Министерство науки и высшего образования РФ

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

Беляев Д. И.

Демин М. С.

Приняли:

Юрова О. В.

Карамышева Н. С.

Пенза 2025

**Название**

Работа с файлами

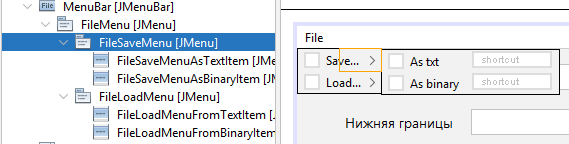
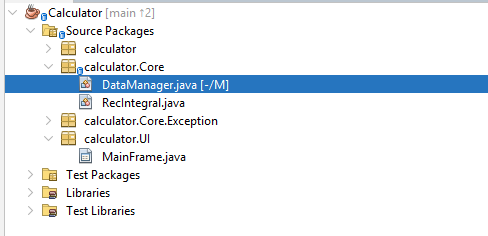
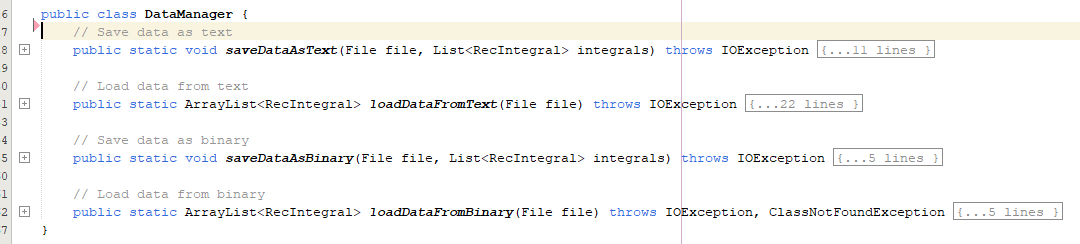
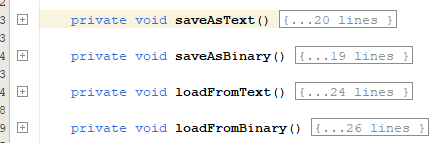
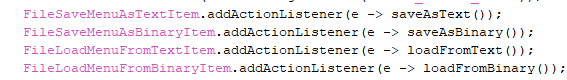
**Цель работы**

Изучить работу с файлами и механизмы сериализации данных

**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.

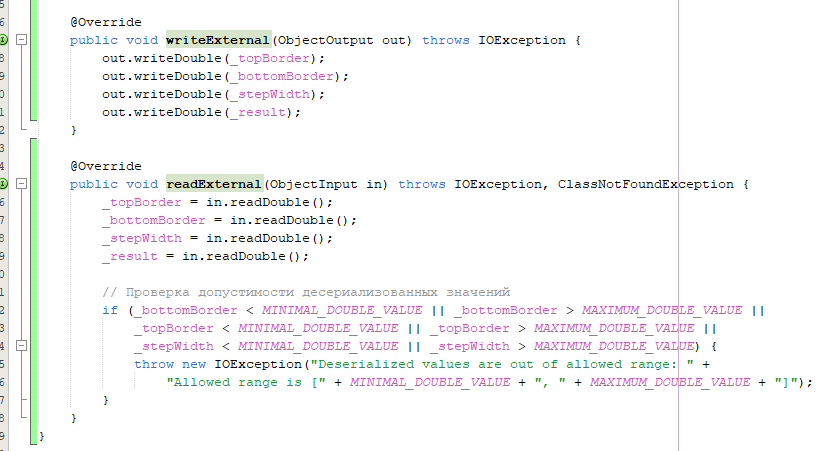
**Ход работы**

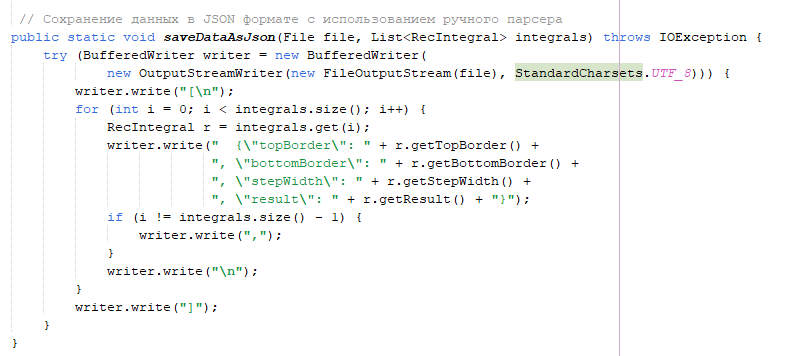
1. Добавили кнопки сохранения и загрузки  
   ****
2. Добавили класс, который отвечает за работу с файлами  
   ****  
   ****
3. Привязали кнопки к функциям  
   **** 
4. Имплементировали интерефейс Serializable для сериализации данных  
   ****

