
            case 4:
            subtracaomatriz.subtOrd4();
            break;
            case 5:
            subtracaomatriz.subtOrd5();
            break;
            default:
            System.out.println("Opção inválida");
            break;
        }
break;
case 5:
System.out.println();
System.out.println("Verificação de igualdade de duas matrizes");
System.out.println();
System.out.println("O processamento opera com duas matrizes \n"
            + "de mesma ordem");

//Leitura da ordem das matrizes;
System.out.println();
System.out.println("Informe a ordem da matriz");
ordem = entrada.nextInt();

    switch (ordem){
    case 1:
    igualmatriz.igualOrd1();
    break;
```

```
                case 2:
                igualmatriz.igualOrd2();
                break;
                case 3:
                igualmatriz.igualOrd3();
                break;
                case 4:
                igualmatriz.igualOrd4();
                break;
                case 5:
                igualmatriz.igualOrd5();
                break;
                default:
                System.out.println("Opção inválida");
                break;
                }
            break;
            default:
                System.out.println("Opção inválida");
            break;
        }

}
//Chamamento dos métodos;
// a) Determinantes;
public void detOrdem1(){}
public void detOrdem2(){}
public void detOrdem3(){}
public void detOrdem4(){}
public void detOrdem5(){}
//b)Multiplicação por escalar;
public void escalarOrdem1(){}
public void escalarOrdem2(){}
public void escalarOrdem3(){}
```

```
        public void escalarOrdem4(){}
        public void escalarOrdem5(){}
        //Adição de matrizes;
        public void adOrd01(){}
        public void adOrd02(){}
        public void adOrd03(){}
        public void adOrd04(){}
        public void adOrd05(){}
        //Subtração de matrizes;
        public void subtOrd1(){}
        public void subtOrd2(){}
        public void subtOrd3(){}
        public void subtOrd4(){}
        public void subtOrd5(){}
        //Igualdade de matrizes;
        public void igualOrd1(){}
        public void igualOrd2(){}
        public void igualOrd3(){}
        public void igualOrd4(){}
        public void igualOrd5(){}

}
```

2. **Código-Fonte da Classe Determinante;**

```
/*
Esta classe realiza o cálculo do determinante de uma matriz quadrada.
*/
```

```
package matrizcalc;

import java.util.Scanner;

public class Determinante {
    Scanner entrada = new Scanner(System.in);
    //Declaração de variáveis;
    int elemento;
    int linha, coluna, matriz;
    int a, b, c;
    int cofatorA, cofatorB, cofatorC, cofatorD;
    int determinante;

    public static void main(String[] args) {

    }

    public void detOrdem1(){
    Scanner entrada = new Scanner(System.in);

    System.out.println("Determinante de ordem 1");
    System.out.println();
    //Leitura do elemento da matriz de ordem 1;
    System.out.println("Informe o elemento");
    int elemento = entrada.nextInt();
    /*O determinante da matriz de ordem 1 é o próprio elemento da matriz */
    System.out.println("O determinante é: " + elemento);
    }

    public void detOrdem2(){
    int linha, coluna;
    int matriz[][] = new int[2][2];
    int ordem = 2;
```

```
System.out.println("Determinante de ordem 2");

//Leitura da matriz de ordem 2;
    for (linha=0; linha<ordem; linha++) {
        System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
    for (coluna=0; coluna<ordem; coluna++) {
        System.out.printf("matriz[%d][%d] = ", linha, coluna);
        matriz[linha][coluna] = entrada.nextInt();
        }
        System.out.printf("\n");
        }

determinante = ((matriz[0][0])*(matriz[1][1]))- ((matriz[1][0])* (matriz[0][1]));
System.out.println("O determinante é: " + determinante);
    }

public void detOrdem3(){
int linha, coluna, matriz[][] = new int[3][3];
int ordem = 3;

System.out.println("Determinante de ordem 3");

//Leitura da matriz de ordem 3;
    for (linha=0; linha<ordem; linha++) {
        System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
    for (coluna=0; coluna<ordem; coluna++) {
        System.out.printf("matriz[%d][%d] = ", linha, coluna);
        matriz[linha][coluna] = entrada.nextInt();
        }
        System.out.printf("\n");
        }

// Resolução pela regra do Cofator;
a = matriz[0][0]* (matriz[1][1]* matriz[2][2] - matriz[2][1]* matriz[1][2]);
```

```
    b = matriz[0][1]* (matriz[1][0]* matriz[2][2] - matriz[2][0]* matriz[1][2]);
    c = matriz[0][2]* (matriz[1][0]* matriz[2][1] - matriz[2][0]* matriz[1][1]);

    determinante = a - b + c;
    System.out.println("O determinante é: " + determinante);
}

    public void detOrdem4(){
    int linha, coluna, matriz[][] = new int[4][4];
    int ordem = 4;

    System.out.println("Determinante de ordem 4");

    //Leitura dos dados da matriz 4;
        for (linha=0; linha<ordem; linha++) {
            System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
        for (coluna=0; coluna<ordem; coluna++) {
            System.out.printf("matriz[%d][%d] = ", linha, coluna);
            matriz[linha][coluna] = entrada.nextInt();
            }
            System.out.printf("\n");
            }

// A resolução é feita pelo Teorema de Laplace;
a = matriz[1][0]* (matriz[2][2]* matriz[3][3] - matriz[3][2]* matriz[2][3]);
b = matriz[1][2]* (matriz[2][0]* matriz[3][3] - matriz[3][0]* matriz[2][3]);
c = matriz[1][3]* (matriz[2][0]* matriz[3][2] - matriz[3][0]* matriz[2][2]);

    cofatorA = (-1)* (a - b + c);

a = matriz[0][0]* (matriz[2][2]* matriz[3][3] - matriz[3][2]* matriz[2][3]);
b = matriz[0][2]* (matriz[2][0]* matriz[3][3] - matriz[3][0]* matriz[2][3]);
c = matriz[0][3]* (matriz[2][0]* matriz[3][2] - matriz[3][0]* matriz[2][2]);
```

```
    cofatorB = 1 * (a - b + c);

a = matriz[0][0]* (matriz[1][2]* matriz[3][3] - matriz[3][2]* matriz[1][3]);
b = matriz[0][2]* (matriz[1][0]* matriz[3][3] - matriz[3][0]* matriz[1][3]);
c = matriz[0][3]* (matriz[1][0]* matriz[3][2] - matriz[3][0]* matriz[1][2]);

 cofatorC = (-1)* (a - b + c);

a = matriz[0][0]* (matriz[1][2]* matriz[2][3] - matriz[2][2]* matriz[1][3]);
b = matriz[0][2]* (matriz[1][0]* matriz[2][3] - matriz[2][0]* matriz[1][3]);
c = matriz[0][3]* (matriz[1][0]* matriz[2][2] - matriz[2][0]* matriz[1][2]);

    cofatorD  = 1 * (a - b + c);

    determinante = (matriz [0][1] * cofatorA) + (matriz[1][1] * cofatorB)
            + (matriz[2][1] * cofatorC) + (matriz[3][1] * cofatorD);
    System.out.println("O determinante é: " + determinante);
    }

    public void detOrdem5(){
    int linha, coluna, matriz[][] = new int[5][5];
    int ordem = 5;

