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

Горбатов К.

Коробкин В.

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**

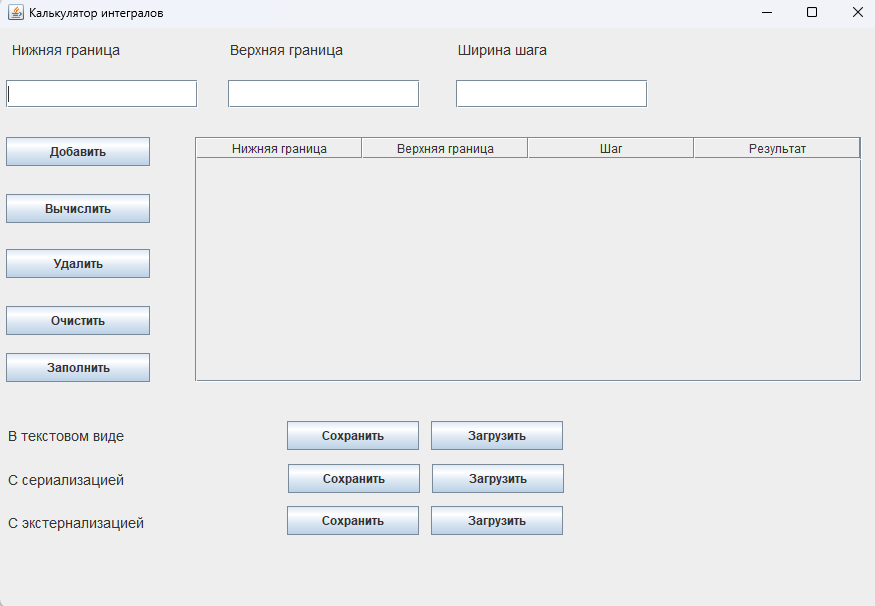
Модифицировать приложение из предыдущей лабораторной работы, реализовав сохранение в файл и загрузку данных из файла. Предусмотреть сохранение данных, как в текстовом виде, так и в двоичном (с использованием механизма сериализации). Для этого нужно добавить 4 кнопки для сохранения и загрузки в текстовом 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 8:



**Ход работы**

**Добавили в интерфейс 6 новые кнопки**

****

**Добавили класс SavedStateSerializable**

package MyLab4;

import java.io.Serializable;

import java.util.ArrayList;

public class SavedStateSerializable implements Serializable {

private static final long serialVersionUID = 2L;

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

public SavedStateSerializable(ArrayList<RecIntegral> listRecIntegral) {

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

}

public ArrayList<RecIntegral> getListRecIntegral() {

return new ArrayList<>(listRecIntegral);

}

@Override

public String toString() {

return "SavedStateSerializable{" +

"listRecIntegral=" + listRecIntegral +

'}';

}

}

**Добавили класс ExternalSavedState**

package MyLab4;

import java.io.Externalizable;

import java.io.IOException;

import java.io.ObjectInput;

import java.io.ObjectOutput;

import java.util.ArrayList;

public class ExternalSavedState implements Externalizable {

private static final long serialVersionUID = 3L;

private ArrayList<RecIntegral> listRecIntegral;

public ExternalSavedState() {

this.listRecIntegral = new ArrayList<>();

}

public ExternalSavedState(ArrayList<RecIntegral> listRecIntegral) {

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

}

@Override

public void writeExternal(ObjectOutput out) throws IOException {

out.writeObject(listRecIntegral);

}

@Override

@SuppressWarnings("unchecked")

public void readExternal(ObjectInput in) throws IOException, ClassNotFoundException {

listRecIntegral = (ArrayList<RecIntegral>) in.readObject();

}

public ArrayList<RecIntegral> getArrListTableData() {

return new ArrayList<>(listRecIntegral);

}

@Override

public String toString() {

return "ExternalSavedState {" +

"listRecIntegral=" + listRecIntegral +

'}';

}

}

**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 = fileManager.getPathTXTFileToSaved();

if (file == null) return;

SavedStateSerializable state = new SavedStateSerializable(listRecIntegral);

fileManager.saveToTextFile(file, state);

}

**Кнопка “Загрузить”**

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

// TODO add your handling code here:

File file = fileManager.getPathTXTFileToLoad();

if (file == null) return;

