Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №3

по курсу «Разработка кроссплатформенных приложений»

на тему «Обработка исключительных ситуаций»

**Выполнил студент группы 20ВВВ1:**

Ионцев В.А.

**Приняли:**

Юрова О.В.

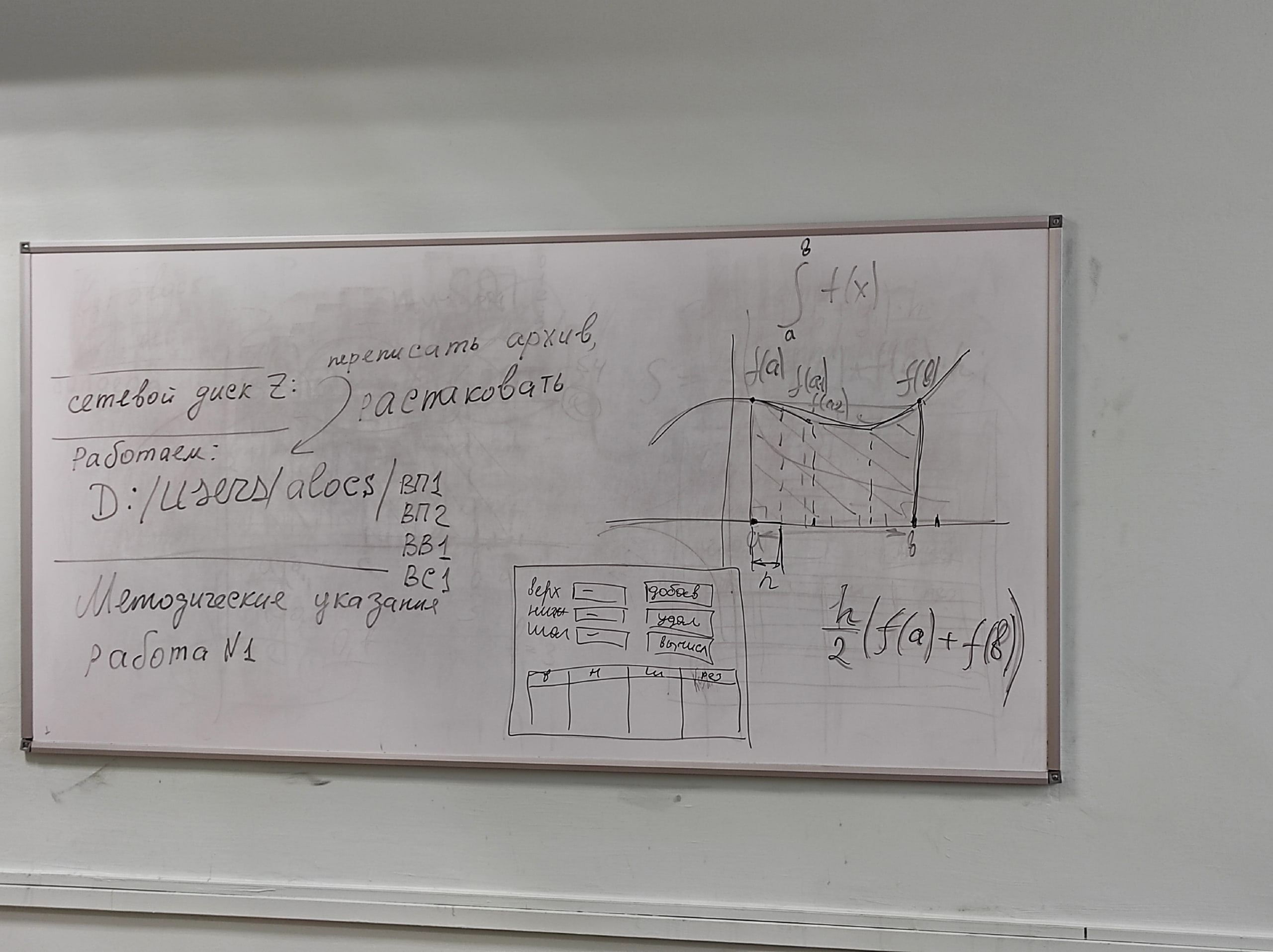
Карамышева Н.С.

