Министерство науки и высшего образования Российской Федерации  
Пензенский государственный университет  
Кафедра вычислительная техника

**ОТЧЕТ**

по лабораторной работе №4

по дисциплине «Программирование на языке Java»

на тему «Работа с файлами»

Выполнили:

студенты группы 22ВВП1

Хоссейни Нежад С.А.С.М.

Захаров А. С.

Сергунов М. Р.

Приняли:

Юрова О.В.

Карамышева Н.С.

Пенза 2025

**Название**

Работа с файлами

**Цель работы**

Изучить работу с файлами и механизмы сериализации данных.

**Лабораторное задание**

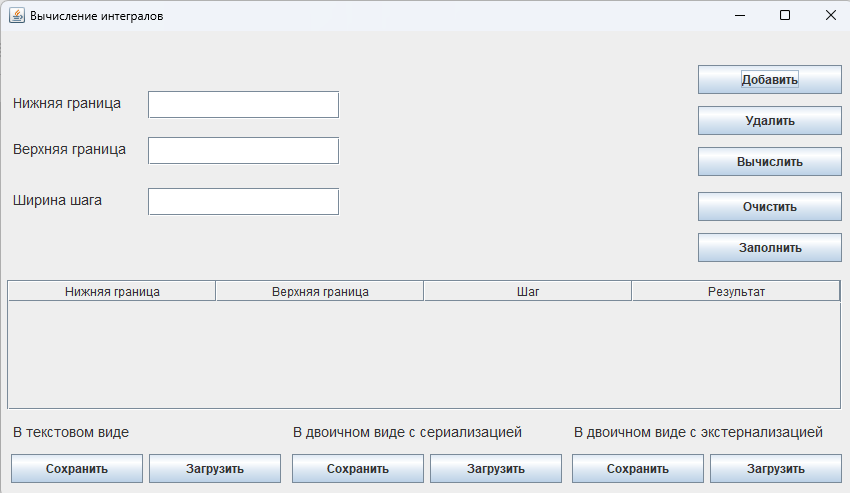
Модифицировать приложение из предыдущей лабораторной работы, реализовав сохранение в файл и загрузку данных из файла. Предусмотреть сохранение данных, как в текстовом виде, так и в двоичном (с использованием механизма сериализации). Для этого нужно добавить 4 кнопки для сохранения и загрузки в текстовом и двоичном виде соответственно. Кроме того, в программе нужно предусмотреть использование стандартного диалога открытия файла (JFileChooser). Оформление лабораторной работы должно быть выполнено в соответствии с требованиями, приведенными в Приложении 2.

Вариант 6:



**Ход работы**

**Добавили в интерфейс 6 новые кнопки**

****

**Обновили класс RecIntegral**

package lab1;

import java.io.Serializable;

/\*\*

\* Класс для хранения данных о вычислении интеграла.

\* Реализует интерфейс Serializable для поддержки сериализации.

\*/

public class RecIntegral implements Serializable {

private static final long serialVersionUID = 1L;

private double widthLim;

private double lowLim;

private double upLim;

private double resIntegral;

/\*\*

\* Конструктор для создания объекта RecIntegral.

\*

\* @param lowLim Нижняя граница.

\* @param upLim Верхняя граница.

\* @param widthLim Ширина шага.

\* @param resIntegral Результат вычисления интеграла.

\*/

public RecIntegral(double lowLim, double upLim, double widthLim, double resIntegral) {

this.widthLim = widthLim;

this.lowLim = lowLim;

this.upLim = upLim;

this.resIntegral = resIntegral;

}

/\*\*

\* Возвращает нижнюю границу.

\*

\* @return Нижняя граница.

\*/

public double getLowLim() {

return lowLim;

}

/\*\*

\* Устанавливает нижнюю границу.

\*

\* @param lowLim Нижняя граница.

\*/

public void setLowLim(double lowLim) {

this.lowLim = lowLim;

}

/\*\*

\* Возвращает верхнюю границу.

\*

\* @return Верхняя граница.

\*/

public double getUpLim() {

return upLim;

}

/\*\*

\* Устанавливает верхнюю границу.

\*

\* @param upLim Верхняя граница.

\*/

public void setUpLim(double upLim) {

this.upLim = upLim;

}

/\*\*

\* Возвращает ширину шага.

\*

\* @return Ширина шага.

\*/

public double getWidthLim() {

return widthLim;

}

/\*\*

\* Устанавливает ширину шага.

\*

\* @param widthLim Ширина шага.