SavedStateSerializable state = fileManager.loadFromTextFile(file);

if (state != null) {

setDataToTable(state.getListRecIntegral());

setDataToArr(state.getListRecIntegral());

}

}

**Добавили функционал для работы в двоичном виде с сериализацией**

**Кнопка “Сохранить”**

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

// TODO add your handling code here:

File file = fileManager.getPathSerFileToSaved();

if (file == null) return;

SavedStateSerializable state = new SavedStateSerializable(listRecIntegral);

fileManager.saveToBinaryFile(file, state);

}

**Кнопка “Загрузить”**

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

// TODO add your handling code here:

File file = fileManager.getPathSerFileToLoad();

if (file == null) return;

SavedStateSerializable state = fileManager.loadFromBinaryFile(file);

if (state != null) {

setDataToTable(state.getListRecIntegral());

setDataToArr(state.getListRecIntegral());

}

}

**Добавили функционал для работы в двоичном виде с экстернализацией**

**Кнопка “Сохранить”**

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

// TODO add your handling code here:

File file = fileManager.getPathExternFileToSaved();

if (file == null) return;

ExternalSavedState state = new ExternalSavedState(listRecIntegral);

fileManager.saveToBinaryExternFile(file, state);

}

**Кнопка “Загрузить”**

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

// TODO add your handling code here:

File file = fileManager.getPathExternFileToLoad();

if (file == null) return;

ExternalSavedState state = fileManager.loadFromBinaryExternFile(file);

if (state != null) {

setDataToTable(state.getArrListTableData());

setDataToArr(state.getArrListTableData());

}

}

**Листинг**

**CalculateIntegral.java**

package MyLab4;

import static java.lang.Math.pow;

import static java.lang.Math.sin;

public class CalculateIntegral {

private double Bottom;

private double Top;

private double step;

private double result;

public CalculateIntegral(double Bottom, double Top, double step) {

this.Bottom = Bottom;

this.Top = Top;

this.step = step;

}

public CalculateIntegral(RecIntegral data) {

this.Bottom = data.Bottom;

this.Top = data.Top;

this.step = data.Step;

}

public double calculate() {

boolean isReversed = Bottom > Top;

if (isReversed) {

double tempBorder = Bottom;

Bottom = Top;

Top = tempBorder;

}

double currentBottom = Bottom;

long count = (long)((Top - Bottom) / step);

double sum = 0;

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

sum += ((step / 2) \* (sin(pow(currentBottom,2)) + sin(pow((currentBottom + step), 2))));

currentBottom += step;

}

if((Top - Bottom) / step > count) {

currentBottom -= step;

double lastStepWeigth = Top - (currentBottom);

sum += ((lastStepWeigth / 2) \* (sin(pow(currentBottom, 2)) + sin(pow(Top, 2))));

}

return isReversed ? -sum : sum;

}

}

**DoubleRenderer.java**

package MyLab4;

import java.text.DecimalFormat;

import javax.swing.table.DefaultTableCellRenderer;

class DoubleRenderer extends DefaultTableCellRenderer {

private DecimalFormat formatter = new DecimalFormat("#.##########");

@Override

protected void setValue(Object value) {

if (value instanceof Double) {

setText(formatter.format(value));

} else {

super.setValue(value);

}

}

}

**ExceptionforRecIntegral.java**

package MyLab4;

public class ExceptionforRecIntegral extends Exception {

public ExceptionforRecIntegral(String description) {

super(description);

}

}

**ExternalSavedState.java**

package MyLab4;

import java.io.Externalizable;

import java.io.IOException;

import java.io.ObjectInput;

import java.io.ObjectOutput;

import java.util.ArrayList;

public class ExternalSavedState implements Externalizable {

private static final long serialVersionUID = 3L;

private ArrayList<RecIntegral> listRecIntegral;

public ExternalSavedState() {

this.listRecIntegral = new ArrayList<>();

}

public ExternalSavedState(ArrayList<RecIntegral> listRecIntegral) {

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

}

@Override

public void writeExternal(ObjectOutput out) throws IOException {

out.writeObject(listRecIntegral);

}

@Override

@SuppressWarnings("unchecked")

public void readExternal(ObjectInput in) throws IOException, ClassNotFoundException {

listRecIntegral = (ArrayList<RecIntegral>) in.readObject();