Пенза 2023 г.

**Цель работы:** изучить механизм обработки исключительных ситуаций.

**Задание:**

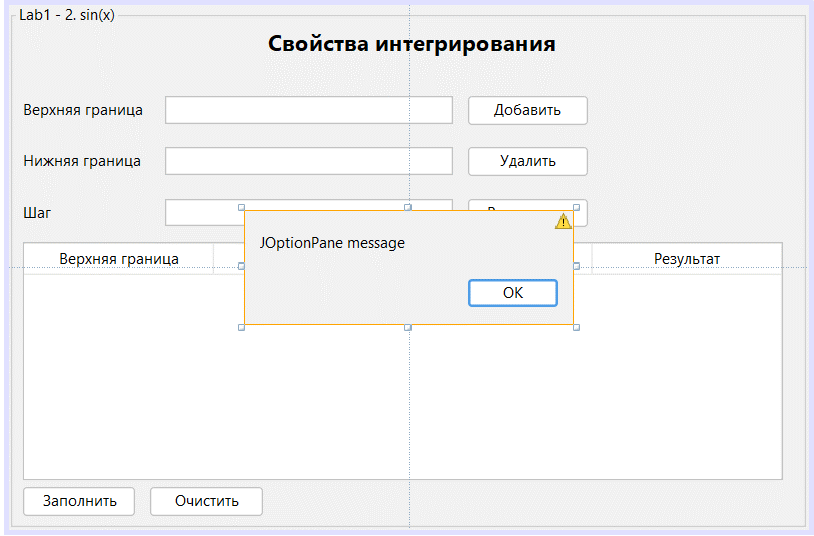
|  |  |
| --- | --- |
| Номер варианта | Функция |
| 2 |  |



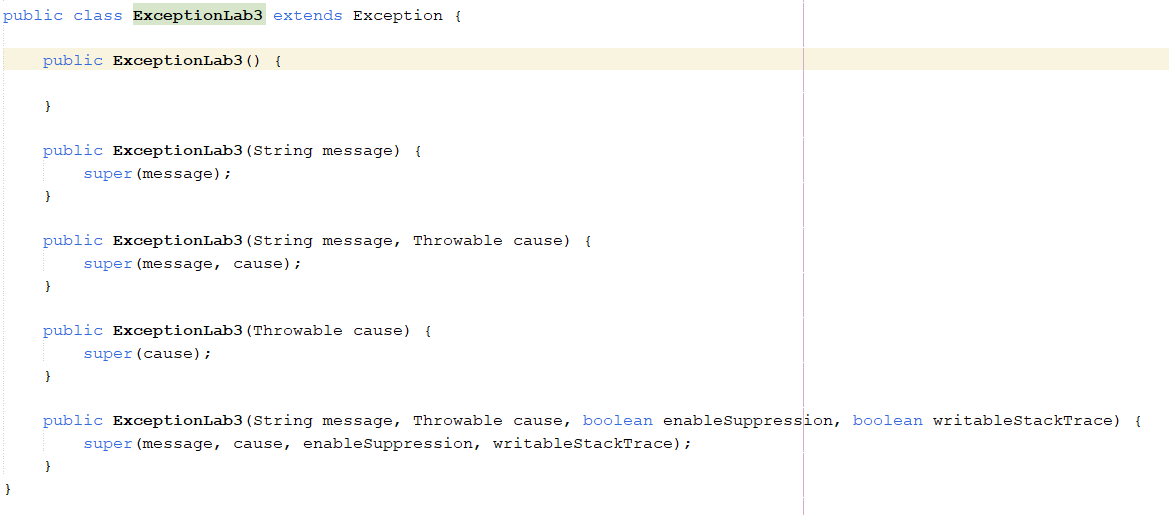
Модифицировать приложение из предыдущей лабораторной работы, реализовав проверку вводимых данных с использованием механизма исключений. Необходимо создать свой класс, унаследованный от класса Exception, и генерировать исключение, если возникает попытка создать экземпляр класса RecIntegral со значениями, не являющимися числами в диапазоне от 0,000001 до 1000000. В качестве обработки исключения необходимо выводить диалог, содержащий предупреждение о некорректности введенных данных. Оформление лабораторной работы должно быть выполнено в соответствии с требованиями, приведенными в Приложении 2.

