Министерство науки и высшего образования РФ

Пензенский государственный университет

Кафедра «Вычислительная техника»

ОТЧЁТ

по лабораторной работе № 5

по курсу «Разработка кроссплатформенных приложений»

на тему «Многопоточность в Java»

Выполнили ст. гр. 22ВВВ1:

Лосяков Г.С.

Осмаев М-А.Х.

Чубаров И.В.

Принял:

Юрова О.В.

Пенза 2025

**Цель работы:** научиться создавать многопоточные приложения c использованием стандартных средств языка Java.

Задание на лабораторную работу

Модифицировать приложение из предыдущей лабораторной работы, реализовав вычисление определенного интеграла в нескольких дополнительных потоках (число потоков определяется номером варианта), снимая нагрузку с основного потока и предотвращая "подвисание" графического интерфейса. Варианты с номерами до 5 включительно реализуют многопоточность путем наследования от класса Thread, остальные реализуют интерфейс Runnable.

Многопоточность реализуется в классе RecIntegral в методах startCalculation и computePartialIntegral.

В первом методе происходит разделение отрезка интегрирования на 6 частей для каждой нити. Код представлен на рис. 1.

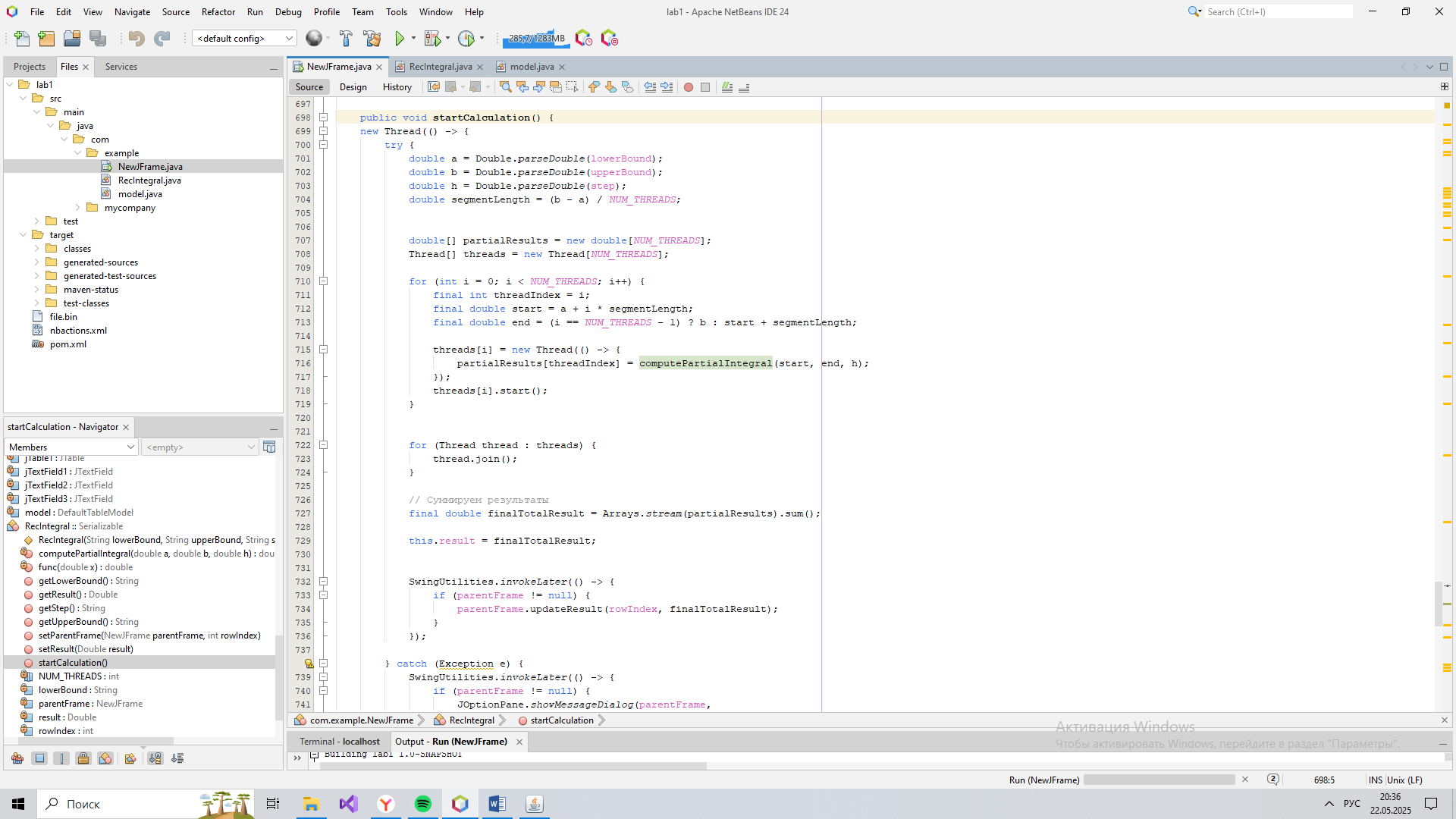


Рис. 1 – Код метода startCalculation

В методе computePartialIntegral происходят сами вычисления. Этот метод был скопирован из предыдущей лабораторной работы. Его код представлен на рис. 2.