    System.out.println("Determinante de ordem 5");

    //Leitura da matriz de ordem 5;
        for (linha=0; linha<ordem; linha++) {
            System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
        for (coluna=0; coluna<ordem; coluna++) {
            System.out.printf("matriz[%d][%d] = ", linha, coluna);
            matriz[linha][coluna] = entrada.nextInt();
            }
            System.out.printf("\n");
```

```
        }

 /* Para a resolução, inicialmente será aplicada a regra de Chió. Ela transforma
      a matriz de ordem 5 em ordem 4. A partir disto, optou-se pela resolução
      com a aplicação de Laplace*/

//Transformação pela regra de Chió.
//Primeira Coluna;
matriz[1][1]= matriz[1][1] - (matriz[0][1] * matriz[1][0]);
matriz[2][1]= matriz[2][1] - (matriz[0][1] * matriz[2][0]);
matriz[3][1]= matriz[3][1] - (matriz[0][1] * matriz[3][0]);
matriz[4][1]= matriz[4][1] - (matriz[0][1] * matriz[4][0]);

//Segunda Coluna;
matriz[1][2]= matriz[1][2] - (matriz[0][2] * matriz[1][0]);
matriz[2][2]= matriz[2][2] - (matriz[0][2] * matriz[2][0]);
matriz[3][2]= matriz[3][2] - (matriz[0][2] * matriz[3][0]);
matriz[4][2]= matriz[4][2] - (matriz[0][2] * matriz[4][0]);

//Terceira Coluna;
matriz[1][3]= matriz[1][3] - (matriz[0][3] * matriz[1][0]);
matriz[2][3]= matriz[2][3] - (matriz[0][3] * matriz[2][0]);
matriz[3][3]= matriz[3][3] - (matriz[0][3] * matriz[3][0]);
matriz[4][3]= matriz[4][3] - (matriz[0][3] * matriz[4][0]);

//Quarta Coluna;
matriz[1][4]= matriz[1][4] - (matriz[0][4] * matriz[1][0]);
matriz[2][4]= matriz[2][4] - (matriz[0][4] * matriz[2][0]);
matriz[3][4]= matriz[3][4] - (matriz[0][4] * matriz[3][0]);
matriz[4][4]= matriz[4][4] - (matriz[0][4] * matriz[4][0]);

/*
Feita a transformações, a resolução se dá por Laplace
*/
```

```
a = matriz[2][2]* (matriz[3][3]* matriz[4][4] - matriz[4][3]* matriz[3][4]);
b = matriz[2][3]* (matriz[3][2]* matriz[4][4] - matriz[4][2]* matriz[3][4]);
c = matriz[2][4]* (matriz[3][2]* matriz[4][3] - matriz[4][2]* matriz[3][3]);
   cofatorA = 1 * (a - b + c);

a = matriz[1][2]* (matriz[3][3]* matriz[4][4] - matriz[4][3]* matriz[3][4]);
b = matriz[1][3]* (matriz[3][2]* matriz[4][4] - matriz[4][2]* matriz[3][4]);
c = matriz[1][4]* (matriz[3][2]* matriz[4][3] - matriz[4][2]* matriz[3][3]);
   cofatorB = (-1)* (a - b + c);

a = matriz[1][2]* (matriz[2][3]* matriz[4][4] - matriz[4][3]* matriz[2][4]);
b = matriz[1][3]* (matriz[2][2]* matriz[4][4] - matriz[4][2]* matriz[2][4]);
c = matriz[1][4]* (matriz[2][2]* matriz[4][3] - matriz[4][2]* matriz[2][3]);
   cofatorC = 1 * (a - b + c);

a = matriz[1][2]* (matriz[2][3]* matriz[3][4] - matriz[3][3]* matriz[2][4]);
b = matriz[1][3]* (matriz[2][2]* matriz[3][4] - matriz[3][2]* matriz[2][4]);
c = matriz[1][4]* (matriz[2][2]* matriz[3][3] - matriz[3][2]* matriz[2][3]);
   cofatorD = (-1)* (a - b + c);

   determinante = matriz[1][1] * cofatorA + matriz[2][1]* cofatorB +
      matriz[3][1] * cofatorC + matriz[4][1]*cofatorD;

   System.out.println("O determinante é: " + determinante);
}
}
```

### 3. Código-Fonte da Classe MultEscalar;

```
/*
Esta classe realiza a multiplicação de uma matriz quadrada por um escalar, que
são os números
*/
package matrizcalc;
```

```
import java.util.Scanner;

public class MultEscalar {
   int ordem, escalar, linha, coluna, matriz;
   int resultado11, resultado12, resultado13, resultado14, resultado15;
   int resultado21, resultado22, resultado23,resultado24, resultado25;
   int resultado31, resultado32, resultado33, resultado34, resultado35;
   int resultado41, resultado42, resultado43, resultado44, resultado45;
   int resultado51, resultado52, resultado53, resultado54, resultado55;



   public static void main(String[] args) {
   int ordem, escalar;

   Scanner entrada = new Scanner(System.in);
   }

      public void escalarOrdem1(){
         Scanner entrada = new Scanner(System.in);
         System.out.println("Multiplicação de matriz de ordem 1");
         System.out.println();

         System.out.println("Informe o escalar");
         int escalar = entrada.nextInt();

         //Leitura da matriz;
         System.out.println("Informe o elemento da matriz");
         int elemento = entrada.nextInt();
         System.out.println("O determinante é: " + ( escalar * elemento ));
      }



      public void escalarOrdem2(){
```

```
Scanner entrada = new Scanner(System.in);

int linha, coluna;
int matriz[][] = new int[2][2];
int ordem = 2;

System.out.println("Multiplicação de matriz de ordem 2");
System.out.println();
System.out.println("Informe o escalar");
escalar = entrada.nextInt();



//Leitura da matriz de ordem 2;
for (linha=0; linha<ordem; linha++) {
   System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
for (coluna=0; coluna<ordem; coluna++) {
   System.out.printf("matriz[%d][%d] = ", linha, coluna);
   matriz[linha][coluna] = entrada.nextInt();
   }
   System.out.printf("\n");
   }

//Cálculo das multiplicações;
resultado11 = matriz[0][0] * escalar;
resultado12= matriz[0][1] * escalar;
resultado21 = matriz[1][0] * escalar;
resultado22 = matriz[1][1] * escalar;



//Apresentação dos resultados;
System.out.println("Resultado: ");
System.out.println(resultado11 + "____" + resultado12);
System.out.println(resultado21 + "____" + resultado22);
```

```
}

public void escalarOrdem3(){
   Scanner entrada = new Scanner(System.in);

   int linha, coluna;
   int matriz[][] = new int[3][3];
   int ordem = 3;

   System.out.println("Multiplicação de matriz de ordem 3");
   System.out.println();
   System.out.println("Informe o escalar");
   int escalar = entrada.nextInt();



   //Leitura da matriz de ordem 3;
   for (linha=0; linha<ordem; linha++) {
      System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
   for (coluna=0; coluna<ordem; coluna++) {
      System.out.printf("matriz[%d][%d] = ", linha, coluna);
      matriz[linha][coluna] = entrada.nextInt();
      }
      System.out.printf("\n");
      }