**Ход работы:**

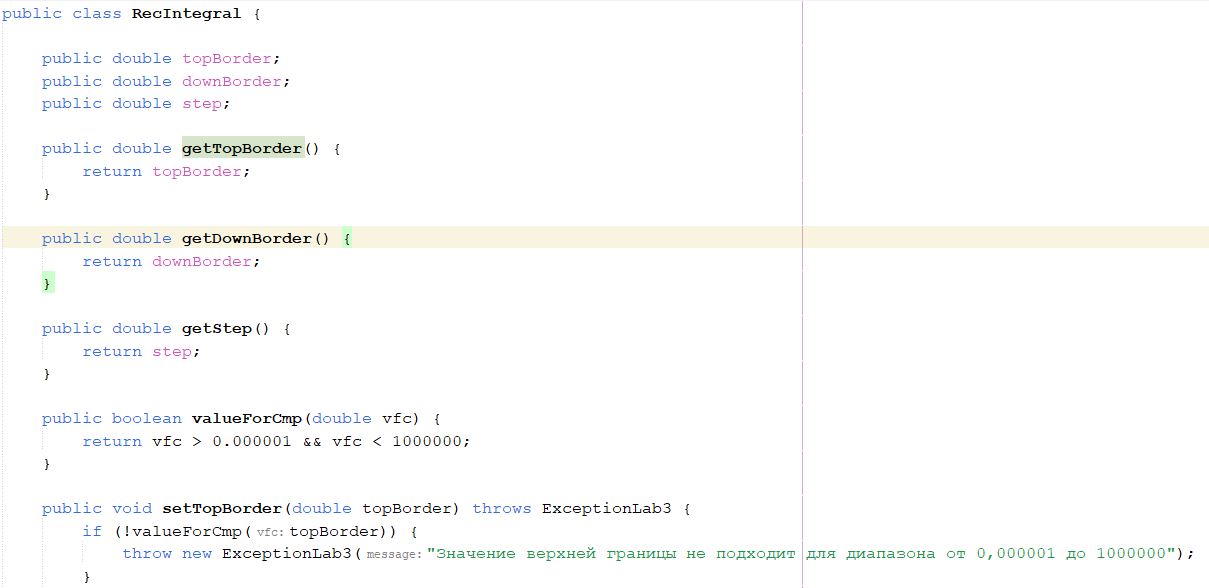
1. Добавили элемент JOptionPane в интерфейс приложения.