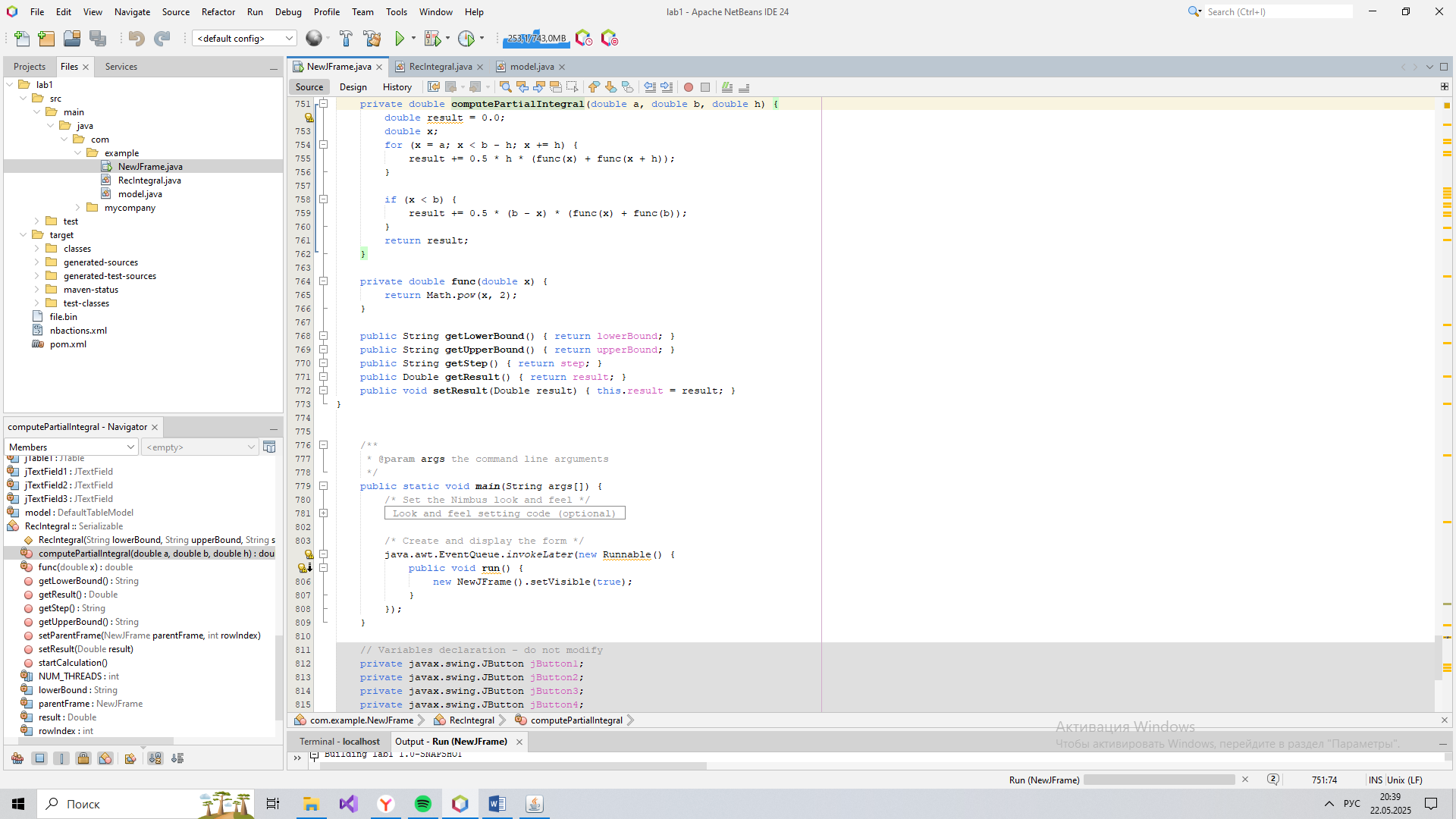


Рис. 2 – Код метода computePartialIntegral

Результат работы программы представлен на рис. 3.