\*/

public void setWidthLim(double widthLim) {

this.widthLim = widthLim;

}

/\*\*

\* Возвращает результат вычисления интеграла.

\*

\* @return Результат вычисления интеграла.

\*/

public double getResIntegral() {

return resIntegral;

}

/\*\*

\* Устанавливает результат вычисления интеграла.

\*

\* @param resIntegral Результат вычисления интеграла.

\*/

public void setResIntegral(double resIntegral) {

this.resIntegral = resIntegral;

}

@Override

public String toString() {

return "RecIntegral{" +

"lowLim=" + lowLim +

", upLim=" + upLim +

", widthLim=" + widthLim +

", resIntegral=" + resIntegral +

'}';

}

}

**Добавили класс SavedState**

package lab1;

import java.io.Serializable;

import java.util.ArrayList;

import java.util.LinkedList;

/\*\*

\* Класс для хранения состояния приложения, включающего два списка объектов RecIntegral.

\* Реализует интерфейс Serializable для поддержки сериализации.

\*/

public class SavedState implements Serializable {

private static final long serialVersionUID = 2L;

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

private ArrayList<RecIntegral> arrTableData = new ArrayList<>();

/\*\*

\* Конструктор для создания объекта SavedState.

\*

\* @param listRecIntegral Список RecIntegral для хранения.

\* @param arrTableData Список данных таблицы для хранения.

\*/

public SavedState(LinkedList<RecIntegral> listRecIntegral, ArrayList<RecIntegral> arrTableData) {

this.listRecIntegral = new LinkedList<>(listRecIntegral);

this.arrTableData = new ArrayList<>(arrTableData);

}

/\*\*

\* Возвращает копию списка RecIntegral вне таблицы.

\*

\* @return Копия списка RecIntegral вне таблицы.

\*/

public LinkedList<RecIntegral> getListRecIntegral() {

return new LinkedList<>(listRecIntegral);

}

/\*\*

\* Возвращает копию списка данных таблицы.

\*

\* @return Копия списка данных таблицы.

\*/

public ArrayList<RecIntegral> getArrListTableData() {

return new ArrayList<>(arrTableData);

}

}

**Добавили функционал для работы в текстовом виде**

**Кнопка “Сохранить”**

private void bSaveObjectTextFormatMouseClicked(java.awt.event.MouseEvent evt) {

// TODO add your handling code here:

File file = getPathTXTFileToSaved();

if (file == null) return;

ArrayList<RecIntegral> arrTableData = getDataArrListFromTable();

SavedState state = new SavedState(listRecIntegral, arrTableData);

saveToTextFile(file, state);

}

**Кнопка “Загрузить”**

private void bLoadObjectTextFormatMouseClicked(java.awt.event.MouseEvent evt) {

// TODO add your handling code here:

File file = getPathTXTFileToLoad();

if (file == null) return;

SavedState state = loadFromTextFile(file);

if (state != null) {

listRecIntegral = state.getListRecIntegral();

setDataToTable(state.getArrListTableData());

}

}

**Добавили функционал для работы в двоичном виде**

**Кнопка “Сохранить”**

private void bSaveObjectBinFormatMouseClicked(java.awt.event.MouseEvent evt) {

// TODO add your handling code here:

File file = getPathSerFileToSaved();

if (file == null) return;

ArrayList<RecIntegral> arrTableData = getDataArrListFromTable();

SavedState state = new SavedState(listRecIntegral, arrTableData);

saveToBinaryFile(file, state);

}

**Кнопка “Загрузить”**

private void bLoadObjectBinFormatMouseClicked(java.awt.event.MouseEvent evt) {

// TODO add your handling code here:

File file = getPathSerFileToLoad();

if (file == null) return;

SavedState state = loadFromBinaryFile(file);

if (state != null) {

listRecIntegral = state.getListRecIntegral();

setDataToTable(state.getArrListTableData());

}

}

**Листинг**

**Frame.java**

package lab1;

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.util.ArrayList;

import javax.swing.table.DefaultTableModel;

import java.util.LinkedList;

import javax.swing.JFileChooser;

import javax.swing.JOptionPane;

import javax.swing.filechooser.FileNameExtensionFilter;

public class Frame extends javax.swing.JFrame {

/\*\*

\* Creates new form Frame

\*/

public Frame() {

initComponents();