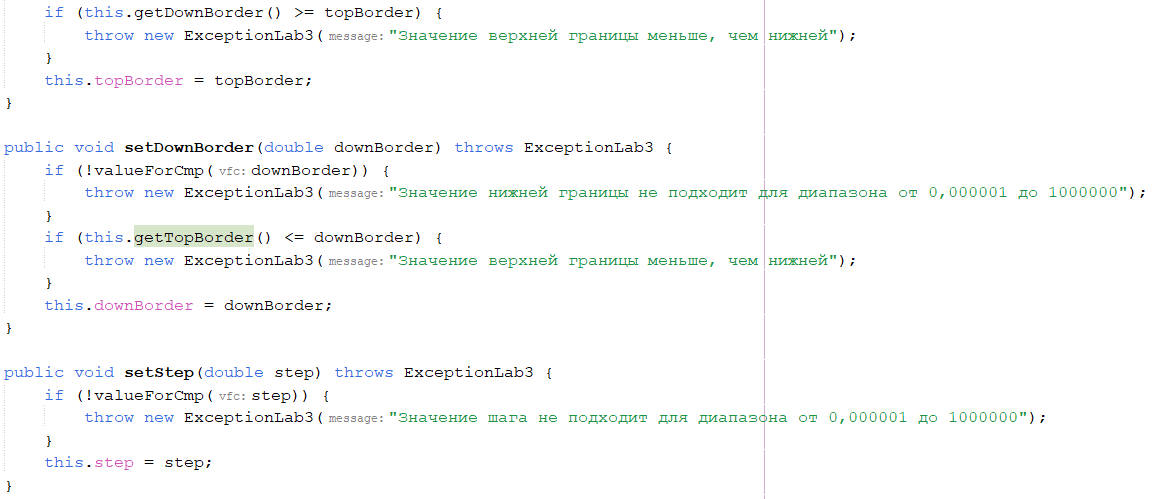
****

1. Создали новый класс ExceptionLab3.



1. Дополнили класс RecIntegral.





1. Дополнили функционал кнопки добавить.



**Листинг:**

**RecIntegral.java**

package my.numberaddition;

import javax.swing.JOptionPane;

/\*\*

\*

\* @author Acer

\*/

public class RecIntegral {

public double topBorder;

public double downBorder;

public double step;

public double getTopBorder() {

return topBorder;

}

public double getDownBorder() {

return downBorder;

}

public double getStep() {

return step;

}

public double getIterationResult() {

return iterationResult;

}

public boolean valueForCmp(double vfc) {

return vfc > 0.000001 && vfc < 1000000;

}

public void setTopBorder(double topBorder) throws ExceptionLab3 {

if (!valueForCmp(topBorder)) {

throw new ExceptionLab3("Значение верхней границы не подходит для диапазона от 0,000001 до 1000000");

}

if (this.getDownBorder() >= topBorder) {

throw new ExceptionLab3("Значение верхней границы меньше, чем нижней");

}

this.topBorder = topBorder;

}

public void setDownBorder(double downBorder) throws ExceptionLab3 {

if (!valueForCmp(downBorder)) {

throw new ExceptionLab3("Значение нижней границы не подходит для диапазона от 0,000001 до 1000000");

}

if (this.getTopBorder() <= downBorder) {

throw new ExceptionLab3("Значение верхней границы меньше, чем нижней");

}

this.downBorder = downBorder;

}

public void setStep(double step) throws ExceptionLab3 {

if (!valueForCmp(step)) {

throw new ExceptionLab3("Значение шага не подходит для диапазона от 0,000001 до 1000000");

}

this.step = step;

}

public void setIterationResult(double iterationResult) throws ExceptionLab3 {

if (this.getStep() >= iterationResult) {

throw new ExceptionLab3("Значение шага больше значения h");

}

this.iterationResult = iterationResult;

}

public RecIntegral(double topBorder, double downBorder, double step) {

this.topBorder = topBorder;

this.downBorder = downBorder;

this.step = step;

}

public RecIntegral() {

this.topBorder = 0;

this.downBorder = 0;

this.step = 0;

}

}

**ContactEditorUI.java**

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

// TODO add your handling code here:

// First we define float variables.

double topBorder, downBorder, step;

topBorder = Double.parseDouble(jTextField1.getText());

downBorder = Double.parseDouble(jTextField2.getText());

step = Double.parseDouble(jTextField3.getText());

RecIntegral ri = new RecIntegral();

try {

ri.setTopBorder(topBorder);

ri.setDownBorder(downBorder);

ri.setStep(step);

list.add(ri);

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

myTable.addRow(new Object[]{topBorder, downBorder, step});

} catch (ExceptionLab3 ex) {

JOptionPane.showMessageDialog(null, ex.getMessage(), "Ошибка", JOptionPane.ERROR\_MESSAGE);

}

}

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

// TODO add your handling code here:

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

int rowCount = myTable.getRowCount();

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

try {

double topBorder, downBorder, step;

topBorder = Double.parseDouble(myTable.getValueAt(i, 0).toString());

downBorder = Double.parseDouble(myTable.getValueAt(i, 1).toString());

step = Double.parseDouble(myTable.getValueAt(i, 2).toString());