*Дополнительное задание*

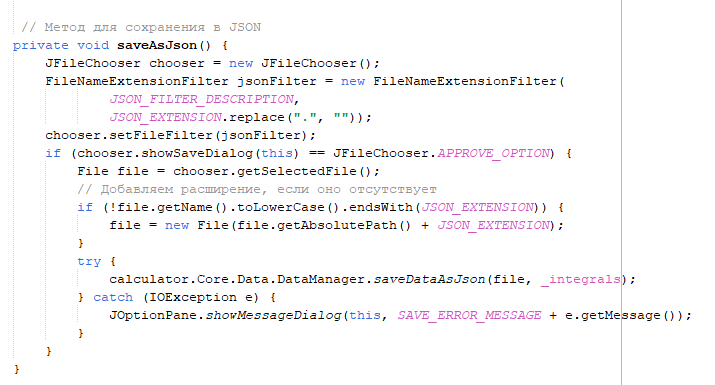
Реализовали Externalizable

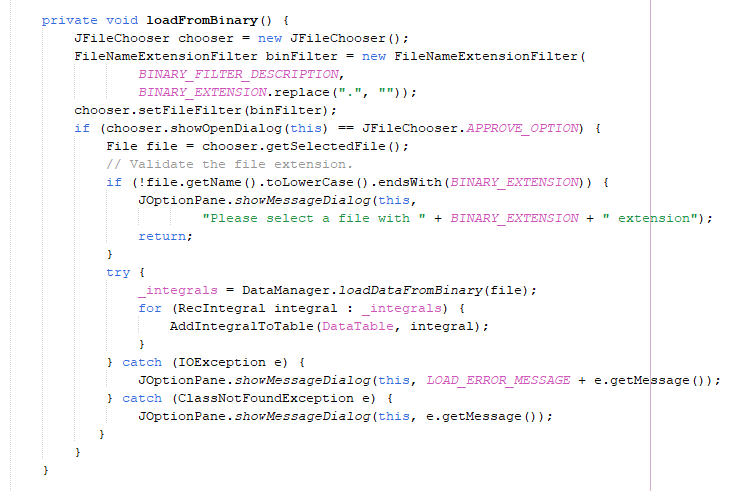




Добавили работу с json  








**Листинг**

*Calculator.java*

package calculator;

public class Calculator {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

MainFrame mainFrame = new MainFrame();

mainFrame.show();

}

}

*MainFrame.java*

package calculator.UI;

import calculator.Core.DataManager;

import calculator.Core.Exception.IntegralValueException;

import calculator.Core.RecIntegral;

import calculator.Core.Exception.StepException;

import java.io.File;

import java.io.IOException;

import java.util.ArrayList;

import javax.swing.JFileChooser;

import javax.swing.JOptionPane;

import javax.swing.JTable;

import javax.swing.filechooser.FileNameExtensionFilter;

import javax.swing.table.DefaultTableModel;

public class MainFrame extends javax.swing.JFrame {

private final int SHADOW\_COLUMN\_NUMBER = 4;

private final String SHADOW\_COLUMN\_TITLE = "ShadowColumn";

// Constants for file extensions and filter descriptions

private static final String TEXT\_EXTENSION = ".txt";

private static final String BINARY\_EXTENSION = ".calcbin";

private static final String TEXT\_FILTER\_DESCRIPTION = "Text Files (\*" + TEXT\_EXTENSION + ")";

private static final String BINARY\_FILTER\_DESCRIPTION = "Calc Binary Files (\*" + BINARY\_EXTENSION + ")";

// Constants for messages

private static final String SAVE\_ERROR\_MESSAGE = "Save error: ";

private static final String LOAD\_ERROR\_MESSAGE = "Load error: ";

private ArrayList<RecIntegral> \_integrals;

/\*\*

\* Creates new form MainFrame

\*/

public MainFrame() {

initComponents();

\_integrals = new ArrayList<>();

DataTable.removeColumn(DataTable.getColumn(SHADOW\_COLUMN\_TITLE));