}

/\*\*

\* 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() {

label1 = new java.awt.Label();

label2 = new java.awt.Label();

label3 = new java.awt.Label();

jTextFieldSH = new javax.swing.JTextField();

jTextFieldNG = new javax.swing.JTextField();

jTextFieldVG = new javax.swing.JTextField();

jScrollPane1 = new javax.swing.JScrollPane();

jTable1 = new javax.swing.JTable();

jButtonDel = new javax.swing.JButton();

jButtonRes = new javax.swing.JButton();

jButtonAdd = new javax.swing.JButton();

jButtonClearTable = new javax.swing.JButton();

jButtonFillTable = new javax.swing.JButton();

bSaveObjectTextFormat = new javax.swing.JButton();

bLoadObjectTextFormat = new javax.swing.JButton();

bSaveObjectBinFormat = new javax.swing.JButton();

bLoadObjectBinFormat = new javax.swing.JButton();

label4 = new java.awt.Label();

label5 = new java.awt.Label();

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

label1.setFont(new java.awt.Font("Dialog", 0, 14)); // NOI18N

label1.setText("Нижняя граница");

label2.setFont(new java.awt.Font("Dialog", 0, 14)); // NOI18N

label2.setText("Ширина шага");

label3.setFont(new java.awt.Font("Dialog", 0, 14)); // NOI18N

label3.setText("Верхняя граница");

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

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

jTextFieldSHActionPerformed(evt);

}

});

jTextFieldNG.setCursor(new java.awt.Cursor(java.awt.Cursor.TEXT\_CURSOR));

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

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

jTextFieldNGActionPerformed(evt);

}

});

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

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

jTextFieldVGActionPerformed(evt);

}

});

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

new Object [][] {},

new String [] {

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

}

));

jTable1.setRowSelectionAllowed(false);

jTable1.addAncestorListener(new javax.swing.event.AncestorListener() {

public void ancestorAdded(javax.swing.event.AncestorEvent evt) {

jTable1AncestorAdded(evt);

}

public void ancestorMoved(javax.swing.event.AncestorEvent evt) {

}

public void ancestorRemoved(javax.swing.event.AncestorEvent evt) {

}

});

jScrollPane1.setViewportView(jTable1);

if (jTable1.getColumnModel().getColumnCount() > 0) {

jTable1.getColumnModel().getColumn(0).setHeaderValue("Нижняя граница");

jTable1.getColumnModel().getColumn(1).setHeaderValue("Верхняя граница");

jTable1.getColumnModel().getColumn(2).setHeaderValue("Шаг");

jTable1.getColumnModel().getColumn(3).setHeaderValue("Результат");

}

jButtonDel.setText("Удалить");

jButtonDel.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

jButtonDelMouseClicked(evt);

}

});

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

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

jButtonDelActionPerformed(evt);

}

});

jButtonRes.setText("Вычислить");

jButtonRes.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

jButtonResMouseClicked(evt);

}

});

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

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

jButtonResActionPerformed(evt);

}

});

jButtonAdd.setText("Добавить");

jButtonAdd.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

jButtonAddMouseClicked(evt);

}

});

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

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

jButtonAddActionPerformed(evt);

}

});

jButtonClearTable.setText("Очистить");

jButtonClearTable.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

jButtonClearTableMouseClicked(evt);

}

});

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

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

jButtonClearTableActionPerformed(evt);

}

});

jButtonFillTable.setText("Заполнить");

jButtonFillTable.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

jButtonFillTableMouseClicked(evt);

}

});

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

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

jButtonFillTableActionPerformed(evt);

}

});

bSaveObjectTextFormat.setText("Сохранить");

bSaveObjectTextFormat.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

bSaveObjectTextFormatMouseClicked(evt);

}

});

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

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

bSaveObjectTextFormatActionPerformed(evt);

}

});

bLoadObjectTextFormat.setText("Загрузить");

bLoadObjectTextFormat.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

bLoadObjectTextFormatMouseClicked(evt);

}

});

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

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

bLoadObjectTextFormatActionPerformed(evt);

}

});

bSaveObjectBinFormat.setText("Сохранить");

bSaveObjectBinFormat.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

bSaveObjectBinFormatMouseClicked(evt);

}

});

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

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

bSaveObjectBinFormatActionPerformed(evt);

}

});

bLoadObjectBinFormat.setText("Загрузить");

bLoadObjectBinFormat.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

bLoadObjectBinFormatMouseClicked(evt);

}

});

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

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

bLoadObjectBinFormatActionPerformed(evt);

}

});

label4.setFont(new java.awt.Font("Dialog", 0, 14)); // NOI18N

label4.setText("В текстовом виде");

label5.setFont(new java.awt.Font("Dialog", 0, 14)); // NOI18N

label5.setText("В двоичном виде");