   //Cálculo das multiplicações;
   resultado11 = matriz[0][0] * escalar;
   resultado12 = matriz[0][1] * escalar;
   resultado13 = matriz[0][2] * escalar;
   resultado21 = matriz[1][0] * escalar;
   resultado22 = matriz[1][1] * escalar;
   resultado23 = matriz[1][2]* escalar;
```

```
        resultado31 = matriz[2][0] * escalar;
        resultado32 = matriz[2][1] * escalar;
        resultado33 = matriz[2][2] * escalar;

         //Apresentação dos resultados;


        System.out.println("Resultado: ");

        System.out.println(resultado11 + "____" +
                resultado12 + "____" + resultado13);

        System.out.println(resultado21 + "____" + resultado22 +
                "____" + resultado23);

        System.out.println(resultado31 + "____" + resultado32 +
                "____" + resultado33);

    }

public void escalarOrdem4(){
    Scanner entrada = new Scanner(System.in);

    int linha, coluna;
    int matriz[][] = new int[4][4];
    int ordem = 4;

        System.out.println("Multiplicação de matriz de ordem 3");
        System.out.println();
        System.out.println("Informe o escalar");
        int escalar = entrada.nextInt();



        //Leitura da matriz de ordem 4;
```

```
for (linha=0; linha<ordem; linha++) {
   System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
for (coluna=0; coluna<ordem; coluna++) {
   System.out.printf("matriz[%d][%d] = ", linha, coluna);
   matriz[linha][coluna] = entrada.nextInt();
   }
   System.out.printf("\n");
   }

//Cálculo das multiplicações;

resultado11 = matriz[0][0] * escalar;
resultado12 = matriz[0][1] * escalar;
resultado13 = matriz[0][2] * escalar;
resultado14 = matriz[0][3] * escalar;
resultado21 = matriz[1][0] * escalar;
resultado22 = matriz[1][1] * escalar;
resultado23 = matriz[1][2] * escalar;
resultado24 = matriz[1][3] * escalar;
resultado31 = matriz[2][0] * escalar;
resultado32 = matriz[2][1] * escalar;
resultado33 = matriz[2][2] * escalar;
resultado34 = matriz[2][3] * escalar;
resultado41 = matriz[3][0] * escalar;
resultado42 = matriz[3][1] * escalar;
resultado43 = matriz[3][2] * escalar;
resultado44= matriz[3][3] * escalar;

//Apresentação dos resultados;


System.out.println("Resultado: ");

System.out.println(resultado11 + "____" + resultado12 + "____"
```

```
                + resultado13 + "____" + resultado14);

        System.out.println(resultado21 + "____" + resultado22 + "____"
                + resultado23 + "____" + resultado24);

        System.out.println(resultado31 + "____" + resultado32 + "____"
                + resultado33 + "____" + resultado34);

        System.out.println(resultado41 + "____" + resultado42 + "____"
                + resultado43 + "____" + resultado44);

    }

public void escalarOrdem5(){
Scanner entrada = new Scanner(System.in);

int linha, coluna;
int matriz[][] = new int[5][5];
int ordem = 5;

    System.out.println("Multiplicação de matriz de ordem 3");
    System.out.println();
    System.out.println("Iinforme o escalar");
    int escalar = entrada.nextInt();


    //Leitura da matriz de ordem 5;
        for (linha=0; linha<ordem; linha++) {
            System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
        for (coluna=0; coluna<ordem; coluna++) {
            System.out.printf("matriz[%d][%d] = ", linha, coluna);
            matriz[linha][coluna] = entrada.nextInt();
            }
            System.out.printf("\n");
```

```
        }

    //Cálculo das multiplicações;
        resultado11 = matriz[0][0] * escalar;
        resultado12= matriz[0][1] * escalar;
        resultado13 = matriz[0][2] * escalar;
        resultado14 = matriz[0][3] * escalar;
        resultado15 = matriz[0][4] * escalar;

        resultado21 = matriz[1][0] * escalar;
        resultado22 = matriz[1][1] * escalar;
        resultado23 = matriz[1][2] * escalar;
        resultado24 = matriz[1][3] * escalar;
        resultado25 = matriz[1][4] * escalar;

        resultado31 = matriz[2][0] * escalar;
        resultado32 = matriz[2][1] * escalar;
        resultado33 = matriz[2][2] * escalar;
        resultado34 = matriz[2][3] * escalar;
        resultado35 = matriz[2][4] * escalar;

        resultado41 = matriz[3][0] * escalar;
        resultado42 = matriz[3][1] * escalar;
        resultado43 = matriz[3][2] * escalar;
        resultado44 = matriz[3][3] * escalar;
        resultado45 = matriz[3][4] * escalar;

        resultado51 = matriz[4][0] * escalar;
        resultado52 = matriz[4][1] * escalar;
        resultado53 = matriz[4][2] * escalar;
        resultado54 = matriz[4][3] * escalar;
        resultado55 = matriz[4][4] * escalar;

        //Apresentação dos resultados;
```

```
        System.out.println("Resultado: ");

        System.out.println(resultado11 + "____" + resultado12 + "____" +
              resultado13 + "____" + resultado14  + "____" + resultado15);

        System.out.println(resultado21 + "____" + resultado22 + "____" +
              resultado23 + "____" + resultado24  + "____" + resultado25);

        System.out.println(resultado31 + "____" + resultado32 + "____" +
              resultado33 + "____" + resultado34  + "____" + resultado35);

        System.out.println(resultado41 + "____" + resultado42 + "____" +
              resultado43 + "____" + resultado44  + "____" + resultado45);

        System.out.println(resultado51 + "____" + resultado52 + "____" +
              resultado53 + "____" + resultado54  + "____" + resultado55);
}
}
```

**4. Código-Fonte da Classe AdicaoMatrizes;**

```
/*
Esta classe realiza da adição de duas matrizes quadradas;
*/


package matrizcalc;

import java.util.Scanner;

public class AdicaoMatriz {
   int ordem, matriz1, matriz2;
   int elemento1, elemento2, resultado;
   int resultado11, resultado12,resultado13, resultado14, resultado15;
```

```
int resultado21, resultado22, resultado23, resultado24, resultado25;
int resultado31, resultado32, resultado33, resultado34, resultado35;
int resultado41, resultado42, resultado43, resultado44, resultado45;
int resultado51, resultado52, resultado53, resultado54, resultado55;


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

public void adOrd01(){
Scanner entrada = new Scanner(System.in);
System.out.println("Adição de matrizes de ordem 1");
System.out.println();
System.out.println("Para esta operação, são necessárias duas matrizes");

//Leitura dos elementos das matrizes de ordem 1;
System.out.println();
System.out.println("Informe o elemento da primeira matriz");
elemento1 = entrada.nextInt();
System.out.println();
System.out.println("Informe o elemento da segunda matriz");
elemento2 = entrada.nextInt();

    resultado = elemento1 + elemento2;
    System.out.println("A soma é: " + resultado);
}

public void adOrd02(){
Scanner entrada = new Scanner(System.in);

int linha, coluna;
int matriz1[][] = new int[2][2];
int matriz2[][] = new int [2][2];
int ordem = 2;
```