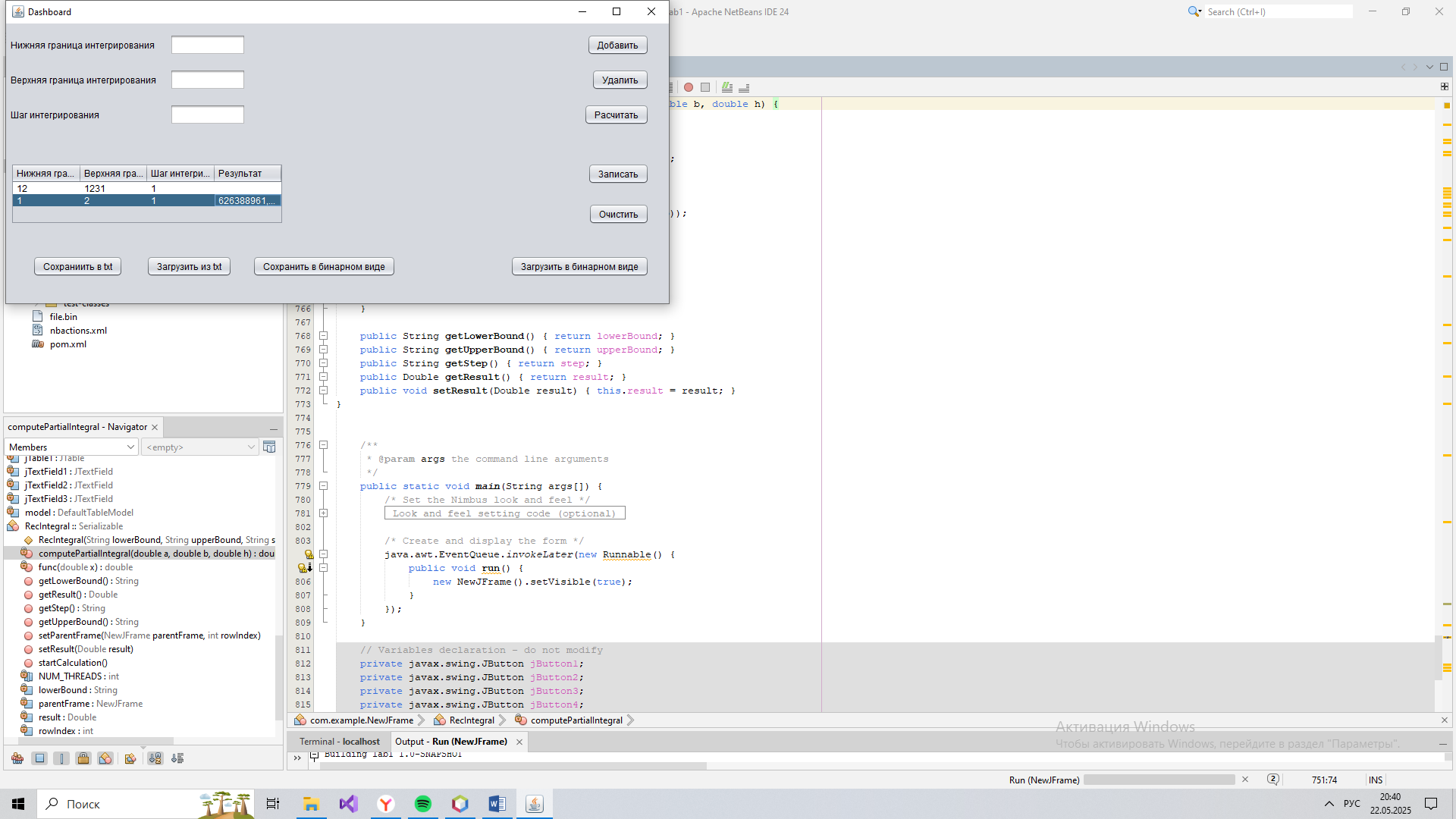


Рис. 3 – Результат работы программы

Приложение 1

Листинг

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template

\*/

package com.example;

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.BufferedInputStream;

import java.io.BufferedOutputStream;

import java.io.BufferedReader;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.io.PrintWriter;

import java.io.Serializable;

import java.lang.Math;

import java.util.Arrays;

import java.util.LinkedList;

import java.util.Vector;

import java.util.concurrent.ExecutorService;

import java.util.concurrent.Executors;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*\*

\*

\* @author 79374

\*/

public class NewJFrame extends javax.swing.JFrame {

private LinkedList<RecIntegral> earlBio = new LinkedList<>();

private ExecutorService executorService;

private DefaultTableModel model;

public NewJFrame() {

initComponents();

model = (DefaultTableModel) jTable1.getModel();

executorService = Executors.newFixedThreadPool(6); // Пул из 6 потоков

}

public void updateResult(int rowIndex, double result) {

// Проверяем, что индекс строки корректен

if (rowIndex >= 0 && rowIndex < model.getRowCount()) {

model.setValueAt(String.format("%.6f", result), rowIndex, 3);

jButton3.setEnabled(true);

}

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jPanel1 = new javax.swing.JPanel();

jScrollPane1 = new javax.swing.JScrollPane();

jTable1 = new javax.swing.JTable();

jButton2 = new javax.swing.JButton();

jButton3 = new javax.swing.JButton();

jTextField1 = new javax.swing.JTextField();

jTextField2 = new javax.swing.JTextField();

jTextField3 = new javax.swing.JTextField();

jLabel1 = new javax.swing.JLabel();

jLabel2 = new javax.swing.JLabel();

jLabel3 = new javax.swing.JLabel();