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()

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

.addGroup(layout.createSequentialGroup()

.addComponent(bSaveObjectTextFormat, javax.swing.GroupLayout.PREFERRED\_SIZE, 144, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(bLoadObjectTextFormat, javax.swing.GroupLayout.PREFERRED\_SIZE, 144, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(24, 24, 24)

.addComponent(bSaveObjectBinFormat, javax.swing.GroupLayout.PREFERRED\_SIZE, 144, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(bLoadObjectBinFormat, javax.swing.GroupLayout.PREFERRED\_SIZE, 144, javax.swing.GroupLayout.PREFERRED\_SIZE))

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

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 609, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGroup(layout.createSequentialGroup()

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

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)

.addComponent(label2, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(label1, javax.swing.GroupLayout.DEFAULT\_SIZE, 117, Short.MAX\_VALUE))

.addComponent(label3, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

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

.addGroup(layout.createSequentialGroup()

.addComponent(jTextFieldVG, javax.swing.GroupLayout.PREFERRED\_SIZE, 192, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 134, Short.MAX\_VALUE)

.addComponent(jButtonRes, javax.swing.GroupLayout.PREFERRED\_SIZE, 144, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createSequentialGroup()

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

.addComponent(jTextFieldNG, javax.swing.GroupLayout.DEFAULT\_SIZE, 192, Short.MAX\_VALUE)

.addComponent(jTextFieldSH))

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

.addGroup(layout.createSequentialGroup()

.addGap(134, 134, 134)

.addComponent(jButtonAdd, javax.swing.GroupLayout.PREFERRED\_SIZE, 144, javax.swing.GroupLayout.PREFERRED\_SIZE))

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

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

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

.addComponent(jButtonClearTable, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.PREFERRED\_SIZE, 144, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jButtonDel, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.PREFERRED\_SIZE, 144, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jButtonFillTable, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.PREFERRED\_SIZE, 144, javax.swing.GroupLayout.PREFERRED\_SIZE)))))))

.addGroup(javax.swing.GroupLayout.Alignment.LEADING, layout.createSequentialGroup()

.addComponent(label4, javax.swing.GroupLayout.PREFERRED\_SIZE, 307, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(label5, javax.swing.GroupLayout.PREFERRED\_SIZE, 284, javax.swing.GroupLayout.PREFERRED\_SIZE))))

.addContainerGap(15, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

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

.addGroup(layout.createSequentialGroup()

.addGap(30, 30, 30)

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

.addGroup(layout.createSequentialGroup()

.addComponent(jButtonAdd, javax.swing.GroupLayout.PREFERRED\_SIZE, 29, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(jButtonDel, javax.swing.GroupLayout.PREFERRED\_SIZE, 29, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(jButtonRes, javax.swing.GroupLayout.PREFERRED\_SIZE, 29, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(16, 16, 16))

.addGroup(layout.createSequentialGroup()

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

.addComponent(jTextFieldNG, javax.swing.GroupLayout.PREFERRED\_SIZE, 28, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(label1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(18, 18, 18)

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

.addComponent(jTextFieldVG, javax.swing.GroupLayout.PREFERRED\_SIZE, 28, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(label3, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(23, 23, 23)))

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

.addGroup(layout.createSequentialGroup()

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

.addComponent(jTextFieldSH, javax.swing.GroupLayout.PREFERRED\_SIZE, 28, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jButtonClearTable, javax.swing.GroupLayout.PREFERRED\_SIZE, 29, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(jButtonFillTable, javax.swing.GroupLayout.PREFERRED\_SIZE, 29, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addComponent(label2, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(22, 22, 22)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 130, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

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

.addComponent(label4, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(label5, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

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

.addComponent(bSaveObjectTextFormat, javax.swing.GroupLayout.PREFERRED\_SIZE, 29, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(bLoadObjectTextFormat, javax.swing.GroupLayout.PREFERRED\_SIZE, 29, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(bSaveObjectBinFormat, javax.swing.GroupLayout.PREFERRED\_SIZE, 29, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(bLoadObjectBinFormat, javax.swing.GroupLayout.PREFERRED\_SIZE, 29, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addContainerGap(14, Short.MAX\_VALUE))

);

label5.getAccessibleContext().setAccessibleName("В двоичном виде");

pack();

}// </editor-fold>

LinkedList<RecIntegral> listRecIntegral = new LinkedList<>();

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

// TODO add your handling code here:

}

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

// TODO add your handling code here:

}

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

// TODO add your handling code here:

}

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

// TODO add your handling code here:

}

private void jTable1AncestorAdded(javax.swing.event.AncestorEvent evt) {

// TODO add your handling code here:

}

private void jButtonDelMouseClicked(java.awt.event.MouseEvent evt) {

// TODO add your handling code here:

if(jTable1.getRowCount() != 0)

{

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

int selectRow = jTable1.getSelectedRow();

if(selectRow == -1) return;

model.removeRow(selectRow);

}

}

public double calculateIntegral(double lowerBorder, double upperBorder, double weight) {

boolean isReversed = false;

if(lowerBorder > upperBorder)

{

isReversed = true;

double tempBorder = lowerBorder;

lowerBorder = upperBorder;

upperBorder = tempBorder;

}

double tempLowerBorder = lowerBorder;

int count = (int) Math.floor((upperBorder - lowerBorder) / weight);

double sum = 0;

for (int j = 0; j < count; j++) {

sum += ((weight / 2) \* (Math.sqrt(tempLowerBorder) + Math.sqrt(tempLowerBorder + weight)));

tempLowerBorder += weight;

}

if((upperBorder - lowerBorder) / weight > count)

{

tempLowerBorder -= weight;

double lastStepWeigth = upperBorder - (tempLowerBorder);

sum += ((lastStepWeigth / 2) \* (Math.sqrt(tempLowerBorder) + Math.sqrt(upperBorder)));

}

if (isReversed)

sum = -sum;

return sum;

}

private void jButtonResMouseClicked(java.awt.event.MouseEvent evt) {

// TODO add your handling code here:

if(jTable1.getRowCount() == 0) return;

int selectRow = jTable1.getSelectedRow();

if(selectRow == -1) return;

double lowerBorder = (double) jTable1.getValueAt(selectRow, 0);

double upperBorder = (double) jTable1.getValueAt(selectRow, 1);

double weight = (double) jTable1.getValueAt(selectRow, 2);

jTable1.setValueAt(calculateIntegral(lowerBorder, upperBorder, weight), selectRow, 3);

}

private void validData(double data) throws DataException {

if (data == 0 || (data >= 0.01 && data <= 10)) {

throw new DataException("Значение должно быть в пределах от 0.000001 до 1000000.");

}

}

private void jButtonAddMouseClicked(java.awt.event.MouseEvent evt) {

// TODO add your handling code here:

try{

double widthLim = Double.parseDouble(jTextFieldSH.getText());

validData(widthLim);

double lowLim = Double.parseDouble(jTextFieldNG.getText());

validData(lowLim);

double upLim = Double.parseDouble(jTextFieldVG.getText());

validData(upLim);

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

model.addRow(new Object[]{lowLim, upLim, widthLim});

}

catch(DataException | NumberFormatException ex){

// Вывод сообщения об ошибке

javax.swing.JOptionPane.showMessageDialog(this,

"Ошибка ввода! Значение должно быть числом в пределах от 0.000001 до 1000000.",

"Ошибка",

javax.swing.JOptionPane.ERROR\_MESSAGE);

}

}

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

// TODO add your handling code here:

}

private void jButtonClearTableMouseClicked(java.awt.event.MouseEvent evt) {

// TODO add your handling code here:

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

int count = model.getRowCount();

for (int i = 0; i < count; i++) {

double lowLim = (double) model.getValueAt(i, 0);

double upLim = (double) model.getValueAt(i, 1);

double widthLim = (double) model.getValueAt(i, 2);

double resIntegral;

Object value = model.getValueAt(i, 3);

if (value instanceof Number) {

resIntegral = ((Number) value).doubleValue();

} else {

resIntegral = Double.MAX\_VALUE;

}

RecIntegral dataIntegral = new RecIntegral(lowLim, upLim, widthLim, resIntegral);

listRecIntegral.add(dataIntegral);

}

model.setRowCount(0);

}

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

// TODO add your handling code here:

}

private void jButtonFillTableMouseClicked(java.awt.event.MouseEvent evt) {

// TODO add your handling code here:

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

for(RecIntegral dataRow : listRecIntegral) {

double lowLim = dataRow.getLowLim();

double upLim = dataRow.getUpLim();

double widthLim = dataRow.getWidthLim();

if (dataRow.getResIntegral() == Double.MAX\_VALUE) {

model.addRow(new Object[]{lowLim, upLim, widthLim});

}

else {

double resIntegral = dataRow.getResIntegral();

model.addRow(new Object[]{lowLim, upLim, widthLim, resIntegral});