FileSaveMenuAsTextItem.addActionListener(e -> saveAsText());

FileSaveMenuAsBinaryItem.addActionListener(e -> saveAsBinary());

FileLoadMenuFromTextItem.addActionListener(e -> loadFromText());

FileLoadMenuFromBinaryItem.addActionListener(e -> loadFromBinary());

}

@SuppressWarnings("unchecked")

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

private void initComponents() {

jMenuItem1 = new javax.swing.JMenuItem();

DataTableScrollPanel = new javax.swing.JScrollPane();

DataTable = new javax.swing.JTable();

TopBorderLabel = new javax.swing.JLabel();

BottomBorderLabel = new javax.swing.JLabel();

StepWidthLabel = new javax.swing.JLabel();

TopBorderTextField = new javax.swing.JTextField();

BottomBorderTextField = new javax.swing.JTextField();

StepWidthTextField = new javax.swing.JTextField();

AddButton = new javax.swing.JButton();

DeleteButton = new javax.swing.JButton();

CalculateButton = new javax.swing.JButton();

ClearTableButton = new javax.swing.JButton();

FillTableButton = new javax.swing.JButton();

MenuBar = new javax.swing.JMenuBar();

FileMenu = new javax.swing.JMenu();

FileSaveMenu = new javax.swing.JMenu();

FileSaveMenuAsTextItem = new javax.swing.JMenuItem();

FileSaveMenuAsBinaryItem = new javax.swing.JMenuItem();

FileLoadMenu = new javax.swing.JMenu();

FileLoadMenuFromTextItem = new javax.swing.JMenuItem();

FileLoadMenuFromBinaryItem = new javax.swing.JMenuItem();

jMenuItem1.setText("jMenuItem1");

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

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

new Object [][] {

},

new String [] {

"Верхняя граница", "Нижняя граница", "Ширина шага", "Результат", "ShadowColumn"

}

) {

boolean[] canEdit = new boolean [] {

false, false, false, false, false

};

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

DataTableScrollPanel.setViewportView(DataTable);

TopBorderLabel.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N

TopBorderLabel.setText("Верхняя граница");

TopBorderLabel.setName(""); // NOI18N

BottomBorderLabel.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N

BottomBorderLabel.setText("Ширина шага");

StepWidthLabel.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N

StepWidthLabel.setText("Нижняя границы");

TopBorderTextField.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N

BottomBorderTextField.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N

StepWidthTextField.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N

AddButton.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N

AddButton.setText("Добавить");

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

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

AddButtonMouseClicked(evt);

}

});

DeleteButton.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N

DeleteButton.setText("Удалить");

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

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

DeleteButtonMouseClicked(evt);

}

});

CalculateButton.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N

CalculateButton.setText("Вычислить");

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

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

CalculateButtonMouseClicked(evt);

}

});

ClearTableButton.setText("Очистить");

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

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

ClearTableButtonMouseClicked(evt);

}

});

FillTableButton.setText("Восстановить");

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

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

FillTableButtonMouseClicked(evt);

}

});

FileMenu.setText("File");

FileSaveMenu.setText("Save...");

FileSaveMenuAsTextItem.setText("As txt");

FileSaveMenu.add(FileSaveMenuAsTextItem);

FileSaveMenuAsBinaryItem.setText("As binary");

FileSaveMenu.add(FileSaveMenuAsBinaryItem);

FileMenu.add(FileSaveMenu);

FileLoadMenu.setText("Load...");

FileLoadMenuFromTextItem.setText("From txt");

FileLoadMenu.add(FileLoadMenuFromTextItem);

FileLoadMenuFromBinaryItem.setText("From binary");

FileLoadMenu.add(FileLoadMenuFromBinaryItem);

FileMenu.add(FileLoadMenu);

MenuBar.add(FileMenu);

setJMenuBar(MenuBar);

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

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

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

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

.addGap(48, 48, 48)

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

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

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

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

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

.addGroup(layout.createSequentialGroup()

.addGap(18, 18, 18)

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

.addComponent(StepWidthTextField, javax.swing.GroupLayout.DEFAULT\_SIZE, 314, Short.MAX\_VALUE)

.addComponent(BottomBorderTextField)))

.addGroup(layout.createSequentialGroup()

.addGap(16, 16, 16)

.addComponent(TopBorderTextField)))

.addGap(18, 18, 18)

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

.addComponent(CalculateButton, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(DeleteButton, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(AddButton, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addGap(48, 48, 48))

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(DataTableScrollPanel)