jButton1 = new javax.swing.JButton();

jButton4 = new javax.swing.JButton();

jButton5 = new javax.swing.JButton();

jButton6 = new javax.swing.JButton();

jButton7 = new javax.swing.JButton();

jButton8 = new javax.swing.JButton();

jButton9 = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

setTitle("Dashboard");

setMinimumSize(new java.awt.Dimension(400, 300));

addWindowListener(new java.awt.event.WindowAdapter() {

public void windowClosing(java.awt.event.WindowEvent evt) {

formWindowClosing(evt);

}

public void windowOpened(java.awt.event.WindowEvent evt) {

formWindowOpened(evt);

}

});

jTable1.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

"Нижняя граница интегрирования", "Верхняя граница интегрирования", "Шаг интегрирования", "Результат"

}

) {

boolean[] canEdit = new boolean [] {

false, true, true, false

};

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

jScrollPane1.setViewportView(jTable1);

jButton2.setText("Удалить");

jButton2.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton2ActionPerformed(evt);

}

});

jButton3.setText("Расчитать");

jButton3.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton3ActionPerformed(evt);

}

});

jLabel1.setText("Нижняя граница интегрирования");

jLabel2.setText("Верхняя граница интегрирования");

jLabel3.setText("Шаг интегрирования");

jButton1.setText("Добавить");

jButton1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton1ActionPerformed(evt);

}

});

jButton4.setText("Записать");

jButton4.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton4ActionPerformed(evt);

}

});

jButton5.setText("Очистить");

jButton5.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton5ActionPerformed(evt);

}

});

jButton6.setText("Сохраниить в txt");

jButton6.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton6ActionPerformed(evt);