}

public ArrayList<RecIntegral> getArrListTableData() {

return new ArrayList<>(listRecIntegral);

}

@Override

public String toString() {

return "ExternalSavedState {" +

"listRecIntegral=" + listRecIntegral +

'}';

}

}

**FileOperations.java**

package MyLab4;

import javax.swing.JFrame;

import java.io.\*;

import java.util.ArrayList;

import javax.swing.\*;

import javax.swing.filechooser.FileNameExtensionFilter;

public class FileOperations {

private final JFrame parentFrame;

public FileOperations(JFrame parentFrame) {

this.parentFrame = parentFrame;

}

//Операции в текстовом виде

public void saveToTextFile(File file, SavedStateSerializable state) {

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

writeRecIntegralList(writer, state.getListRecIntegral());

} catch (IOException e) {

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

}

}

public SavedStateSerializable loadFromTextFile(File file) {

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

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

parseTextFile(reader, listRecIntegral);

return new SavedStateSerializable(listRecIntegral);

} catch (IOException e) {

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

return null;

}

}

//Операции с сериализацией

public void saveToBinaryFile(File file, SavedStateSerializable data) {

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

oos.writeObject(data);

} catch (IOException e) {

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

}

}

public SavedStateSerializable loadFromBinaryFile(File file) {

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

return (SavedStateSerializable) ois.readObject();

} catch (IOException | ClassNotFoundException e) {

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

return null;

}

}

//Операции с экстернализацией

public void saveToBinaryExternFile(File file, ExternalSavedState data) {

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

oos.writeObject(data);

} catch (IOException e) {

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

}

}

public ExternalSavedState loadFromBinaryExternFile(File file) {

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

return (ExternalSavedState) ois.readObject();

} catch (IOException | ClassNotFoundException e) {

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

return null;

}

}

private 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(parentFrame, "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(parentFrame, "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(parentFrame, "Файл не существует.", "Ошибка", JOptionPane.ERROR\_MESSAGE);

return null;

}

return file;

} else {

return null;

}

}

}

public File getPathSerFileToSaved() {

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

}

public File getPathSerFileToLoad() {

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

}

public File getPathTXTFileToSaved() {

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

}

public File getPathTXTFileToLoad() {

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

}

public File getPathExternFileToSaved() {

return getFilePath(JFileChooser.SAVE\_DIALOG, ".dat", "Externalizable Files (\*.dat)");

}

public File getPathExternFileToLoad() {

return getFilePath(JFileChooser.OPEN\_DIALOG, ".dat", "Externalizable Files (\*.dat)");

}

private void writeRecIntegralList(FileWriter writer, Iterable<RecIntegral> list) throws IOException {

for (RecIntegral recIntegral : list) {

writer.write(String.format("%f;%f;%f;%f%n",

recIntegral.getBottom(),

recIntegral.getTop(),

recIntegral.getStep(),

recIntegral.getRes()));

}

}

private void parseTextFile(BufferedReader reader, ArrayList<RecIntegral> listRecIntegral) throws IOException {

String line;

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

line = line.trim();

if (line.isEmpty()) {

continue;

}

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

if (parts.length == 4) {

RecIntegral recIntegral = createRecIntegralFromParts(parts);

listRecIntegral.add(recIntegral);

}

}

}

private RecIntegral createRecIntegralFromParts(String[] parts) {

String lowLimStr = parts[0].replace(',', '.');

String upLimStr = parts[1].replace(',', '.');

String widthLimStr = parts[2].replace(',', '.');

String resIntegralStr = parts[3].replace(',', '.');

double lowLim = Double.parseDouble(lowLimStr);

double upLim = Double.parseDouble(upLimStr);

double widthLim = Double.parseDouble(widthLimStr);

double resIntegral = Double.parseDouble(resIntegralStr);

return new RecIntegral(lowLim, upLim, widthLim, resIntegral);

}

}

**Frame.java**

package MyLab4;

import java.io.File;

import java.util.ArrayList;

import javax.swing.table.DefaultTableModel;

public class Frame extends javax.swing.JFrame {

private final FileOperations fileManager = new FileOperations(this);