```
System.out.println("Adição de matrizes de ordem 2");
System.out.println();
System.out.println("Para esta operação, são necessárias duas matrizes");
System.out.println();

//Leitura da primeira matriz de ordem 2;
System.out.println("Primeira matriz");
    for (linha=0; linha<ordem; linha++) {
        System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
    for (coluna=0; coluna<ordem; coluna++) {
        System.out.printf("matriz[%d][%d] = ", linha, coluna);
        matriz1[linha][coluna] = entrada.nextInt();
        }
        System.out.printf("\n");
        }
//Leitura da segunda matriz de ordem 2;
System.out.println();
System.out.println("Segunda matriz");
    for (linha=0; linha<ordem; linha++) {
        System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
    for (coluna=0; coluna<ordem; coluna++) {
        System.out.printf("matriz[%d][%d] = ", linha, coluna);
        matriz2[linha][coluna] = entrada.nextInt();
        }
        System.out.printf("\n");
        }

    //Cálculo da adição;
    //Primeira linha da matriz;
    resultado11 = matriz1[0][0] + matriz2[0][0];
    resultado12 = matriz1[0][1] + matriz2[0][1];
    //Segunda linha da matriz;
    resultado21 = matriz1[1][0] + matriz2[1][0];
```

```
    resultado22 = matriz1[1][1] + matriz2[1][1];

    //Apresentação dos resultados:
    System.out.println(resultado11 + "__" + resultado12);
    System.out.println(resultado21 + "__" + resultado22);

}


public void adOrd03(){
Scanner entrada = new Scanner(System.in);
System.out.println();

int linha, coluna;
int matriz1[][] = new int[3][3];
int matriz2[][] = new int [3][3];
int ordem = 3;

System.out.println("Adição de matrizes de ordem 3");
System.out.println();
System.out.println("Para esta operação, são necessárias duas matrizes");
System.out.println();

//Leitura da primeira matriz de ordem 3;
System.out.println("Primeira matriz");
    for (linha=0; linha<ordem; linha++) {
        System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
    for (coluna=0; coluna<ordem; coluna++) {
        System.out.printf("matriz[%d][%d] = ", linha, coluna);
        matriz1[linha][coluna] = entrada.nextInt();
        }
        System.out.printf("\n");
        }
```

```
//Leitura da segunda matriz de ordem 3;
System.out.println("Segunda matriz");
   for (linha=0; linha<ordem; linha++) {
      System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
   for (coluna=0; coluna<ordem; coluna++) {
      System.out.printf("matriz[%d][%d] = ", linha, coluna);
      matriz2[linha][coluna] = entrada.nextInt();
      }
      System.out.printf("\n");
      }

   //Cálculo de adição;
   //Primeira linha da matriz;
   resultado11 = matriz1[0][0] + matriz2[0][0];
   resultado12 = matriz1[0][1] + matriz2[0][1];
   resultado13 = matriz1[0][2] + matriz2[0][2];
   //Segunda linha da matriz;
   resultado21 = matriz1[1][0] + matriz2[1][0];
   resultado22 = matriz1[1][1] + matriz2[1][1];
   resultado23 = matriz1[1][2] + matriz2[1][2];
   //Terceira linha da matriz;
   resultado31 = matriz1[2][0] + matriz2[2][0];
   resultado32 = matriz1[2][1] + matriz2[2][1];
   resultado33 = matriz1[2][2] + matriz2[2][2];

   //Apresentação dos resultados;
   System.out.println(resultado11 + "__" + resultado12 + "__" + resultado13);
   System.out.println(resultado21 + "__" + resultado22 + "__" + resultado23);
   System.out.println(resultado31 + "__" + resultado32 + "__" + resultado33);
}
public void adOrd04(){
Scanner entrada = new Scanner(System.in);

int linha, coluna;
```

```
int matriz1[][] = new int[4][4];
int matriz2[][] = new int [4][4];
int ordem = 4;

System.out.println("Adição de matrizes de ordem 4");
System.out.println();
System.out.println("Para esta operação, são necessárias duas matrizes");

//Leitura da primeira matriz de ordem 4;
System.out.println();
System.out.println("Primeira matriz");
   for (linha=0; linha<ordem; linha++) {
      System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
   for (coluna=0; coluna<ordem; coluna++) {
      System.out.printf("matriz[%d][%d] = ", linha, coluna);
      matriz1[linha][coluna] = entrada.nextInt();
      }
      System.out.printf("\n");
      }
//Leitura da segunda matriz de ordem 4;
System.out.println("Segunda matriz");
   for (linha=0; linha<ordem; linha++) {
      System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
   for (coluna=0; coluna<ordem; coluna++) {
      System.out.printf("matriz[%d][%d] = ", linha, coluna);
      matriz2[linha][coluna] = entrada.nextInt();
      }
      System.out.printf("\n");
      }

   //Cálculo de Adição;
   //Primeira linha da matriz;
   resultado11 = matriz1[0][0] + matriz2[0][0];
   resultado12 = matriz1[0][1] + matriz2[0][1];
```

```
        resultado13 = matriz1[0][2] + matriz2[0][2];
        resultado14 = matriz1[0][3] + matriz2[0][3];
        //Segunda linha da matriz;
        resultado21 = matriz1[1][0] + matriz2[1][0];
        resultado22 = matriz1[1][1] + matriz2[1][1];
        resultado23 = matriz1[1][2] + matriz2[1][2];
        resultado24 = matriz1[1][3] + matriz2[1][3];
        //Terceira linha da matriz;
        resultado31 = matriz1[2][0] + matriz2[2][0];
        resultado32 = matriz1[2][1] + matriz2[2][1];
        resultado33 = matriz1[2][2] + matriz2[2][2];
        resultado34 = matriz1[2][3] + matriz2[2][3];
        //Quarta linha da matriz;
        resultado41 = matriz1[3][0] + matriz2[3][0];
        resultado42 = matriz1[3][1] + matriz2[3][1];
        resultado43 = matriz1[3][2] + matriz2[3][2];
        resultado44 = matriz1[3][3] + matriz2[3][3];

        //Apresentação dos resultados;
        System.out.println(resultado11 + "__" + resultado12 + "__" + resultado13 + "__"
+ resultado14);
        System.out.println(resultado21 + "__" + resultado22 + "__" + resultado23 + "__"
+ resultado24);
        System.out.println(resultado31 + "__" + resultado32 + "__" + resultado33 + "__"
+ resultado34);
        System.out.println(resultado41 + "__" + resultado42 + "__" + resultado43 + "__"
+ resultado44);

    }

    public void adOrd05(){

    int linha, coluna;
    int matriz1[][] = new int[5][5];
```

```
int matriz2[][] = new int[5][5];
int ordem = 5;

Scanner entrada = new Scanner(System.in);

System.out.println("Adição de matrizes de ordem 5");
    System.out.println();
    System.out.println("Para esta operação, são necessárias duas matrizes");
    System.out.println();