}

});

jButton7.setText("Загрузить из txt");

jButton7.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton7ActionPerformed(evt);

}

});

jButton8.setText("Сохранить в бинарном виде");

jButton8.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton8ActionPerformed(evt);

}

});

jButton9.setText("Загрузить в бинарном виде");

jButton9.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton9ActionPerformed(evt);

}

});

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);

jPanel1.setLayout(jPanel1Layout);

jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addGap(29, 29, 29)

.addComponent(jButton6)

.addGap(31, 31, 31)

.addComponent(jButton7)

.addGap(27, 27, 27)

.addComponent(jButton8)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 151, Short.MAX\_VALUE)

.addComponent(jButton9))

.addGroup(jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel1)

.addComponent(jLabel2)

.addComponent(jLabel3))

.addGap(18, 18, 18)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(jTextField3, javax.swing.GroupLayout.DEFAULT\_SIZE, 100, Short.MAX\_VALUE)

.addComponent(jTextField2, javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(jTextField1, javax.swing.GroupLayout.Alignment.TRAILING)))

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 360, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jButton1, javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(jButton3, javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(jButton4, javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(jButton5, javax.swing.GroupLayout.Alignment.TRAILING))))

.addGap(20, 20, 20))

);

jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel1Layout.createSequentialGroup()

.addGap(14, 14, 14)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel1)

.addComponent(jButton1))

.addGap(18, 18, 18)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jTextField2, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel2))

.addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING))

.addGap(18, 18, 18)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jTextField3, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel3)

.addComponent(jButton3))

.addGap(50, 50, 50)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 81, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGroup(jPanel1Layout.createSequentialGroup()

.addComponent(jButton4)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jButton5)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 41, Short.MAX\_VALUE)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jButton6)

.addComponent(jButton7)

.addComponent(jButton9)

.addComponent(jButton8))

.addGap(35, 35, 35))

);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(0, 0, Short.MAX\_VALUE))

);

pack();

}// </editor-fold>

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

try {

addRow();

} catch (RecIntegralException ex) {

Logger.getLogger(NewJFrame.class.getName()).log(Level.SEVERE, null, ex);

}

}

private void formWindowOpened(java.awt.event.WindowEvent evt) {

try{

FileInputStream file = new FileInputStream("file.bin");

ObjectInputStream input = new ObjectInputStream(file);

Vector<Vector> jTable1 = (Vector<Vector>)input.readObject();

input.close();

file.close();

}

catch(Exception ex){

ex.printStackTrace();

}

}

private void formWindowClosing(java.awt.event.WindowEvent evt) {

DefaultTableModel model = (DefaultTableModel) jTable1.getModel();

Vector<Vector> jTable1 = model.getDataVector();

try{

FileOutputStream file = new FileOutputStream("file.bin");

ObjectOutputStream output = new ObjectOutputStream(file);

output.writeObject(jTable1);

output.close();

file.close();

}

catch(Exception ex){

ex.printStackTrace();

}

}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

deleteRow();

}

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

countInt();

}

private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {

DefaultTableModel model = (DefaultTableModel) jTable1.getModel();

for (RecIntegral record : earlBio) {

model.addRow(new Object[]{

record.getLowerBound(),

record.getUpperBound(),

record.getStep(),

record.getResult() != null ? record.getResult() : ""

});

}

}

private void refreshTable() {

DefaultTableModel model = (DefaultTableModel) jTable1.getModel();

model.setRowCount(0);

for (RecIntegral record : earlBio) {

model.addRow(new Object[]{

record.getLowerBound(),

record.getUpperBound(),

record.getStep(),

record.getResult()

});

}

}

private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

deleteRows();

}

private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {

saveToTextFile();

}

private void jButton7ActionPerformed(java.awt.event.ActionEvent evt) {

loadFromTextFile();

}

private void jButton8ActionPerformed(java.awt.event.ActionEvent evt) {

saveToBinaryFile();

}

private void jButton9ActionPerformed(java.awt.event.ActionEvent evt) {

loadFromBinaryFile();

}

private void deleteRows(){

DefaultTableModel model = (DefaultTableModel) jTable1.getModel();

model.setRowCount(0);