}

}

listRecIntegral.clear();

}

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

// TODO add your handling code here:

}

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

// TODO add your handling code here:

}

public static void saveToTextFile(File file, SavedState state) {

try (FileWriter writer = new FileWriter(file)) {

for (RecIntegral recIntegral : state.getListRecIntegral()) {

writer.write(recIntegral.getLowLim() + "," +

recIntegral.getUpLim() + "," +

recIntegral.getWidthLim() + "," +

recIntegral.getResIntegral() + "\n");

}

writer.write("---\n");

for (RecIntegral recIntegral : state.getArrListTableData()) {

writer.write(recIntegral.getLowLim() + "," +

recIntegral.getUpLim() + "," +

recIntegral.getWidthLim() + "," +

recIntegral.getResIntegral() + "\n");

}

} catch (IOException e) {

JOptionPane.showMessageDialog(null, "Ошибка при сохранении файла: " + e.getMessage(), "Ошибка", JOptionPane.ERROR\_MESSAGE);

}

}

public static SavedState loadFromTextFile(File file) {

LinkedList<RecIntegral> listRecIntegral = new LinkedList<>();

ArrayList<RecIntegral> arrTableData = new ArrayList<>();

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

String line;

boolean isListRecIntegral = true;

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

if (line.equals("---")) {

isListRecIntegral = false;

continue;

}

String[] parts = line.split(",");

if (parts.length == 4) {

double lowLim = Double.parseDouble(parts[0]);

double upLim = Double.parseDouble(parts[1]);

double widthLim = Double.parseDouble(parts[2]);

double resIntegral = Double.parseDouble(parts[3]);

RecIntegral recIntegral = new RecIntegral(lowLim, upLim, widthLim, resIntegral);

if (isListRecIntegral) {

listRecIntegral.add(recIntegral);

} else {

arrTableData.add(recIntegral);

}

}

}

return new SavedState(listRecIntegral, arrTableData);

} catch (IOException e) {

JOptionPane.showMessageDialog(null, "Ошибка при загрузке файла: " + e.getMessage(), "Ошибка", JOptionPane.ERROR\_MESSAGE);

return null;

}

}

private void bSaveObjectTextFormatMouseClicked(java.awt.event.MouseEvent evt) {

// TODO add your handling code here:

File file = getPathTXTFileToSaved();

if (file == null) return;

ArrayList<RecIntegral> arrTableData = getDataArrListFromTable();

SavedState state = new SavedState(listRecIntegral, arrTableData);

saveToTextFile(file, state);

}

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

// TODO add your handling code here:

}

private void bLoadObjectTextFormatMouseClicked(java.awt.event.MouseEvent evt) {

// TODO add your handling code here:

File file = getPathTXTFileToLoad();

if (file == null) return;

SavedState state = loadFromTextFile(file);

if (state != null) {

listRecIntegral = state.getListRecIntegral();

setDataToTable(state.getArrListTableData());

}

}

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

// TODO add your handling code here:

}

private ArrayList<RecIntegral> getDataArrListFromTable() {

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

int count = model.getRowCount();

ArrayList<RecIntegral> data = new ArrayList<>();

for (int i = 0; i < count; i++) {

double lowLim = (double) model.getValueAt(i, 0);

double upLim = (double) model.getValueAt(i, 1);

double widthLim = (double) model.getValueAt(i, 2);

double resIntegral;

Object value = model.getValueAt(i, 3);

if (value instanceof Number) {

resIntegral = ((Number) value).doubleValue();

} else {

resIntegral = Double.MAX\_VALUE;

}

data.add(new RecIntegral(lowLim, upLim, widthLim, resIntegral));

}

return data;

}

private void setDataToTable(ArrayList<RecIntegral> data) {

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

model.setRowCount(0);

for (RecIntegral row : data) {

if (row.getResIntegral() == Double.MAX\_VALUE) {

model.addRow(new Object[]{row.getLowLim(), row.getUpLim(), row.getWidthLim()});

} else {

model.addRow(new Object[]{row.getLowLim(), row.getUpLim(), row.getWidthLim(), row.getResIntegral()});

}

}

}

public static void saveToBinaryFile(File file, SavedState data) {

try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream(file))) {

oos.writeObject(data);

} catch (IOException e) {

JOptionPane.showMessageDialog(null, "Ошибка при сохранении файла: " + e.getMessage(), "Ошибка", JOptionPane.ERROR\_MESSAGE);

}

}

public static SavedState loadFromBinaryFile(File file) {

try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(file))) {

return (SavedState) ois.readObject();