//Leitura da primeira matriz de ordem 5;
System.out.println("Primeira matriz");
    for (linha=0; linha<ordem; linha++) {
        System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
    for (coluna=0; coluna<ordem; coluna++) {
        System.out.printf("matriz[%d][%d] = ", linha, coluna);
        matriz1[linha][coluna] = entrada.nextInt();
        }
        System.out.printf("\n");
        }

//Leitura da segunda matriz de ordem 5;
System.out.println("Segunda matriz");
    for (linha=0; linha<ordem; linha++) {
        System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
    for (coluna=0; coluna<ordem; coluna++) {
        System.out.printf("matriz[%d][%d] = ", linha, coluna);
        matriz2[linha][coluna] = entrada.nextInt();
        }
        System.out.printf("\n");
        }

    //Cálculo da Adição;
    //Primeira linha da matriz;
```

```
resultado11 = matriz1[0][0] + matriz2[0][0];
resultado12 = matriz1[0][1] + matriz2[0][1];
resultado13 = matriz1[0][2] + matriz2[0][2];
resultado14 = matriz1[0][3] + matriz2[0][3];
resultado15 = matriz1[0][4] + matriz2[0][4];
//Segunda linha da matriz;
resultado21 = matriz1[1][0] + matriz2[1][0];
resultado22 = matriz1[1][1] + matriz2[1][1];
resultado23 = matriz1[1][2] + matriz2[1][2];
resultado24 = matriz1[1][3] + matriz2[1][3];
resultado25 = matriz1[1][4] + matriz2[1][4];
//Terceira linha da matriz;
resultado31 = matriz1[2][0] + matriz2[2][0];
resultado32 = matriz1[2][1] + matriz2[2][1];
resultado33 = matriz1[2][2] + matriz2[2][2];
resultado34 = matriz1[2][3] + matriz2[2][3];
resultado35 = matriz1[2][4] + matriz2[2][4];
//Quarta linha da matriz;
resultado41 = matriz1[3][0] + matriz2[3][0];
resultado42 = matriz1[3][1] + matriz2[3][1];
resultado43 = matriz1[3][2] + matriz2[3][2];
resultado44 = matriz1[3][3] + matriz2[3][3];
resultado45 = matriz1[3][4] + matriz2[3][4];
//Quinta linha da matriz;
resultado51 = matriz1[4][0] + matriz2[4][0];
resultado52 = matriz1[4][1] + matriz2[4][1];
resultado53 = matriz1[4][2] + matriz2[4][2];
resultado54 = matriz1[4][3] + matriz2[4][3];
resultado55 = matriz1[4][4] + matriz2[4][4];

//Apresentação dos resultados;
System.out.println(resultado11 + "__" + resultado12 + "__" + resultado13
        + "__" + resultado14 + "__" + resultado15);
```

```
        System.out.println(resultado21 + "__" + resultado22 + "__" + resultado23
                + "__" + resultado24 + "__" + resultado25);

        System.out.println(resultado31 + "__" + resultado32 + "__" + resultado33
                + "__" + resultado34 + "__" + resultado35);

        System.out.println(resultado41 + "__" + resultado42 + "__" + resultado43
                + "__" + resultado44 + "__" + resultado45);

        System.out.println(resultado51 + "__" + resultado52 + "__" + resultado53
                + "__" + resultado54 + "__" + resultado55);
    }
}
```

**5. Código-Fonte da Classe SubtracaoMatrizes;**

```
/*
Esta classe realiza a subtração de duas matrizes quadradas;
*/
package matrizcalc;

import java.util.Scanner;
public class SubtracaoMatriz {

    public static void main(String[] args) {
        Scanner entrada = new Scanner(System.in);
        int ordem, matriz1, matriz2;
    int elemento1, elemento2, resultado;
    int resultado11, resultado12,resultado13, resultado14, resultado15;
    int resultado21, resultado22, resultado23, resultado24, resultado25;
    int resultado31, resultado32, resultado33, resultado34, resultado35;
    int resultado41, resultado42, resultado43, resultado44, resultado45;
    int resultado51, resultado52, resultado53, resultado54, resultado55;

    }
```

```
public void subtOrd1(){
Scanner entrada = new Scanner(System.in);

    System.out.println("Subtração de duas matrizes de ordem 1");
    System.out.println();

    System.out.println("Informe o elemento da primeira matriz");
    int elemento1 = entrada.nextInt();

    System.out.println("Informe o elemento da segunda matriz");
    int elemento2 = entrada.nextInt();
    System.out.println();

    int resultado = elemento1 - elemento2;

    System.out.println("A subtração é: " + resultado);
}


public void subtOrd2(){
Scanner entrada = new Scanner(System.in);

int linha, coluna;
int matriz1[][] = new int[2][2];
int matriz2[][] = new int [2][2];
int ordem = 2;

System.out.println("Subtração de matrizes de ordem 2");
System.out.println();


    //Leitura da primeira matriz de ordem 2;
    System.out.println("Primeira matriz");
```

```
        for (linha=0; linha<ordem; linha++) {
            System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
        for (coluna=0; coluna<ordem; coluna++) {
            System.out.printf("matriz[%d][%d] = ", linha, coluna);
            matriz1[linha][coluna] = entrada.nextInt();
            }
            System.out.printf("\n");
            }

    //Leitura da segunda matriz de ordem 2;
    System.out.println("Segunda matriz");
        for (linha=0; linha<ordem; linha++) {
            System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
        for (coluna=0; coluna<ordem; coluna++) {
            System.out.printf("matriz[%d][%d] = ", linha, coluna);
            matriz2[linha][coluna] = entrada.nextInt();
            }
            System.out.printf("\n");
            }

            int resultado11 = matriz1[0][0] - matriz2[0][0];
            int resultado12 = matriz1[0][1] - matriz2[0][1];

            int resultado21 = matriz1[1][0] - matriz2[1][0];
            int resultado22 = matriz1[1][1] - matriz2[1][1];

            //Apresentação de resultados;
            System.out.println( resultado11 + "__" + resultado12);

            System.out.println( resultado21 + "__" + resultado22);
}


public void subtOrd3(){
```

```
Scanner entrada = new Scanner(System.in);

int linha, coluna;
int matriz1[][] = new int[3][3];
int matriz2[][] = new int [3][3];
int ordem = 3;

System.out.println("Subtração de matrizes de ordem 3");
System.out.println();

//Leitura da primeira matriz de ordem 3;
   System.out.println("Primeira matriz");
      for (linha=0; linha<ordem; linha++) {
         System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
      for (coluna=0; coluna<ordem; coluna++) {
         System.out.printf("matriz[%d][%d] = ", linha, coluna);
         matriz1[linha][coluna] = entrada.nextInt();
         }
         System.out.printf("\n");
         }

//Leitura da segunda matriz de ordem 3;
   System.out.println("Segunda matriz");
      for (linha=0; linha<ordem; linha++) {
         System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
      for (coluna=0; coluna<ordem; coluna++) {
         System.out.printf("matriz[%d][%d] = ", linha, coluna);
         matriz2[linha][coluna] = entrada.nextInt();
         }
         System.out.printf("\n");
         }
```