}

private void addRow() throws RecIntegralException {

String lowerBound = jTextField1.getText().trim();

String upperBound = jTextField2.getText().trim();

String step = jTextField3.getText().trim();

if (!lowerBound.isEmpty() && !upperBound.isEmpty() && !step.isEmpty()) {

try {

RecIntegral newRecord = new RecIntegral(

lowerBound,

upperBound,

step,

this,

model.getRowCount()

);

earlBio.add(newRecord);

model.addRow(new Object[]{

newRecord.getLowerBound(),

newRecord.getUpperBound(),

newRecord.getStep(),

""

});

// Очищаем поля ввода

jTextField1.setText("");

jTextField2.setText("");

jTextField3.setText("");

} catch (RecIntegralException e) {

JOptionPane.showMessageDialog(this,

e.getMessage(),

"Ошибка ввода",

JOptionPane.ERROR\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this,

"Пожалуйста, заполните все поля.",

"Ошибка",

JOptionPane.WARNING\_MESSAGE);

}

}

private void deleteRow() {

int row = jTable1.getSelectedRow();

if (row < 0) {

JOptionPane.showMessageDialog(this, "В таблице нет строк. Пожалуйста, создайте их.");

} else {

DefaultTableModel model = (DefaultTableModel) jTable1.getModel();

earlBio.remove(row);

model.removeRow(row);

}

}

private void countInt() {

int selectedRow = jTable1.getSelectedRow();

if (selectedRow != -1 && selectedRow < earlBio.size()) {

try {

RecIntegral record = earlBio.get(selectedRow);

jButton3.setEnabled(false);

record.startCalculation();

} catch (Exception e) {

e.printStackTrace();

jButton3.setEnabled(true);

JOptionPane.showMessageDialog(this, "Ошибка: " + e.getMessage());

}

} else {

JOptionPane.showMessageDialog(this, "Пожалуйста, выберите корректную строку для вычисления");

}

}

public void enableCalculateButton() {

jButton3.setEnabled(true);

}

/\*private double computeIntegral(double lowerBound,double upperBound, double step){

double result = 0.0;

double x;

for (x = lowerBound; x < upperBound - step; x += step) {

result += 0.5 \* step \* (func(x) + func(x + step)); // Пример функции f(x) = x^2

}

result += 0.5 \* step \* (func(x) + func(upperBound - x));

return result;

}

private double func(double x){

return Math.pow(x, 2);

}\*/

private void saveToTextFile() {

JFileChooser fileChooser = new JFileChooser();

fileChooser.setDialogTitle("Сохранить как текстовый файл");

int userSelection = fileChooser.showSaveDialog(this);

if (userSelection == JFileChooser.APPROVE\_OPTION) {

File fileToSave = fileChooser.getSelectedFile();

try (PrintWriter writer = new PrintWriter(new FileWriter(fileToSave))) {

DefaultTableModel model = (DefaultTableModel) jTable1.getModel();

for (int i = 0; i < model.getRowCount(); i++) {

for (int j = 0; j < model.getColumnCount(); j++) {

Object value = model.getValueAt(i, j);

writer.print(value != null ? value.toString() : "");

if (j < model.getColumnCount() - 1) {

writer.print(",");

}

}

writer.println();

}

JOptionPane.showMessageDialog(this, "Данные успешно сохранены в текстовый файл.");

} catch (IOException e) {

JOptionPane.showMessageDialog(this, "Ошибка при сохранении файла: " + e.getMessage(),

"Ошибка", JOptionPane.ERROR\_MESSAGE);

}

}

}

private void loadFromTextFile() {

JFileChooser fileChooser = new JFileChooser();

fileChooser.setDialogTitle("Загрузить из текстового файла");

int userSelection = fileChooser.showOpenDialog(this);

if (userSelection == JFileChooser.APPROVE\_OPTION) {

File fileToLoad = fileChooser.getSelectedFile();

try (BufferedReader reader = new BufferedReader(new FileReader(fileToLoad))) {

DefaultTableModel model = (DefaultTableModel) jTable1.getModel();

model.setRowCount(0); // Очищаем таблицу перед загрузкой

String line;

while ((line = reader.readLine()) != null) {

// Разделяем строку с учетом пустых значений между запятыми

String[] parts = line.split(",(?=([^\"]\*\"[^\"]\*\")\*[^\"]\*$)", -1);

if (parts.length == model.getColumnCount()) {

// Заменяем пустые строки на null или оставляем как есть

Object[] rowData = new Object[parts.length];

for (int i = 0; i < parts.length; i++) {

rowData[i] = parts[i].isEmpty() ? null : parts[i];