/\*\*

\* 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();

bSaveObjectSerBinFormat = new javax.swing.JButton();

bLoadObjectSerBinFormat = new javax.swing.JButton();

textTXT = new java.awt.Label();

textSer = new java.awt.Label();

textExtern = new java.awt.Label();

bSaveObjecExternBinFormat = new javax.swing.JButton();

bLoadObjecExterntBinFormat = new javax.swing.JButton();

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);

}

});

bSaveObjectSerBinFormat.setText("Сохранить");

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

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

bSaveObjectSerBinFormatMouseClicked(evt);

}

});

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

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

bSaveObjectSerBinFormatActionPerformed(evt);

}

});

bLoadObjectSerBinFormat.setText("Загрузить");

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

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

bLoadObjectSerBinFormatMouseClicked(evt);

}

});

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

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

bLoadObjectSerBinFormatActionPerformed(evt);

}

});

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

textTXT.setText("В текстовом виде");

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

textSer.setText("C сериализацией");

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

textExtern.setText("C экстернализацией");

bSaveObjecExternBinFormat.setText("Сохранить");

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

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

bSaveObjecExternBinFormatMouseClicked(evt);

}

});

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

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

bSaveObjecExternBinFormatActionPerformed(evt);

}

});

bLoadObjecExterntBinFormat.setText("Загрузить");

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

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

bLoadObjecExterntBinFormatMouseClicked(evt);

}

});

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

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

bLoadObjecExterntBinFormatActionPerformed(evt);

}

});

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

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

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

.addGroup(layout.createSequentialGroup()

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

.addGroup(layout.createSequentialGroup()

.addGap(6, 6, 6)

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

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

.addGroup(layout.createSequentialGroup()

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

.addComponent(textTXT, javax.swing.GroupLayout.PREFERRED\_SIZE, 270, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(textSer, javax.swing.GroupLayout.PREFERRED\_SIZE, 271, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(textExtern, javax.swing.GroupLayout.PREFERRED\_SIZE, 270, javax.swing.GroupLayout.PREFERRED\_SIZE))

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

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

.addGroup(layout.createSequentialGroup()

.addComponent(bSaveObjectTextFormat, javax.swing.GroupLayout.PREFERRED\_SIZE, 132, javax.swing.GroupLayout.PREFERRED\_SIZE)

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

.addComponent(bLoadObjectTextFormat, javax.swing.GroupLayout.PREFERRED\_SIZE, 132, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createSequentialGroup()

.addGap(1, 1, 1)

.addComponent(bSaveObjectSerBinFormat, javax.swing.GroupLayout.PREFERRED\_SIZE, 132, javax.swing.GroupLayout.PREFERRED\_SIZE)

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

.addComponent(bLoadObjectSerBinFormat, javax.swing.GroupLayout.PREFERRED\_SIZE, 132, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createSequentialGroup()

.addComponent(bSaveObjecExternBinFormat, javax.swing.GroupLayout.PREFERRED\_SIZE, 132, javax.swing.GroupLayout.PREFERRED\_SIZE)

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

.addComponent(bLoadObjecExterntBinFormat, javax.swing.GroupLayout.PREFERRED\_SIZE, 132, javax.swing.GroupLayout.PREFERRED\_SIZE))))

.addGroup(layout.createSequentialGroup()

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

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

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

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

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

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

.addGap(45, 45, 45)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 667, javax.swing.GroupLayout.PREFERRED\_SIZE))))

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(label1, javax.swing.GroupLayout.PREFERRED\_SIZE, 117, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(101, 101, 101)

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

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

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

.addGap(36, 36, 36)

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

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

.addComponent(label2, javax.swing.GroupLayout.PREFERRED\_SIZE, 117, javax.swing.GroupLayout.PREFERRED\_SIZE))))

.addContainerGap(19, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

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

.addGroup(layout.createSequentialGroup()

.addContainerGap()

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

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

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

.addComponent(label2, 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.BASELINE)

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

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

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

.addGap(29, 29, 29)

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

.addGroup(layout.createSequentialGroup()

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

.addGap(28, 28, 28)

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

.addGap(26, 26, 26)

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

.addGap(28, 28, 28)

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

.addGap(18, 18, 18)

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

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 0, Short.MAX\_VALUE))