.addContainerGap())

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

.addContainerGap()

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

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

.addComponent(FillTableButton, javax.swing.GroupLayout.PREFERRED\_SIZE, 317, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap())

);

layout.setVerticalGroup(

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

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

.addGap(15, 15, 15)

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

.addComponent(TopBorderLabel)

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

.addComponent(AddButton))

.addGap(18, 18, 18)

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

.addComponent(DeleteButton, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.PREFERRED\_SIZE, 27, javax.swing.GroupLayout.PREFERRED\_SIZE)

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

.addComponent(StepWidthLabel)

.addComponent(BottomBorderTextField, 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(BottomBorderLabel)

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

.addComponent(CalculateButton))

.addGap(18, 18, 18)

.addComponent(DataTableScrollPanel, javax.swing.GroupLayout.PREFERRED\_SIZE, 122, javax.swing.GroupLayout.PREFERRED\_SIZE)

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

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

.addComponent(ClearTableButton)

.addComponent(FillTableButton))

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

);

pack();

}// </editor-fold>

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

// Check if all fields field

String topBorderFieldText = TopBorderTextField.getText();

String bottomBordeFieldText = BottomBorderTextField.getText();

String stepWidthFieldText = StepWidthTextField.getText();

if (stringIsNullOrEmpty(topBorderFieldText) ||

stringIsNullOrEmpty(bottomBordeFieldText) ||

stringIsNullOrEmpty(stepWidthFieldText)) {

return;

}

// Try to convert

double topBorder;

double bottomBorder;

double stepWidth;

try {

topBorder = Double.parseDouble(topBorderFieldText);

bottomBorder = Double.parseDouble(bottomBordeFieldText);

stepWidth = Double.parseDouble(stepWidthFieldText);

} catch (NumberFormatException e) {

return;

}

// Step must be greater then zero

if (stepWidth <= 0) {

return;

}

// Add data to table

RecIntegral integral;

try {

integral = new RecIntegral(topBorder, bottomBorder, stepWidth);

} catch(IntegralValueException ex){

JOptionPane.showMessageDialog(

this,

ex.getMessage(),

ex.getExceptionName(),

JOptionPane.ERROR\_MESSAGE);

return;

}

\_integrals.add(integral);

AddIntegralToTable(DataTable, integral);

}

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

// Check if row selected

int selectedRow = DataTable.getSelectedRow();

if (selectedRow < 0) {

return;

}

// Remove row

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

\_integrals.remove(

(RecIntegral)model.getValueAt(selectedRow, SHADOW\_COLUMN\_NUMBER));

model.removeRow(selectedRow);

}

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

// Check if row selected

int selectedRow = DataTable.getSelectedRow();

if (selectedRow < 0) {

return;

}

// Check if selected row not empty

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

RecIntegral integral =

(RecIntegral)model.getValueAt(selectedRow, SHADOW\_COLUMN\_NUMBER);

// Calculate result

try {

integral.calculateIntegral();

} catch(StepException ex){

JOptionPane.showMessageDialog(

this,

ex.getMessage(),

ex.getExceptionName(),

JOptionPane.ERROR\_MESSAGE);

return;

}

// Set result to table

model.setValueAt(integral.getResult(), selectedRow, 3);

}

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

((DefaultTableModel) DataTable.getModel()).setRowCount(0);

}

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

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

model.setRowCount(0);

for (RecIntegral integral : \_integrals) {

AddIntegralToTable(DataTable, integral);

}

}

private void saveAsText() {

JFileChooser chooser = new JFileChooser();

FileNameExtensionFilter textFilter =

new FileNameExtensionFilter(

"Text Files (\*" + TEXT\_EXTENSION + ")",

TEXT\_EXTENSION.replace(".", ""));

chooser.setFileFilter(textFilter);

if (chooser.showSaveDialog(this) == JFileChooser.APPROVE\_OPTION) {

File file = chooser.getSelectedFile();

// Append the .txt extension if it is missing.

if (!file.getName().toLowerCase().endsWith(TEXT\_EXTENSION)) {

file = new File(file.getAbsolutePath() + TEXT\_EXTENSION);

}

try {

DataManager.saveDataAsText(file, \_integrals);

} catch (IOException e) {

JOptionPane.showMessageDialog(this, SAVE\_ERROR\_MESSAGE + e.getMessage());

}

}

}

private void saveAsBinary() {

JFileChooser chooser = new JFileChooser();