RecIntegral ri = new RecIntegral();

ri.setStep(step);

if ((downBorder > topBorder) || (downBorder == 0) || (topBorder == 0) || (step == 0)) {

return;

}

if ((downBorder < 0) && (topBorder > 0)) {

myTable.setValueAt("Infinity", i, 3);

} else {

double nextStep = downBorder;

double prevStep = downBorder;

double result = 0;

do {

nextStep = nextStep + step;

if (nextStep > topBorder) {

nextStep = topBorder;

}

double iterationResult = f(nextStep) + f(prevStep);

ri.setIterationResult(iterationResult);

iterationResult = iterationResult \* (nextStep - prevStep) / 2;

result += iterationResult;

} while (nextStep != topBorder);

myTable.setValueAt(result, i, 3);

}

} catch (ExceptionLab3 ex) {

JOptionPane.showMessageDialog(null, ex.getMessage(), "Ошибка", JOptionPane.ERROR\_MESSAGE);

}

}

}

**ExceptionLab3.Java**

public class ExceptionLab3 extends Exception {

public ExceptionLab3() {

}

public ExceptionLab3(String message) {

super(message);

}

public ExceptionLab3(String message, Throwable cause) {

super(message, cause);

}

public ExceptionLab3(Throwable cause) {

super(cause);

}

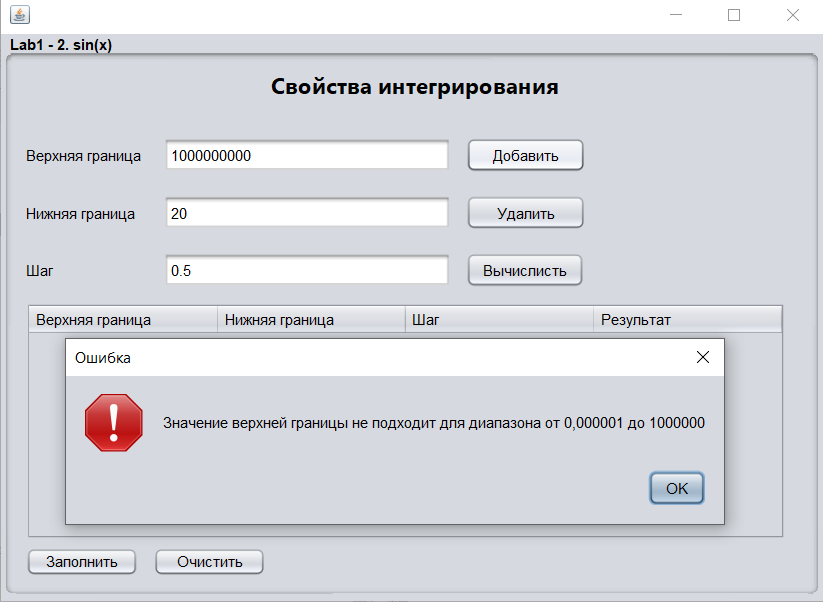
public ExceptionLab3(String message, Throwable cause, boolean enableSuppression, boolean writableStackTrace) {

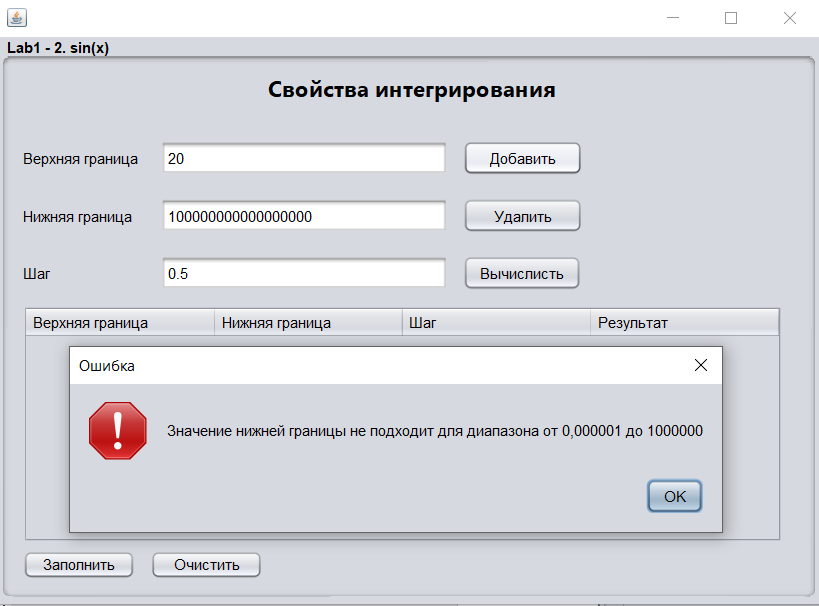
super(message, cause, enableSuppression, writableStackTrace);