.addGap(39, 39, 39)

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

.addGroup(layout.createSequentialGroup()

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

.addGap(20, 20, 20)

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

.addGap(19, 19, 19)

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

.addGroup(layout.createSequentialGroup()

.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))

.addGap(14, 14, 14)

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

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

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

.addGap(13, 13, 13)

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

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

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

.addContainerGap(72, Short.MAX\_VALUE))

);

textExtern.getAccessibleContext().setAccessibleDescription("");

pack();

}// </editor-fold>

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

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:

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

int indexROW = jTable1.getSelectedRow();

tbl.removeRow(indexROW);

listRecIntegral.remove(indexROW);

}

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;

try {

RecIntegral data = RecIntegralValidator.validateAndParse(

jTable1.getValueAt(selectRow, 0).toString(),

jTable1.getValueAt(selectRow, 1).toString(),

jTable1.getValueAt(selectRow, 2).toString()

);

CalculateIntegral integral = new CalculateIntegral(data);

double res = integral.calculate();

jTable1.getColumnModel().getColumn(3).setCellRenderer(new DoubleRenderer());

jTable1.setValueAt(res, selectRow, 3);

data.setRes(res);

listRecIntegral.set(selectRow + (listRecIntegral.size() - jTable1.getRowCount()), data);

} catch(ExceptionforRecIntegral ex){

javax.swing.JOptionPane.showMessageDialog(this,

ex.getMessage(),

"Ошибка",

javax.swing.JOptionPane.ERROR\_MESSAGE);

}

}

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

// TODO add your handling code here:

try{

RecIntegral data = RecIntegralValidator.validateAndParse(

jTextFieldNG.getText(),

jTextFieldVG.getText(),

jTextFieldSH.getText()

);

((DefaultTableModel) jTable1.getModel()).addRow(

new Object[]{data.getBottom(), data.getTop(), data.getStep()}

);

listRecIntegral.add(data);

}

catch(ExceptionforRecIntegral ex){

javax.swing.JOptionPane.showMessageDialog(this,

ex.getMessage(),

"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 tbl = (DefaultTableModel) jTable1.getModel();

for (int i = tbl.getRowCount(); i > 0; i--) {

tbl.removeRow(i - 1);

}

}

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 tbl = (DefaultTableModel) jTable1.getModel();

Object rowData[] = new Object[4];

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

rowData[0] = listRecIntegral.get(i).Bottom;

rowData[1] = listRecIntegral.get(i).Top;

rowData[2] = listRecIntegral.get(i).Step;

if(listRecIntegral.get(i).Res == 0) {

rowData[3] = null;

}

else {

rowData[3] = listRecIntegral.get(i).Res;

}

tbl.addRow(rowData);

}

}

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:

}

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

// TODO add your handling code here:

File file = fileManager.getPathTXTFileToSaved();

if (file == null) return;

SavedStateSerializable state = new SavedStateSerializable(listRecIntegral);

fileManager.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 = fileManager.getPathTXTFileToLoad();

if (file == null) return;

SavedStateSerializable state = fileManager.loadFromTextFile(file);

if (state != null) {

setDataToTable(state.getListRecIntegral());

setDataToArr(state.getListRecIntegral());

}

}

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

// TODO add your handling code here:

}

private void setDataToTable(ArrayList<RecIntegral> data) {

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

model.setRowCount(0);

for (RecIntegral row : data) {

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

model.addRow(new Object[]{row.getBottom(), row.getTop(), row.getStep()});

} else {

model.addRow(new Object[]{row.getBottom(), row.getTop(), row.getStep(), row.getRes()});

}

}

}

private void setDataToArr(ArrayList<RecIntegral> data) {

for (RecIntegral row : data) {

listRecIntegral.add(row);

}

}

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

// TODO add your handling code here:

File file = fileManager.getPathSerFileToSaved();

if (file == null) return;

SavedStateSerializable state = new SavedStateSerializable(listRecIntegral);

fileManager.saveToBinaryFile(file, state);

}

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

// TODO add your handling code here:

}

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

// TODO add your handling code here:

File file = fileManager.getPathSerFileToLoad();

if (file == null) return;

SavedStateSerializable state = fileManager.loadFromBinaryFile(file);

if (state != null) {

setDataToTable(state.getListRecIntegral());

setDataToArr(state.getListRecIntegral());