FileNameExtensionFilter binFilter = new FileNameExtensionFilter(

BINARY\_FILTER\_DESCRIPTION,

BINARY\_EXTENSION.replace(".", ""));

chooser.setFileFilter(binFilter);

if (chooser.showSaveDialog(this) == JFileChooser.APPROVE\_OPTION) {

File file = chooser.getSelectedFile();

// Append the .calcbin extension if it is missing.

if (!file.getName().toLowerCase().endsWith(BINARY\_EXTENSION)) {

file = new File(file.getAbsolutePath() + BINARY\_EXTENSION);

}

try {

DataManager.saveDataAsBinary(file, \_integrals);

} catch (IOException e) {

JOptionPane.showMessageDialog(this, SAVE\_ERROR\_MESSAGE + e.getMessage());

}

}

}

private void loadFromText() {

JFileChooser chooser = new JFileChooser();

FileNameExtensionFilter textFilter = new FileNameExtensionFilter(

TEXT\_FILTER\_DESCRIPTION,

TEXT\_EXTENSION.replace(".", ""));

chooser.setFileFilter(textFilter);

if (chooser.showOpenDialog(this) == JFileChooser.APPROVE\_OPTION) {

File file = chooser.getSelectedFile();

// Validate the file extension.

if (!file.getName().toLowerCase().endsWith(TEXT\_EXTENSION)) {

JOptionPane.showMessageDialog(this,

"Please select a file with " + TEXT\_EXTENSION + " extension");

return;

}

try {

\_integrals = DataManager.loadDataFromText(file);

for (RecIntegral integral : \_integrals) {

AddIntegralToTable(DataTable, integral);

}

} catch (IOException e) {

JOptionPane.showMessageDialog(this, LOAD\_ERROR\_MESSAGE + e.getMessage());

}

}

}

private void loadFromBinary() {

JFileChooser chooser = new JFileChooser();

FileNameExtensionFilter binFilter = new FileNameExtensionFilter(

BINARY\_FILTER\_DESCRIPTION,

BINARY\_EXTENSION.replace(".", ""));

chooser.setFileFilter(binFilter);

if (chooser.showOpenDialog(this) == JFileChooser.APPROVE\_OPTION) {

File file = chooser.getSelectedFile();

// Validate the file extension.

if (!file.getName().toLowerCase().endsWith(BINARY\_EXTENSION)) {

JOptionPane.showMessageDialog(this,

"Please select a file with " + BINARY\_EXTENSION + " extension");

return;

}

try {

\_integrals = DataManager.loadDataFromBinary(file);

for (RecIntegral integral : \_integrals) {

AddIntegralToTable(DataTable, integral);

}

} catch (IOException e) {

JOptionPane.showMessageDialog(this, LOAD\_ERROR\_MESSAGE + e.getMessage());

} catch (ClassNotFoundException e) {

JOptionPane.showMessageDialog(this, e.getMessage());

}

}

}

private void AddIntegralToTable(JTable table, RecIntegral integral) {

double result = integral.getResult();

((DefaultTableModel)table.getModel()).addRow(

new Object[]{

integral.getTopBorder(),

integral.getBottomBorder(),

integral.getStepWidth(),

result == Double.NaN ? "" : result,

integral});

}

private boolean stringIsNullOrEmpty(String str) {

return str == null || str.trim().isEmpty();

}

// Variables declaration - do not modify

private javax.swing.JButton AddButton;

private javax.swing.JLabel BottomBorderLabel;

private javax.swing.JTextField BottomBorderTextField;

private javax.swing.JButton CalculateButton;

private javax.swing.JButton ClearTableButton;

private javax.swing.JTable DataTable;

private javax.swing.JScrollPane DataTableScrollPanel;

private javax.swing.JButton DeleteButton;

private javax.swing.JMenu FileLoadMenu;

private javax.swing.JMenuItem FileLoadMenuFromBinaryItem;

private javax.swing.JMenuItem FileLoadMenuFromTextItem;

private javax.swing.JMenu FileMenu;

private javax.swing.JMenu FileSaveMenu;

private javax.swing.JMenuItem FileSaveMenuAsBinaryItem;

private javax.swing.JMenuItem FileSaveMenuAsTextItem;

private javax.swing.JButton FillTableButton;

private javax.swing.JMenuBar MenuBar;

private javax.swing.JLabel StepWidthLabel;

private javax.swing.JTextField StepWidthTextField;

private javax.swing.JLabel TopBorderLabel;

private javax.swing.JTextField TopBorderTextField;

private javax.swing.JMenuItem jMenuItem1;