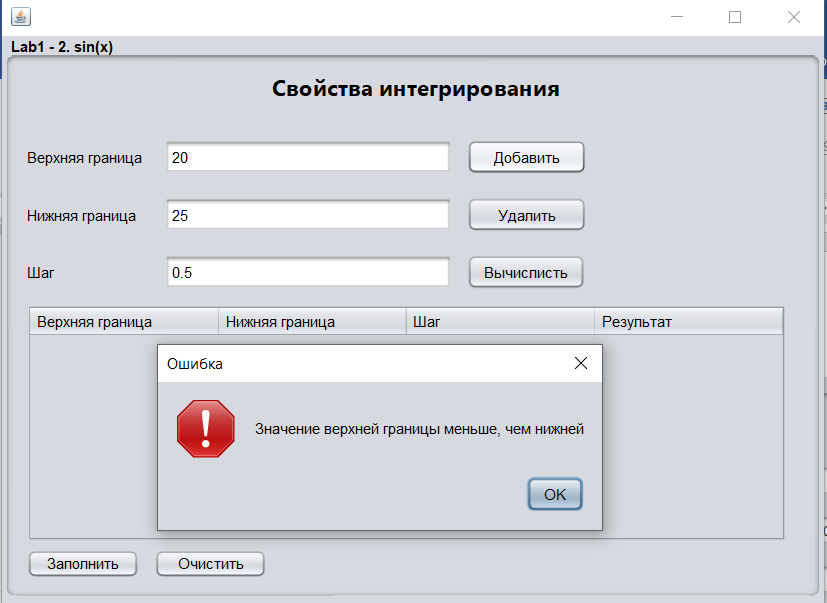
}

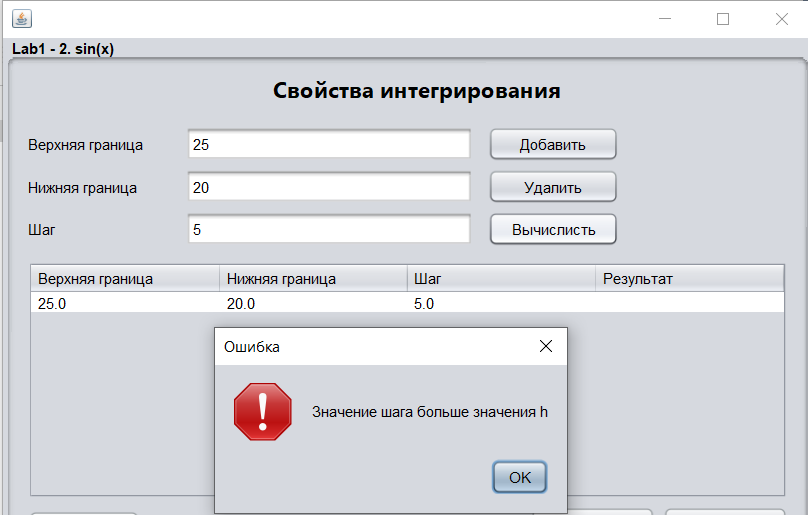
}

**Результаты выполнения работы:**









**Выводы:**

В процессе выполнения работы мы изучили механизм обработки исключительных ситуаций.