}

}

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

// TODO add your handling code here:

}

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

// TODO add your handling code here:

File file = fileManager.getPathExternFileToSaved();

if (file == null) return;

ExternalSavedState state = new ExternalSavedState(listRecIntegral);

fileManager.saveToBinaryExternFile(file, state);

}

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

// TODO add your handling code here:

}

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

// TODO add your handling code here:

File file = fileManager.getPathExternFileToLoad();

if (file == null) return;

ExternalSavedState state = fileManager.loadFromBinaryExternFile(file);

if (state != null) {

setDataToTable(state.getArrListTableData());

setDataToArr(state.getArrListTableData());

}

}

private void bLoadObjecExterntBinFormatActionPerformed(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 bLoadObjecExterntBinFormat;

private javax.swing.JButton bLoadObjectSerBinFormat;

private javax.swing.JButton bLoadObjectTextFormat;

private javax.swing.JButton bSaveObjecExternBinFormat;

private javax.swing.JButton bSaveObjectSerBinFormat;

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 textExtern;

private java.awt.Label textSer;

private java.awt.Label textTXT;

// End of variables declaration

}

**MyLab4.java**

package MyLab4;

public class MyLab4 {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

// TODO code application logic here

Frame frame = new Frame();

frame.setTitle("Калькулятор интегралов");

frame.show();

}

}

**RecIntegral.java**

package MyLab4;

import java.io.Serializable;

public class RecIntegral implements Serializable {

public double Top, Bottom, Step, Res;

public RecIntegral(double bottom, double top, double step) {

this.Top = top;

this.Bottom = bottom;

this.Step = step;

this.Res = Double.MAX\_VALUE;

}

public RecIntegral(double bottom, double top, double step, double res) {

this.Top = top;

this.Bottom = bottom;

this.Step = step;

this.Res = res;

}

public RecIntegral() {}

public double getBottom() {

return Bottom;

}

public void setBottom(double Bottom) {

this.Bottom = Bottom;

}

public double getTop() {

return Top;

}

public void setTop(double Top) {

this.Top = Top;

}

public double getStep() {

return Step;

}

public void setStep(double Step) {

this.Step = Step;

}

public double getRes() {

return Res;

}

public void setRes(double Res) {

this.Res = Res;

}

@Override

public String toString() {

return "RecIntegral{" +

"bottom=" + Bottom +

", Top=" + Top +

", Step=" + Step +

", Res=" + Res +

'}';

}

}

**RecIntegralValidator.java**

package MyLab4;

public class RecIntegralValidator {

public static RecIntegral validateAndParse(String lower, String upper, String step) throws ExceptionforRecIntegral {

double lowerVal = parseValue(lower);

double upperVal = parseValue(upper);

double stepVal = parseValue(step);

validateRange(lowerVal, upperVal, stepVal);

return new RecIntegral(lowerVal, upperVal, stepVal);

}

private static double parseValue(String input) throws ExceptionforRecIntegral {

try {

double value = Double.parseDouble(input);

if (value == 0 || value < 0.000001 || value > 1000000) {

throw new ExceptionforRecIntegral("Значение должно быть от 0.000001 до 1000000");

}

return value;

} catch (NumberFormatException e) {

throw new ExceptionforRecIntegral("Некорректный числовой формат");

}

}

private static void validateRange(double lower, double upper, double step) throws ExceptionforRecIntegral {

if (step <= 0) {

throw new ExceptionforRecIntegral("Шаг должен быть положительным");

}

if (Math.abs(upper - lower) < step) {

throw new ExceptionforRecIntegral("Интервал должен быть не меньше шага");

}

}

}

**SavedStateSerializable.java**

package MyLab4;

import java.io.Serializable;

import java.util.ArrayList;

public class SavedStateSerializable implements Serializable {

private static final long serialVersionUID = 2L;

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

public SavedStateSerializable(ArrayList<RecIntegral> listRecIntegral) {

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

}

public ArrayList<RecIntegral> getListRecIntegral() {

return new ArrayList<>(listRecIntegral);

}

@Override

public String toString() {

return "SavedStateSerializable{" +

"listRecIntegral=" + listRecIntegral +

'}';

}

}

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