Министерство науки и высшего образования РФ

Федеральное государственное бюджетное

образовательное учреждение высшего образования

ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

Отчет по лабораторной работе №5

**Разработка пользовательского интерфейса с использованием технологии swing**

дисциплина «Технология разработки программного обеспечения»

|  |  |
| --- | --- |
| Выполнил: студент группы ИВТм-1301 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Савин Д.А. / |
|  |  |
| Проверил: доцент кафедры ЭВМ | \_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Чистяков Г.А. / |

Киров 2022

**Цель**: Целью работы является получение навыков разработки графического пользовательского интерфейса с применением технологии Swing.

**Задачи**:

1. Выбрать и согласовать с преподавателем задачу, для решения которой может быть использована библиотека, разработанная в ходе предыдущей лабораторной работы.
2. Разработать программу для решения выбранной задачи (взаимодействие с пользователем должно осуществляться с применением пользовательского интерфейса).

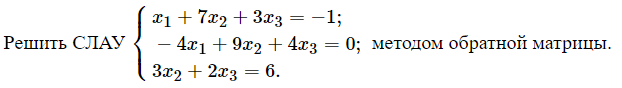
Код:

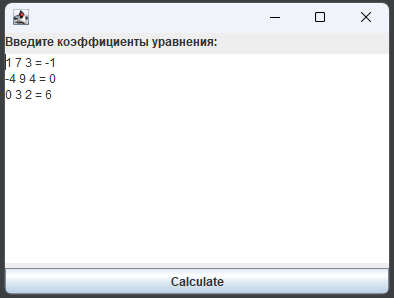
|  |
| --- |
| **import** **javax.swing.\***;  **import** **javax.swing.event.DocumentEvent**;  **import** **javax.swing.event.DocumentListener**;  **import** **java.awt.event.ActionEvent**;  **import** **java.awt.event.ActionListener**;  **import** **java.util.ArrayList**;  **import** **java.util.Arrays**;  **import** **java.util.List**;  **import** **java.util.Objects**;  **import** **static** java.lang.System.out;  **public** **class** **solve\_slau** **extends** JFrame  {  **private** JPanel panel1;  **private** JTextArea textArea1;  **private** JButton button1;  **public** **solve\_slau**() {  List<Double[]> mA = **new** ArrayList<>();  List<**double**[]> mB = **new** ArrayList<>();  setContentPane(panel1);  setVisible(**true**);  setSize(**400**,**300**);  setLocation(**500**,**400**);  textArea1.setText("1 7 3 = -1\n-4 9 4 = 0\n0 3 2 = 6");  setDefaultCloseOperation(WindowConstants.EXIT\_ON\_CLOSE);  button1.addActionListener(**new** ActionListener()  {  **@Override**  **public** **void** **actionPerformed**(ActionEvent e)  {  **if** (!Objects.equals(textArea1.getText(), ""))  {  String[] secondMatrix = textArea1.getText().split("\n");  **for** (String matrix : secondMatrix)  {  String[] a = matrix.split("=");  mB.add(**new** **double**[]{Double.parseDouble(a[**1**])});  List<Double> \_ar = **new** ArrayList<Double>();  String[] ar = a[**0**].split(" ");  **for** (String s : ar)  {  \_ar.add(Double.parseDouble(s));  }  mA.add(\_ar.toArray(**new** Double[**0**]));  }  }  **float**[][] matrixA = **new** **float**[mA.size()][mA.size()];  **float**[][] matrixB = **new** **float**[mB.size()][**1**];  **for** (**int** i=**0**; i<mA.size(); i++)  {  **for** (**int** j=**0**; j<mA.size(); j++)  {  matrixA[i][j]=Float.parseFloat(String.valueOf(mA.get(i)[j]));  }  }  **for** (**int** i=**0**; i<mB.size(); i++)  {  **for** (**int** j=**0**; j<**1**; j++)  {  matrixB[i][j]=Float.parseFloat(String.valueOf(mB.get(i)[j]));  }  }  **try**  {  **double** det = MatrixLibrary.GetDeterminant(matrixA);  **float**[][] invMat = MatrixLibrary.Jordan\_Gauss(matrixA, **true**);  **float**[][] matrixC = MatrixLibrary.MultiplyMatrix(invMat, matrixB);  String res = showMatrix(matrixC);  textArea1.append("\n\ndeterminant=" + det + "\nResult:\n"+res);  button1.setEnabled(**false**);  mA.clear();  mB.clear();  } **catch** (Exception ex)  {  **throw** **new** **RuntimeException**(ex);  }  }  });  textArea1.getDocument().addDocumentListener(**new** DocumentListener()  {  **@Override**  **public** **void** **removeUpdate**(DocumentEvent e) {  button1.setEnabled(**true**);  }  **@Override**  **public** **void** **insertUpdate**(DocumentEvent e) {  button1.setEnabled(**true**);  }  **@Override**  **public** **void** **changedUpdate**(DocumentEvent arg0) {  button1.setEnabled(**true**);  }  });  }  **public** **static** **void** **main**(String[] args) {  **new** **solve\_slau**();  }  **private** **static** String **showMatrix**(**float** [][]matrix)  {  StringBuilder res = **new** StringBuilder();  **for** (**float**[] doubles : matrix)  {  **for** (**int** j = **0**; j < matrix[**0**].length; j++)  {  res.append(doubles[j]).append(" ");  }  res.append("\n");  }  **return** res.toString();  }  **private** **void** **createUIComponents**() {}  } |

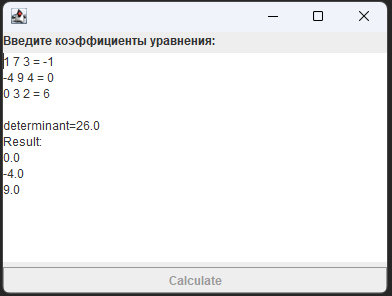
Доступные методы:

1. Нахождение обратной матрицы с помощью метода алгебраических дополнений.
2. Нахождение обратной матрицы с помощью метода Гаусса-Жордано.
3. Нахождение детерминанта матрицы.
4. Транспонирование матрицы.
5. Перемножение матриц.

Сценарий использования:







Вывод:

Благодаря Swing, можно реализовать интерфейсную часть, для удобного взаимодействия с программой.