} catch (IOException | ClassNotFoundException e) {

JOptionPane.showMessageDialog(null, "Ошибка при загрузке файла: " + e.getMessage(), "Ошибка", JOptionPane.ERROR\_MESSAGE);

return null;

}

}

private static File getFilePath(int mode, String extension, String description) {

JFileChooser fileChooser = new JFileChooser();

FileNameExtensionFilter filter = new FileNameExtensionFilter(description, extension.substring(1));

fileChooser.setFileFilter(filter);

while (true) {

int option = (mode == JFileChooser.SAVE\_DIALOG) ? fileChooser.showSaveDialog(null) : fileChooser.showOpenDialog(null);

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

File file = fileChooser.getSelectedFile();

if (file == null) return null;

if (!file.getName().contains(".")) {

file = new File(file.getAbsolutePath() + extension);

}

else if (!file.getName().toLowerCase().endsWith(extension)) {

JOptionPane.showMessageDialog(

null,

"Файл должен иметь расширение " + extension + ". Пожалуйста, выберите другой файл.",

"Ошибка",

JOptionPane.ERROR\_MESSAGE

);

continue;

}

if (mode == JFileChooser.SAVE\_DIALOG && file.exists()) {

int overwriteOption = JOptionPane.showConfirmDialog(

null,

"Файл уже существует. Перезаписать?",

"Предупреждение",

JOptionPane.YES\_NO\_OPTION

);

if (overwriteOption != JOptionPane.YES\_OPTION) {

return null;

}

}

if (mode == JFileChooser.OPEN\_DIALOG && !file.exists()) {

JOptionPane.showMessageDialog(

null,

"Файл не существует.",

"Ошибка",

JOptionPane.ERROR\_MESSAGE

);

return null;

}

return file;

} else {

return null;

}

}

}

public static File getPathSerFileToSaved() {

return getFilePath(JFileChooser.SAVE\_DIALOG, ".ser", "Serialized Files (\*.ser)");

}

public static File getPathSerFileToLoad() {

return getFilePath(JFileChooser.OPEN\_DIALOG, ".ser", "Serialized Files (\*.ser)");

}

public static File getPathTXTFileToSaved() {

return getFilePath(JFileChooser.SAVE\_DIALOG, ".txt", "Text Files (\*.txt)");

}

public static File getPathTXTFileToLoad() {

return getFilePath(JFileChooser.OPEN\_DIALOG, ".txt", "Text Files (\*.txt)");

}

private void bSaveObjectBinFormatMouseClicked(java.awt.event.MouseEvent evt) {

// TODO add your handling code here:

File file = getPathSerFileToSaved();

if (file == null) return;

ArrayList<RecIntegral> arrTableData = getDataArrListFromTable();

SavedState state = new SavedState(listRecIntegral, arrTableData);

saveToBinaryFile(file, state);

}

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

// TODO add your handling code here:

}

private void bLoadObjectBinFormatMouseClicked(java.awt.event.MouseEvent evt) {

// TODO add your handling code here:

File file = getPathSerFileToLoad();

if (file == null) return;

SavedState state = loadFromBinaryFile(file);

if (state != null) {

listRecIntegral = state.getListRecIntegral();

setDataToTable(state.getArrListTableData());

}

}

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

// TODO add your handling code here:

}

/\*\*

\* @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(Frame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

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

} catch (IllegalAccessException ex) {

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

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Frame.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 Frame().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton bLoadObjectBinFormat;

private javax.swing.JButton bLoadObjectTextFormat;

private javax.swing.JButton bSaveObjectBinFormat;

private javax.swing.JButton bSaveObjectTextFormat;

private javax.swing.JButton jButtonAdd;

private javax.swing.JButton jButtonClearTable;

private javax.swing.JButton jButtonDel;

private javax.swing.JButton jButtonFillTable;

private javax.swing.JButton jButtonRes;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JTable jTable1;

private javax.swing.JTextField jTextFieldNG;

private javax.swing.JTextField jTextFieldSH;

private javax.swing.JTextField jTextFieldVG;

private java.awt.Label label1;

private java.awt.Label label2;

private java.awt.Label label3;

private java.awt.Label label4;

private java.awt.Label label5;

// End of variables declaration

}

**Lab1.java**

package lab1;

public class Lab1 {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

// TODO code application logic here

Frame frame = new Frame();

frame.setTitle("Вычисление интегралов");

frame.show();

}

}

**RecIntegral.java**

package lab1;

import java.io.Serializable;

/\*\*

\* Класс для хранения данных о вычислении интеграла.