```
        //Primeira linha da matriz;
        int resultado11 = matriz1[0][0] - matriz2[0][0];
        int resultado12 = matriz1[0][1] - matriz2[0][1];
        int resultado13 = matriz1[0][2] - matriz2[0][2];
        //Segunda linha da matriz;
        int resultado21 = matriz1[1][0] - matriz2[1][0];
        int resultado22 = matriz1[1][1] - matriz2[1][1];
        int resultado23 = matriz1[1][2] - matriz2[1][2];
        //Terceira linha da matriz;
        int resultado31 = matriz1[2][0] - matriz2[2][0];
        int resultado32 = matriz1[2][1] - matriz2[2][1];
        int resultado33 = matriz1[2][2] - matriz2[2][2];

        //Apresentação dos resultados:

        System.out.println(resultado11 + "__" + resultado12 +
              "__" + resultado13);

        System.out.println(resultado21 + "__" + resultado22 + "__"
              + resultado23);

        System.out.println(resultado31 + "__" + resultado32 + "__"
              + resultado33);
}
public void subtOrd4(){
Scanner entrada = new Scanner(System.in);

int linha, coluna;
int matriz1[][] = new int[4][4];
int matriz2[][] = new int [4][4];
int ordem = 4;

System.out.println("Subtração de matrizes de ordem 4");
System.out.println();
```

```
//Leitura da primeira matriz de ordem 4;
System.out.println("Primeira matriz");
    for (linha=0; linha<ordem; linha++) {
        System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
    for (coluna=0; coluna<ordem; coluna++) {
        System.out.printf("matriz[%d][%d] = ", linha, coluna);
        matriz1[linha][coluna] = entrada.nextInt();
        }
        System.out.printf("\n");
        }

//Leitura da segunda matriz de ordem 4;
System.out.println("Segunda matriz");
    for (linha=0; linha<ordem; linha++) {
        System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
    for (coluna=0; coluna<ordem; coluna++) {
        System.out.printf("matriz[%d][%d] = ", linha, coluna);
        matriz2[linha][coluna] = entrada.nextInt();
        }
        System.out.printf("\n");
        }


        //Primeira linha da matriz;
        int resultado11 = matriz1[0][0] - matriz2[0][0];
        int resultado12 = matriz1[0][1] - matriz2[0][1];
        int resultado13 = matriz1[0][2] - matriz2[0][2];
        int resultado14 = matriz1[0][3] - matriz2[0][3];
        //Segunda linha da matriz;
        int resultado21 = matriz1[1][0] - matriz2[1][0];
        int resultado22 = matriz1[1][1] - matriz2[1][1];
        int resultado23 = matriz1[1][2] - matriz2[1][2];
        int resultado24 = matriz1[1][3] - matriz2[1][3];
```

```
        //Terceira linha da matriz;
        int resultado31 = matriz1[2][0] - matriz2[2][0];
        int resultado32 = matriz1[2][1] - matriz2[2][1];
        int resultado33 = matriz1[2][2] - matriz2[2][2];
        int resultado34 = matriz1[2][3] - matriz2[2][3];
        //Quarta linha da matriz;
        int resultado41 = matriz1[3][0] - matriz2[3][0];
        int resultado42 = matriz1[3][1] - matriz2[3][1];
        int resultado43 = matriz1[3][2] - matriz2[3][2];
        int resultado44 = matriz1[3][3] - matriz2[3][3];

        //Apresentação dos resultados;


        System.out.println(resultado11 + "__" + resultado12 + "__" +
                resultado13 + "__" + resultado14);

        System.out.println(resultado21 + "__" + resultado22 + "__" +
                resultado23 + "__" + resultado24);

        System.out.println(resultado31 + "__" + resultado32 + "__" +
                resultado33 + "__" + resultado34);

        System.out.println(resultado41 + "__" + resultado42 + "__" +
                resultado43 + "__" + resultado44);
    }

    public void subtOrd5(){
    Scanner entrada = new Scanner(System.in);

    int linha, coluna;
    int matriz1[][] = new int[5][5];
    int matriz2[][] = new int [5][5];
    int ordem = 5;
```

```
System.out.println("Subtração de matrizes de ordem 5");
System.out.println();

//Leitura da primeira matriz de ordem 5;
System.out.println("Primeira matriz");
    for (linha=0; linha<ordem; linha++) {
        System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
    for (coluna=0; coluna<ordem; coluna++) {
        System.out.printf("matriz[%d][%d] = ", linha, coluna);
        matriz1[linha][coluna] = entrada.nextInt();
        }
        System.out.printf("\n");
        }

//Leitura da segunda matriz;
 System.out.println("Segunda matriz");
    for (linha=0; linha<ordem; linha++) {
        System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
    for (coluna=0; coluna<ordem; coluna++) {
        System.out.printf("matriz[%d][%d] = ", linha, coluna);
        matriz2[linha][coluna] = entrada.nextInt();
        }
        System.out.printf("\n");
        }


        //Primeira linha da matriz;
        int resultado11 = matriz1[0][0] - matriz2[0][0];
        int resultado12 = matriz1[0][1] - matriz2[0][1];
        int resultado13 = matriz1[0][2] - matriz2[0][2];
        int resultado14 = matriz1[0][3] - matriz2[0][3];
        int resultado15 = matriz1[0][4] - matriz2[0][4];
```

```
//Segunda linha da matriz;
int resultado21 = matriz1[1][0] - matriz2[1][0];
int resultado22 = matriz1[1][1] - matriz2[1][1];
int resultado23 = matriz1[1][2] - matriz2[1][2];
int resultado24 = matriz1[1][3] - matriz2[1][3];
int resultado25 = matriz1[1][4] - matriz2[1][4];

//Terceira linha da matriz;
int resultado31 = matriz1[2][0] - matriz2[2][0];
int resultado32 = matriz1[2][1] - matriz2[2][1];
int resultado33 = matriz1[2][2] - matriz2[2][2];
int resultado34 = matriz1[2][3] - matriz2[2][3];
int resultado35 = matriz1[2][4] - matriz2[2][4];

//Quarta linha da matriz;
int resultado41 = matriz1[3][0] - matriz2[3][0];
int resultado42 = matriz1[3][1] - matriz2[3][1];
int resultado43 = matriz1[3][2] - matriz2[3][2];
int resultado44 = matriz1[3][3] - matriz2[3][3];
int resultado45 = matriz1[3][4] - matriz2[3][4];

//Quinta linha da matriz;
int resultado51 = matriz1[4][0] - matriz2[4][0];
int resultado52 = matriz1[4][1] - matriz2[4][1];
int resultado53 = matriz1[4][2] - matriz2[4][2];
int resultado54 = matriz1[4][3] - matriz2[4][3];
int resultado55 = matriz1[4][4] - matriz2[4][4];

//Apresentação dos resultados;

System.out.println(resultado11 + "__" + resultado12 + "__" +
      resultado13 + "__" + resultado14 + "__" + resultado15);

System.out.println(resultado21 + "__" + resultado22 + "__" +
```

```
                resultado23 + "__" + resultado24 + "__" + resultado25);

        System.out.println(resultado31 + "__" + resultado32 + "__" +
                resultado33 + "__" + resultado34 + "__" + resultado35);

        System.out.println(resultado41 + "__" + resultado42 + "__" +
                resultado43 + "__" + resultado44 + "__" + resultado45);