}

model.addRow(rowData);

}

}

JOptionPane.showMessageDialog(this, "Данные успешно загружены из текстового файла.");

} catch (IOException e) {

JOptionPane.showMessageDialog(this, "Ошибка при загрузке файла: " + e.getMessage(),

"Ошибка", JOptionPane.ERROR\_MESSAGE);

}

}

}

private void saveToBinaryFile() {

JFileChooser fileChooser = new JFileChooser();

fileChooser.setDialogTitle("Сохранить как бинарный файл");

int userSelection = fileChooser.showSaveDialog(this);

if (userSelection == JFileChooser.APPROVE\_OPTION) {

File fileToSave = fileChooser.getSelectedFile();

try (ObjectOutputStream oos = new ObjectOutputStream(

new BufferedOutputStream(new FileOutputStream(fileToSave)))) {

// Создаем список для сериализации (только данные, без Swing-компонентов)

LinkedList<Object[]> dataToSave = new LinkedList<>();

for (RecIntegral record : earlBio) {

dataToSave.add(new Object[]{

record.getLowerBound(),

record.getUpperBound(),

record.getStep(),

record.getResult()

});

}

oos.writeObject(dataToSave);

JOptionPane.showMessageDialog(this,

"Данные успешно сохранены в бинарный файл.",

"Сохранение", JOptionPane.INFORMATION\_MESSAGE);

} catch (IOException e) {

JOptionPane.showMessageDialog(this,

"Ошибка при сохранении файла: " + e.getMessage(),

"Ошибка", JOptionPane.ERROR\_MESSAGE);

e.printStackTrace();

}

}

}

private void loadFromBinaryFile() {

JFileChooser fileChooser = new JFileChooser();

fileChooser.setDialogTitle("Загрузить из бинарного файла");

int userSelection = fileChooser.showOpenDialog(this);

if (userSelection == JFileChooser.APPROVE\_OPTION) {

File fileToLoad = fileChooser.getSelectedFile();

try (ObjectInputStream ois = new ObjectInputStream(

new BufferedInputStream(new FileInputStream(fileToLoad)))) {

earlBio.clear();

model.setRowCount(0);

LinkedList<Object[]> loadedData = (LinkedList<Object[]>) ois.readObject();

for (int i = 0; i < loadedData.size(); i++) {

Object[] row = loadedData.get(i);

try {

RecIntegral record = new RecIntegral(

(String)row[0],

(String)row[1],

(String)row[2],

this,

i

);

if (row[3] != null) {

record.setResult((Double)row[3]);

}

earlBio.add(record);

model.addRow(new Object[]{

record.getLowerBound(),

record.getUpperBound(),

record.getStep(),

record.getResult() != null ? record.getResult() : ""

});

record.setParentFrame(this, i);

} catch (RecIntegralException e) {

JOptionPane.showMessageDialog(this,

"Ошибка при загрузке записи: " + e.getMessage(),

"Ошибка",

JOptionPane.ERROR\_MESSAGE);

}

}

JOptionPane.showMessageDialog(this,

"Данные успешно загружены из бинарного файла.",

"Загрузка",

JOptionPane.INFORMATION\_MESSAGE);

} catch (IOException | ClassNotFoundException e) {

JOptionPane.showMessageDialog(this,

"Ошибка при загрузке файла: " + e.getMessage(),

"Ошибка",

JOptionPane.ERROR\_MESSAGE);

}

}

}

private Vector<String> getColumnIdentifiers(DefaultTableModel model) {

Vector<String> columnIdentifiers = new Vector<>();

for (int i = 0; i < model.getColumnCount(); i++) {

columnIdentifiers.add(model.getColumnName(i));

}

return columnIdentifiers;