// End of variables declaration

}

*RecIntegral.java*

package calculator.Core;

import calculator.Core.Exception.StepException;

import calculator.Core.Exception.IntegralValueException;

import java.io.Serializable;

public class RecIntegral implements Serializable {

private static final long serialVersionUID = 1L;

private static final double MINIMAL\_DOUBLE\_VALUE = 0.000001;

private static final double MAXIMUM\_DOUBLE\_VALUE = 1000000;

private double \_topBorder = 0.0;

private double \_bottomBorder = 0.0;

private double \_stepWidth = 0.0;

private double \_result = 0.0;

public RecIntegral(double topBorder, double bottomBorder, double stepWidth) throws IntegralValueException {

this(topBorder, bottomBorder, stepWidth, Double.NaN);

}

public RecIntegral(

double topBorder,

double bottomBorder,

double stepWidth,

double previewResult)

throws IntegralValueException {

\_topBorder = topBorder;

\_bottomBorder = bottomBorder;

// Check of values in range

if (bottomBorder < MINIMAL\_DOUBLE\_VALUE || bottomBorder > MAXIMUM\_DOUBLE\_VALUE ||

topBorder < MINIMAL\_DOUBLE\_VALUE || topBorder > MAXIMUM\_DOUBLE\_VALUE ||

stepWidth < MINIMAL\_DOUBLE\_VALUE || stepWidth > MAXIMUM\_DOUBLE\_VALUE) {

throw new IntegralValueException(

MINIMAL\_DOUBLE\_VALUE, MAXIMUM\_DOUBLE\_VALUE);

}

\_stepWidth = stepWidth;

\_result = previewResult;

}

public double getTopBorder() { return \_topBorder; }

public double getBottomBorder() { return \_bottomBorder; }

public double getStepWidth() { return \_stepWidth; }

public double getResult() { return \_result; }

public void calculateIntegral() throws StepException {

double sign = 1.0;

// If the top border is less than the bottom border, swap them and invert the sign of the result

if (\_topBorder < \_bottomBorder) {

double temp = \_topBorder;

\_topBorder = \_bottomBorder;

\_bottomBorder = temp;

sign = -1.0;

}

if (\_stepWidth > \_topBorder - \_bottomBorder) {

throw new StepException(\_stepWidth, \_bottomBorder, \_topBorder);

}

double sum = 0.0;

// Integrate using the trapezoidal rule

for (double x = \_bottomBorder; x < \_topBorder; x += \_stepWidth) {

double nextX = Math.min(x + \_stepWidth, \_topBorder);

// Calculate the area of the trapezoid between x and nextX

double area = (nextX - x) \* (Math.exp(-x) + Math.exp(-nextX)) / 2.0;

sum += area;

}

\_result = sign \* sum;

}

}

*DataManager.java*

package calculator.Core;

import java.io.BufferedReader;

import java.io.BufferedWriter;

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 java.util.List;

public class DataManager {

// Save data as text

public static void saveDataAsText(File file, List<RecIntegral> integrals) throws IOException {

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

for (RecIntegral integral : integrals) {

writer.write(integral.getTopBorder() + ";"

+ integral.getBottomBorder() + ";"

+ integral.getStepWidth() + ";"

+ integral.getResult());

writer.newLine();

}

}

}

// Load data from text

public static ArrayList<RecIntegral> loadDataFromText(File file) throws IOException {

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

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

String line;

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

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

if (parts.length < 4) {

continue;

}

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

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

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

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

RecIntegral integral = new RecIntegral(top, bottom, step, result);

integrals.add(integral);

}

} catch (Exception e) {

throw new IOException("Error when load data: " + e.getMessage());

}

return integrals;

}

// Save data as binary

public static void saveDataAsBinary(File file, List<RecIntegral> integrals) throws IOException {

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

oos.writeObject(integrals);

}

}

// Load data from binary

public static ArrayList<RecIntegral> loadDataFromBinary(File file) throws IOException, ClassNotFoundException {

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

return (ArrayList<RecIntegral>) ois.readObject();