        System.out.println(resultado51 + "__" + resultado52 + "__" +
                resultado53 + "__" + resultado54 + "__" + resultado55);
    }
}
```

### 6. Código-Fonte da Classe IgualdadeMatrizes;

```
/*
Esta classe verifica se duas matrizes quadradas são iguais ou não.
*/
package matrizcalc;
import java.util.Scanner;
public class IgualdadeMatriz {


    public static void main(String[] args) {
    Scanner entrada = new Scanner(System.in);
    int elemento1, elemento2;
    boolean diferenca = false;

    }

    public void igualOrd1(){
        Scanner entrada = new Scanner(System.in);

        System.out.println("Igualdade de duas matrizes de ordem 1");
        System.out.println();
        //Leitura da primeira matriz de ordem 1;
        System.out.println("Informe o elemento da primeira matriz");
        int elemento1 = entrada.nextInt();
        System.out.println();
        //Leitura da primeira matriz de ordem 1;
        System.out.println("Informe o elemento da segunda matriz");
        int elemento2 = entrada.nextInt();
```

```
        if (elemento1 == elemento2){
            System.out.println("Igualdade de matrizes");
        }else{
            System.out.println("Matrizes não iguais");
        }
}

public void igualOrd2(){
Scanner entrada = new Scanner(System.in);

int linha, coluna;
int matriz1[][] = new int[2][2];
int matriz2[][] = new int [2][2];
int ordem = 2;

System.out.println("Igualdade de matrizes de ordem 2");
System.out.println();

//Leitura da primeira matriz de ordem 2;
System.out.println("Primeira matriz");
    for (linha=0; linha<ordem; linha++) {
        System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
    for (coluna=0; coluna<ordem; coluna++) {
        System.out.printf("matriz[%d][%d] = ", linha, coluna);
        matriz1[linha][coluna] = entrada.nextInt();
        }
        System.out.printf("\n");
        }

//Leitura da segunda matriz de ordem 2;
System.out.println();
System.out.println("Segunda matriz");
    for (linha=0; linha<ordem; linha++) {
        System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
    for (coluna=0; coluna<ordem; coluna++) {
        System.out.printf("matriz[%d][%d] = ", linha, coluna);
        matriz2[linha][coluna] = entrada.nextInt();
        }
        System.out.printf("\n");
        }


            boolean diferenca = false;
            for (linha=0; linha<ordem; linha++){
            for (coluna=0; coluna<ordem; coluna++){

                if (matriz1[linha][coluna] != matriz2[linha][coluna]){
                    diferenca = true;
                    }
                }
```

```
            }
            if (diferenca == false){
            System.out.println("Matrizes Idênticas");
            }else{
            System.out.println("Matrizes desiguais");
            }
            }


public void igualOrd3(){
Scanner entrada = new Scanner(System.in);

int linha, coluna;
int matriz1[][] = new int[3][3];
int matriz2[][] = new int [3][3];
int ordem = 3;

System.out.println("Igualdade de matrizes de ordem 3");
System.out.println();

//Leitura da primeira matriz de ordem 3;
System.out.println("Primeira matriz");
    for (linha=0; linha<ordem; linha++) {
         System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
    for (coluna=0; coluna<ordem; coluna++) {
        System.out.printf("matriz[%d][%d] = ", linha, coluna);
        matriz1[linha][coluna] = entrada.nextInt();
        }
        System.out.printf("\n");
        }

//Leitura da 2ª matriz de ordem 3;
System.out.println("Segunda matriz");
    for (linha=0; linha<ordem; linha++) {
        System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
    for (coluna=0; coluna<ordem; coluna++) {
        System.out.printf("matriz[%d][%d] = ", linha, coluna);
        matriz2[linha][coluna] = entrada.nextInt();
        }
        System.out.printf("\n");
        }

            boolean diferenca = false;
            for (linha=0; linha<ordem; linha++){
            for (coluna=0; coluna<ordem; coluna++){

                if (matriz1[linha][coluna] != matriz2[linha][coluna]){
                    diferenca = true;
                }
                }
```

```
            }

            if (diferenca == false){
            System.out.println("Matrizes Idênticas");
            }else{
            System.out.println("Matrizes desiguais");
            }
 }


public void igualOrd4(){
Scanner entrada = new Scanner(System.in);

int linha, coluna;
int matriz1[][] = new int[4][4];
int matriz2[][] = new int[4][4];
int ordem = 4;

System.out.println("Igualdade de matrizes de ordem 4");
System.out.println();

//Leitura da primeira matriz de ordem 4;
System.out.println("Primeira matriz");
    for (linha=0; linha<ordem; linha++) {
        System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
    for (coluna=0; coluna<ordem; coluna++) {
        System.out.printf("matriz[%d][%d] = ", linha, coluna);
        matriz1[linha][coluna] = entrada.nextInt();
        }
        System.out.printf("\n");
        }

//Leitura da segunda matriz de ordem 4;
System.out.println("Segunda matriz");
    for (linha=0; linha<ordem; linha++) {
        System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
    for (coluna=0; coluna<ordem; coluna++) {
        System.out.printf("matriz[%d][%d] = ", linha, coluna);
        matriz2[linha][coluna] = entrada.nextInt();
        }
        System.out.printf("\n");
        }

            boolean diferenca = false;
            for (linha=0; linha<ordem; linha++){
            for (coluna=0; coluna<ordem; coluna++){

                if (matriz1[linha][coluna] != matriz2[linha][coluna]){
                        diferenca = true;
                }
                }
```

```
            }

            if (diferenca == false){
            System.out.println("Matrizes Idênticas");
            }else{
            System.out.println("Matrizes desiguais");
            }
}

public void igualOrd5(){
Scanner entrada = new Scanner(System.in);

int linha, coluna;
int matriz1[][] = new int[5][5];
int matriz2[][] = new int[5][5];
int ordem = 5;

System.out.println("Igualdade de matrizes de ordem 5");
System.out.println();

//Leitura da primeira matriz de ordem 5;
System.out.println("Primeira matriz");
   for (linha=0; linha<ordem; linha++) {
      System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
   for (coluna=0; coluna<ordem; coluna++) {
      System.out.printf("matriz[%d][%d] = ", linha, coluna);
      matriz1[linha][coluna] = entrada.nextInt();
      }
      System.out.printf("\n");
      }

//Leitura da segunda matriz de ordem 5;
System.out.println("Segunda matriz");
   for (linha=0; linha<ordem; linha++) {
      System.out.printf("Informe os elementos %da. linha:\n", (linha+1));
   for (coluna=0; coluna<ordem; coluna++) {
      System.out.printf("matriz[%d][%d] = ", linha, coluna);
      matriz2[linha][coluna] = entrada.nextInt();
      }
      System.out.printf("\n");
      }

         boolean diferenca = false;
         for (linha=0; linha<ordem; linha++){
         for (coluna=0; coluna<ordem; coluna++){

            if (matriz1[linha][coluna] != matriz2[linha][coluna]){
               diferenca = true;
               }
            }
            }
```