}

class RecIntegralException extends Exception {

public RecIntegralException(String message) {

super(message);

}

}

public class RecIntegral implements Serializable {

private final String lowerBound;

private final String upperBound;

private final String step;

private Double result;

private transient NewJFrame parentFrame;

private transient int rowIndex;

private static final int NUM\_THREADS = 6;

public void setParentFrame(NewJFrame parentFrame, int rowIndex) {

this.parentFrame = parentFrame;

this.rowIndex = rowIndex;

}

public RecIntegral(String lowerBound, String upperBound, String step, NewJFrame parentFrame, int rowIndex)

throws RecIntegralException {

double lower;

double upper;

double stepValue;

try {

lower = Double.parseDouble(lowerBound);

upper = Double.parseDouble(upperBound);

stepValue = Double.parseDouble(step);

} catch (NumberFormatException e) {

throw new RecIntegralException("Некорректный числовой формат");

}

if (lower < 0.000001 || lower > 1000000 || upper < 0.000001 || upper > 1000000 ||

stepValue < 0.000001 || stepValue > 1000000) {

throw new RecIntegralException("Значения должны быть в диапазоне от 0,000001 до 1000000");

}

if (upper <= lower) {

throw new RecIntegralException("Верхний предел должен быть больше нижнего");

}

if (stepValue > (upper - lower)) {

throw new RecIntegralException("Шаг должен быть меньше интервала между пределами");

}

this.lowerBound = lowerBound;

this.upperBound = upperBound;

this.step = step;

this.parentFrame = parentFrame;

this.rowIndex = rowIndex;

}

public void startCalculation() {

new Thread(() -> {

try {

double a = Double.parseDouble(lowerBound);

double b = Double.parseDouble(upperBound);

double h = Double.parseDouble(step);

double segmentLength = (b - a) / NUM\_THREADS;

double[] partialResults = new double[NUM\_THREADS];

Thread[] threads = new Thread[NUM\_THREADS];

for (int i = 0; i < NUM\_THREADS; i++) {

final int threadIndex = i;

final double start = a + i \* segmentLength;

final double end = (i == NUM\_THREADS - 1) ? b : start + segmentLength;

threads[i] = new Thread(() -> {

partialResults[threadIndex] = computePartialIntegral(start, end, h);

});

threads[i].start();

}

for (Thread thread : threads) {

thread.join();

}

// Суммируем результаты

final double finalTotalResult = Arrays.stream(partialResults).sum();

this.result = finalTotalResult;

SwingUtilities.invokeLater(() -> {

if (parentFrame != null) {

parentFrame.updateResult(rowIndex, finalTotalResult);

}

});

} catch (Exception e) {

SwingUtilities.invokeLater(() -> {

if (parentFrame != null) {

JOptionPane.showMessageDialog(parentFrame,

"Ошибка при вычислении: " + e.getMessage(),

"Ошибка", JOptionPane.ERROR\_MESSAGE);

parentFrame.enableCalculateButton();

}

});

}

}).start();

}

private double computePartialIntegral(double a, double b, double h) {

double result = 0.0;

double x;

for (x = a; x < b - h; x += h) {

result += 0.5 \* h \* (func(x) + func(x + h));

}

if (x < b) {

result += 0.5 \* (b - x) \* (func(x) + func(b));

}

return result;

}

private double func(double x) {

return Math.pow(x, 2);

}

public String getLowerBound() { return lowerBound; }

public String getUpperBound() { return upperBound; }

public String getStep() { return step; }

public Double getResult() { return result; }

public void setResult(Double result) { this.result = result; }

}

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new NewJFrame().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton jButton1;

private javax.swing.JButton jButton2;

private javax.swing.JButton jButton3;

private javax.swing.JButton jButton4;

private javax.swing.JButton jButton5;

private javax.swing.JButton jButton6;

private javax.swing.JButton jButton7;

private javax.swing.JButton jButton8;

private javax.swing.JButton jButton9;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JPanel jPanel1;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JTable jTable1;

private javax.swing.JTextField jTextField1;

private javax.swing.JTextField jTextField2;

private javax.swing.JTextField jTextField3;

// End of variables declaration

}