\* Реализует интерфейс Serializable для поддержки сериализации.

\*/

public class RecIntegral implements Serializable {

private static final long serialVersionUID = 1L;

private double widthLim;

private double lowLim;

private double upLim;

private double resIntegral;

/\*\*

\* Конструктор для создания объекта RecIntegral.

\*

\* @param lowLim Нижняя граница.

\* @param upLim Верхняя граница.

\* @param widthLim Ширина шага.

\* @param resIntegral Результат вычисления интеграла.

\*/

public RecIntegral(double lowLim, double upLim, double widthLim, double resIntegral) {

this.widthLim = widthLim;

this.lowLim = lowLim;

this.upLim = upLim;

this.resIntegral = resIntegral;

}

/\*\*

\* Возвращает нижнюю границу.

\*

\* @return Нижняя граница.

\*/

public double getLowLim() {

return lowLim;

}

/\*\*

\* Устанавливает нижнюю границу.

\*

\* @param lowLim Нижняя граница.

\*/

public void setLowLim(double lowLim) {

this.lowLim = lowLim;

}

/\*\*

\* Возвращает верхнюю границу.

\*

\* @return Верхняя граница.

\*/

public double getUpLim() {

return upLim;

}

/\*\*

\* Устанавливает верхнюю границу.

\*

\* @param upLim Верхняя граница.

\*/

public void setUpLim(double upLim) {

this.upLim = upLim;

}

/\*\*

\* Возвращает ширину шага.

\*

\* @return Ширина шага.

\*/

public double getWidthLim() {

return widthLim;

}

/\*\*

\* Устанавливает ширину шага.

\*

\* @param widthLim Ширина шага.

\*/

public void setWidthLim(double widthLim) {

this.widthLim = widthLim;

}

/\*\*

\* Возвращает результат вычисления интеграла.

\*

\* @return Результат вычисления интеграла.

\*/

public double getResIntegral() {

return resIntegral;

}

/\*\*

\* Устанавливает результат вычисления интеграла.

\*

\* @param resIntegral Результат вычисления интеграла.

\*/

public void setResIntegral(double resIntegral) {

this.resIntegral = resIntegral;

}

@Override

public String toString() {

return "RecIntegral{" +

"lowLim=" + lowLim +

", upLim=" + upLim +

", widthLim=" + widthLim +

", resIntegral=" + resIntegral +

'}';

}

}

**DataException.java**

package lab1;

/\*\*

\* Исключение, которое выбрасывается при некорректных данных.

\* Наследуется от класса Exception.

\*/

public class DataException extends Exception {

/\*\*

\* Создает новое исключение с указанным сообщением.

\*

\* @param message Сообщение об ошибке.

\*/

public DataException (String message){

super(message);

}

}

**SavedState.java**

package lab1;

import java.io.Serializable;

import java.util.ArrayList;

import java.util.LinkedList;

/\*\*

\* Класс для хранения состояния приложения, включающего два списка объектов RecIntegral.

\* Реализует интерфейс Serializable для поддержки сериализации.

\*/

public class SavedState implements Serializable {

private static final long serialVersionUID = 2L;

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

private ArrayList<RecIntegral> arrTableData = new ArrayList<>();

/\*\*

\* Конструктор для создания объекта SavedState.

\*

\* @param listRecIntegral Список RecIntegral для хранения.

\* @param arrTableData Список данных таблицы для хранения.

\*/

public SavedState(LinkedList<RecIntegral> listRecIntegral, ArrayList<RecIntegral> arrTableData) {

this.listRecIntegral = new LinkedList<>(listRecIntegral);

this.arrTableData = new ArrayList<>(arrTableData);

}

/\*\*

\* Возвращает копию списка RecIntegral вне таблицы.

\*

\* @return Копия списка RecIntegral вне таблицы.

\*/

public LinkedList<RecIntegral> getListRecIntegral() {

return new LinkedList<>(listRecIntegral);

}

/\*\*

\* Возвращает копию списка данных таблицы.

\*

\* @return Копия списка данных таблицы.

\*/

public ArrayList<RecIntegral> getArrListTableData() {

return new ArrayList<>(arrTableData);

}

@Override

public String toString() {

return "SavedState{" +

"listRecIntegral=" + listRecIntegral +

", arrTableData=" + arrTableData +

'}';

}

}

**Вывод:** В ходе выполнения лабораторной работы изучили работу с файлами и механизмы сериализации данных.