```
                if (diferenca == false){
                System.out.println("Matrizes Idênticas");
                }else{
                System.out.println("Matrizes desiguais");
                }
}
}
```

**APÊNDICE E – Relatório de Resultados**

Determinante de uma matriz – Opção 1

| ORDEM | MATRIZ | RESULTADO | V/F |
|---|---|---|---|
| **1** | M[10] | 10 | V |
| | M[-25] | -25 | V |
| **2** | M[(1,2), (-5,-3)] | 7 | V |
| **3** | M[(5,0,1), (-2,3,4), (0,2,-1)] | -59 | V |

**Disponível em:** http://www.mundoeducacao.com/matematica/determinante-matriz-ordem-1-2-ou-3.htm - Acesso em: 21/04/2015

| | | | |
|---|---|---|---|
| **4** | M[(2,3,5,6), (4,2,1,1,), (5,1,2,3), (6,1,3,2)] | 84 | V |
| Disponível em: http://www.alunosonline.com.br/matematica/o-teorema-laplace.html – Acesso em: 21/04/2015 | | | |
| **5** | M[(1,2,5,3,2),(1,3,7,3,4), (0,5,2,2,1), (1,3,0,1,2), (0,6,7,4,7)] | 148 | V |
| Disponível em: http://www.mundoeducacao.com/matematica/regra-chio-nos-calculos-dos-determinantes.htm - Acesso em 21/04/2015 | | | |

Multiplicação de uma matriz por escalar – Opção 2

Observação: Para o escalar, utilizamos como valor de escalar, o número 2.

| ORDEM | MATRIZ | RESULTADO | V/F |
|---|---|---|---|
| **1** | M[3] | 6 | V |
| **2** | M[(2,3),(4,5)] | M[(4,6),(8,10)] | V |
| **3** | M[(2,3,4), (5,8,3), (9,1,7)] | M[(4,6,8), (10,16,6), (18,2,14)] | V |
| **4** | M[(1,3,5,7), (2,4,6,8), (3,7,2,1), (6,2,1,4)] | M[(2,6,10,14), (4,8,12,16), (6,14,4,2), (12, 4, 2, 8)] | V |
| **5** | M[(2,3,7,4,6), (4,5,8,3,6), (5,2,6,7,5), (1,4,3,2,3), (2,1,7,1,4)] | M[(4,6,14,8,12), (8,10,16,6,12), (10,4,12,14,10), (2,8,6,4,6), (4,2,14,2,8)] | V |

Adição de duas matrizes – Opção 3

| ORDEM | MATRIZ 1 | MATRIZ 2 | RESULTADO | V/F |
|---|---|---|---|---|
| **1** | M[2] | M[5] | M[7] | V |
| **2** | M[(2,3), (4,5)] | M[(3,9),(2,1)] | M[(5,12),(6,6)] | V |
| **3** | M[(2,3,4), (5,6,7), (8,9,0)] | M[(6,7,4), (3,8,1), (2,5,9)] | M(8,10,8), (8,14,8), (10,14,9)] | V |
| **4** | M[(2,3,4,5), (5,6,7, 8), (8,9,0, 1), (1,4,2,0)] | M[(6,7,4,2), (3,8,1, 9), (2,5,9,3), (0,2,3,1)] | M[(8,10,8,7), (8,14,8,7), (10,14,9,4), | V |

| | | | | |
|---|---|---|---|---|
| | | | (1,6,6,1)] | |
| **5** | M[(2,4,6,8,10), (3,5,7,9,11), (1,3,5,7,9), (4,3,0,1,2), (0,6,8,4,5)] | M[(3,4,1,2,3), (1,3,2,4,5), (2,5,7,3,2), (4,1,4,7,4), (5,7,2,5,1)] | M[(5,8,7,10,13), (4,8,9,13,16), (3,8,12,10,11), (8,4,4,8,6,), (5,13,10,9,6)] | V |

Subtração de duas matrizes – Opção 4

| **ORDEM** | **MATRIZ 1** | **MATRIZ 2** | **RESULTADO** | **V/F** |
|---|---|---|---|---|
| **1** | M[2] | M[5] | M[-3] | V |
| **2** | M[(2,3), (4,5)] | M[(3,9),(2,1)] | M{[-1,-6],[2,4]} | V |
| **3** | M[(2,3,4), (5,6,7), (8,9,0)] | M[(6,7,4), (3,8,1), (2,5,9)] | M{[-4,-4,0], [2,-2,6], [6,4,-9]} | V |
| **4** | M[(2,3,4,5), (5,6,7, 8), (8,9,0, 1), (1,4,2,0)] | M[(6,7,4,2), (3,8,1, 9), (2,5,9,3), (0,2,3,1)] | M{[-4,-4,0,3], [2,-2,6,-1], [6,4,-9,-2], [1,2,-1,-1]} | V |
| **5** | M[(2,4,6,8,10), (3,5,7,9,11), (1,3,5,7,9), (4,3,0,1,2), (0,6,8,4,5)] | M[(3,4,1,2,3), (1,3,2,4,5), (2,5,7,3,2), (4,1,4,7,4), (5,7,2,5,1)] | M{[-1,0,5,6,7], [2,2,5,5,6], [-1,-2,-2,4,7], {0,2,-4,-6,-2], [-5,-V1,6,-1,4]} | V |

Igualdade de matrizes –Opção 5

| **ORDEM** | **MATRIZ 1** | **MATRIZ 2** | **RESULTADO** | **V/F** |
|---|---|---|---|---|
| **1** | M[2] | M[5] | FALSO | V |
| | M[2] | M[2] | VERDADEIRO | V |
| | | | | V |
| **2** | M[(2,3), (4,5)] | M[(3,9), (2,1)] | FALSO | V |
| | M[(2,3), (4,5)] | M[(2,3), (4,5)] | VERDADEIRO | V |
| **3** | M[(2,3,4), (5,6,7), (8,9,0)] | M[(6,7,4), (3,8,1), (2,5,9)] | FALSO | V |
| | M[(2,3,4), (5,6,7), (8,9,0)] | M[(2,3,4), (5,6,7), (8,9,0)] | VERDADEIRO | V |

| | | | | |
|---|---|---|---|---|
| | | | | V |
| **4** | M[(2,3,4,5), (5,6,7, 8), (8,9,0, 1), (1,4,2,0)] | M[(6,7,4,2), (3,8,1, 9), (2,5,9,3), (0,2,3,1)] | FALSO | V |
| | M[(2,3,4,5), (5,6,7, 8), (8,9,0, 1), (1,4,2,0)] | M[(2,3,4,5), (5,6,7, 8), (8,9,0, 1), (1,4,2,0)] | VERDADEIRO | V |
| **5** | M[(2,4,6,8,10), (3,5,7,9,11), (1,3,5,7,9), (4,3,0,1,2), (0,6,8,4,5)] | M[(3,4,1,2,3), (1,3,2,4,5), (2,5,7,3,2), (4,1,4,7,4), (5,7,2,5,1)] | FALSO | V |
| | M[(2,4,6,8,10), (3,5,7,9,11), (1,3,5,7,9), (4,3,0,1,2), (0,6,8,4,5)] | M[(2,4,6,8,10), (3,5,7,9,11), (1,3,5,7,9), (4,3,0,1,2), (0,6,8,4,5)] | VERDADEIRO | V |