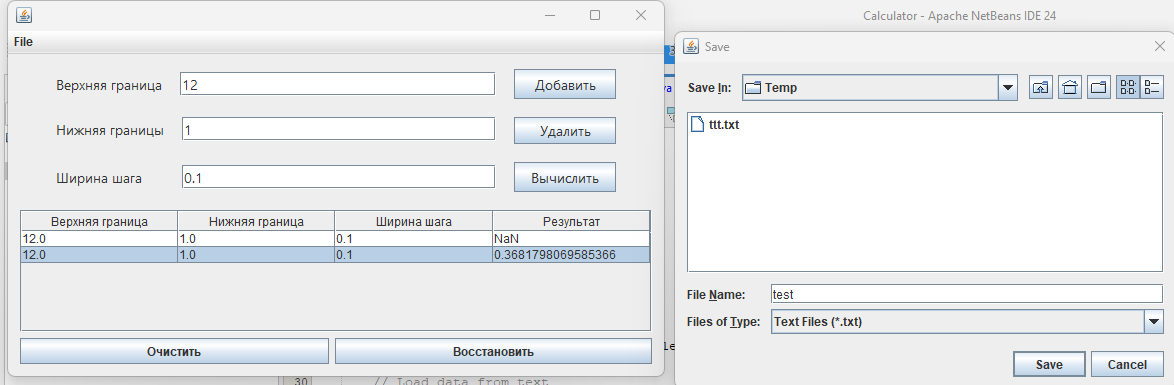
}

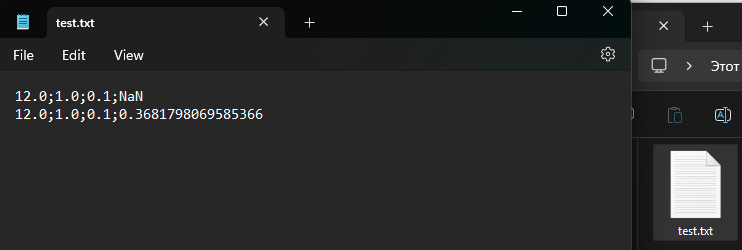
}

}

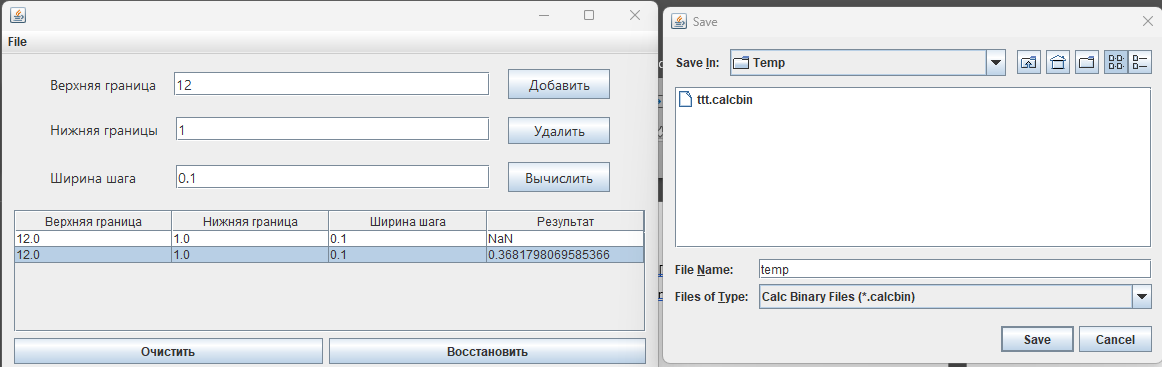
**Результат выполнения программы**

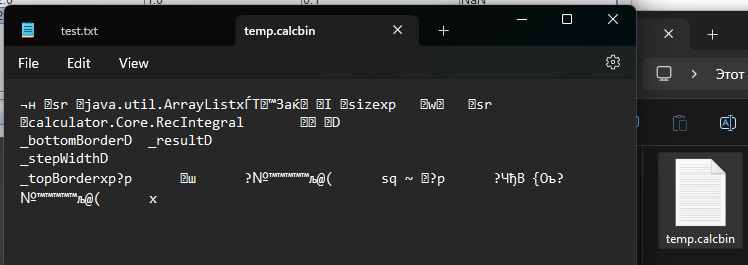
Сохранение в текстовом формате

****

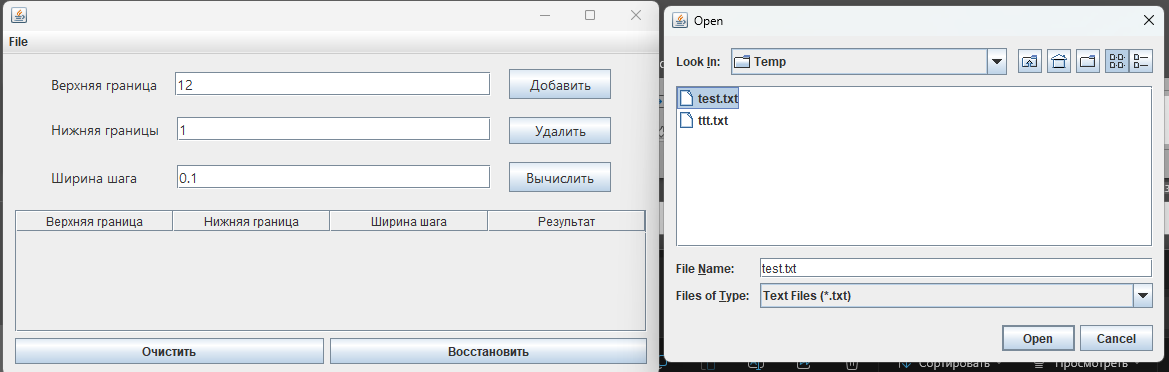
