23. [Sebesta, 2000] Projete um tipo de dado abstrato para uma abstração de matriz em uma linguagem que você conheça, incluindo operações de adição, de subtração e de multiplicação de matrizes.

```
package com.ybadoo.ensino.plp.modulo10.exercicio23;

public class Matriz
{
  /**
   * Valores da matriz
   */
  private double matriz[][];

  /**
   * Construtor da classe
   *
   * @param linhas quantidade de linhas da matriz
   * @param colunas quantidade de colunas da matriz
   * @throws IllegalArgumentException quantidade de linhas e/ou colunas invalido
   */
  public Matriz(int linhas, int colunas) throws IllegalArgumentException
  {
    // Validar a quantidade de linhas e de colunas da matriz desejada
    if((linhas > 0) && (colunas > 0))
    {
      // Instanciar a matriz
      this.matriz = new double[linhas][colunas];

      return;
    }

    throw new IllegalArgumentException("Quantidade de linhas e/ou colunas " +
                                                                    "inválido!");
  }

  /**
   * Retornar a quantidade de colunas da matriz
   *
   * @return quantidade de colunas da matriz
   */
  public int getColumnLength()
  {
    return this.matriz[0].length;
  }

  /**
   * Recuperar um elemento da matriz
   *
   * @param linha posicao do elemento na linha da matriz
   * @param coluna posicao do elemento na coluna da matriz
   * @return valor do elemento desejado
   * @throws ArrayIndexOutOfBoundsException tentativa de acesso a um elemento numa
   * posicao inexistente
   */
  public double getElement(int linha, int coluna) throws
                                                    ArrayIndexOutOfBoundsException
  {
    return this.matriz[linha][coluna];
  }
```

```
/**
 * Realizar a operacao de adicao entre duas matrizes
 *
 * @param mat matriz a ser adicionada
 * @throws IllegalArgumentException matrizes de tamanhos incompativeis para a
 * operacao de adicao
 */
public void add(Matriz mat) throws IllegalArgumentException
{
  // Validar a quantidade de linhas e colunas das duas matrizes
  if((getLineLength() == mat.getLineLength()) &&
     (getColumnLength() == mat.getColumnLength()))
  {
    // Realizar a operacao de adicao
    for(int linha = 0; linha < getLineLength(); linha++)
    {
      for(int coluna = 0; coluna < getColumnLength(); coluna++)
      {
        setElement(getElement(linha, coluna) + mat.getElement(linha, coluna),
                                                                    linha, coluna);
      }
    }

    return;
  }

  throw new IllegalArgumentException("Matrizes de tamanhos incompatíveis " +
                                                    "para a operação de adição!");
}

/**
 * Realizar a operacao de subtracao entre duas matrizes
 *
 * @param mat matriz a ser subtraida
 * @throws IllegalArgumentException matrizes de tamanhos incompativeis para a
 * operacao de subtracao
 */
public void minus(Matriz mat) throws IllegalArgumentException
{
  // Validar a quantidade de linhas e colunas das duas matrizes
  if((getLineLength() == mat.getLineLength()) &&
     (getColumnLength() == mat.getColumnLength()))
  {
    // Realizar a operacao de subtracao
    for(int linha = 0; linha < getLineLength(); linha++)
    {
      for(int coluna = 0; coluna < getColumnLength(); coluna++)
      {
        setElement(getElement(linha, coluna) - mat.getElement(linha, coluna),
                                                                    linha, coluna);
      }
    }

    return;
  }

  throw new IllegalArgumentException("Matrizes de tamanhos incompatíveis " +
                                                    "para a operação de subtração!");
}
```

```
/**
 * Realizar a operacao de multiplicacao entre duas matrizes
 *
 * @param mat matriz a ser multiplicada
 * @throws IllegalArgumentException matrizes de tamanhos incompativeis para a
 * operacao de multiplicacao
 */
public void multi(Matriz mat) throws IllegalArgumentException
{
  // Validar o tamanho das duas matrizes
  if(getColumnLength() == mat.getLineLength())
  {
    // Declarar e instanciar uma matriz auxiliar
    double aux[][] = new double[getLineLength()][mat.getColumnLength()];

    // Realizar a multiplicacao entre as duas matrizes
    for(int linha = 0; linha < getLineLength(); linha++)
    {
      for(int coluna = 0; coluna < mat.getColumnLength(); coluna++)
      {
        aux[linha][coluna] = 0.0;

        for(int i = 0; i < getColumnLength(); i++)
        {
          aux[linha][coluna] = aux[linha][coluna] +
                               getElement(linha, i) * mat.getElement(i, coluna);
        }
      }
    }

    // Armazenar o resultado
    this.matriz = aux;

    return;
  }

  throw new IllegalArgumentException("Matrizes de tamanhos incompatíveis " +
                                     "para a operação de subtração!");
}

/**
 * Armazenar um elemento na matriz
 *
 * @param element valor do elemento a ser armazenado
 * @param linha posicao do elemento na linha da matriz
 * @param coluna posicao do elemento na coluna da matriz
 * @throws ArrayIndexOutOfBoundsException tentativa de armazenar um elemento
 * numa posicao inexistente
 */
public void setElement(double element, int linha, int coluna) throws
                                                   ArrayIndexOutOfBoundsException
{
  this.matriz[linha][coluna] = element;
}
```

```
  /* (non-Javadoc)
   * @see java.lang.Object#toString()
   */
  public String toString()
  {
    StringBuffer out = new StringBuffer();

    for(int linha = 0; linha < getLineLength(); linha++)
    {
      for(int coluna = 0; coluna < getColumnLength(); coluna++)
      {
        out.append(getElement(linha, coluna)).append("\t");
      }

      out.append("\n");
    }

    return out.toString();
  }

  /**
   * Retornar a quantidade de linhas da matriz
   *
   * @return quantidade de linhas da matriz
   */
  public int getLineLength()
  {
    return this.matriz.length;
  }
}
```

```java
package com.ybadoo.ensino.plp.modulo10.exercicio23;

/**
 * Aplicacao para testar a classe Matriz
 */
public class MatrizTest
{
  /**
   * Funcao principal do java
   *
   * @param args argumentos passados via linha de comando
   */
  public static void main(String[] args)
  {
    // Declarar e inicializar a matriz 1
    Matriz mat1 = new Matriz(3, 3);
    for(int i = 0, c = 1; i < 3; i++)
    {
      for(int j = 0; j < 3; j++, c++)
      {
        mat1.setElement(c, i, j);
      }
    }

    System.out.println("Matriz 01\n" + mat1);

    // Declarar e inicializar a matriz 2
    Matriz mat2 = new Matriz(3, 3);
    for(int i = 0, c = 9; i < 3; i++)
    {
      for(int j = 0; j < 3; j++, c--)
      {
        mat2.setElement(c, i, j);
      }
    }

    System.out.println("Matriz 02\n" + mat2);


    // Somar a matriz 01 com a matriz 02
    mat1.add(mat2);

    System.out.println("Matriz 01 + Matriz02\n" + mat1);
  }
}
```