****

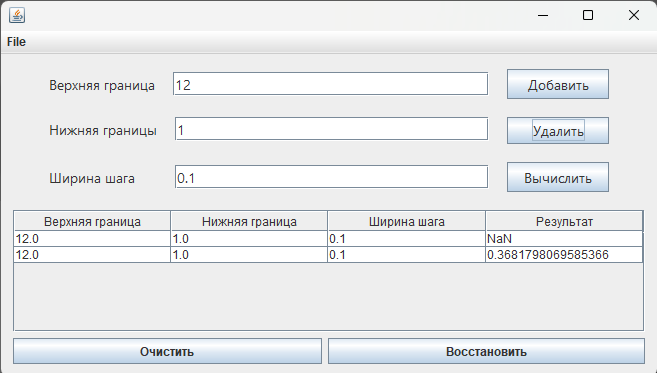
Сохранение в бинарном формате

****

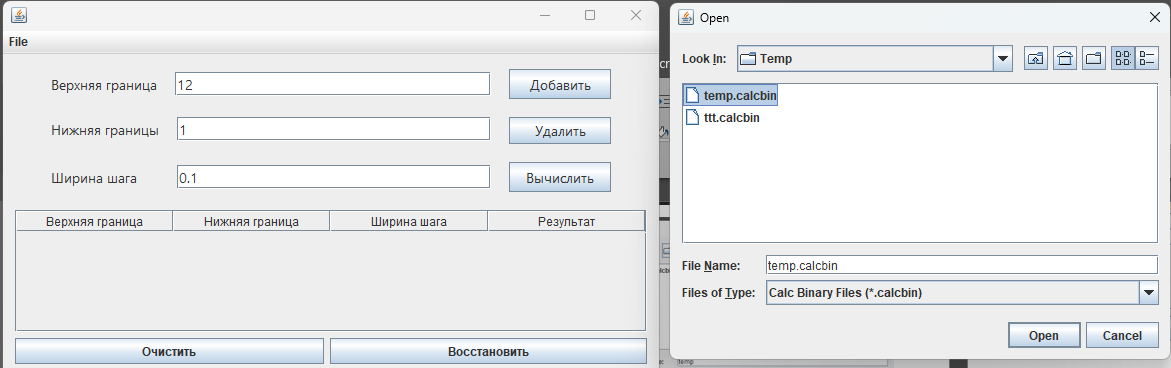
****

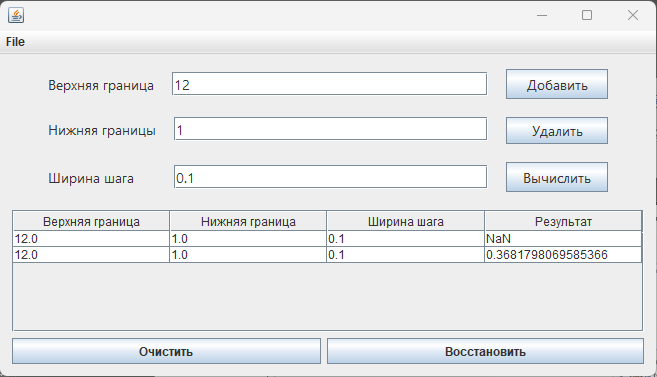
Загрузка из .txt файла

****

****

Загрузка из .calcbin файла

****

****

**Вывод**

Изучили работу с файлами и механизмы сериализации данных