Министерство науки и высшего образования РФ

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №3

по дисциплине «Программирование на языке Java»

на тему «Обработка исключительных ситуаций»

Выполнили: студенты группы 22ВВП1

Беляев Д. И.

Демин М. С.

Приняли:

Юрова О. В.

Карамышева Н. С.

Пенза 2025

**Название**

Обработка исключительных ситуаций

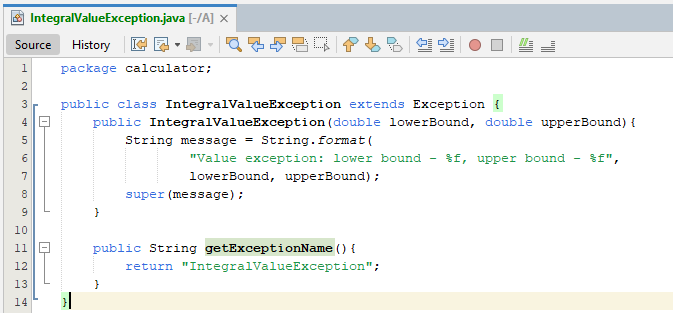
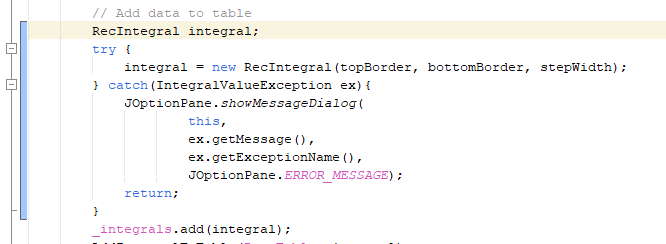
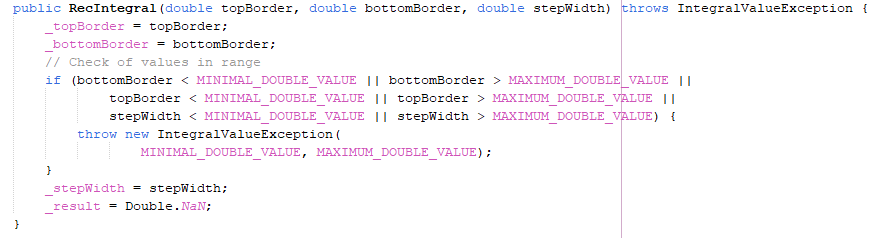
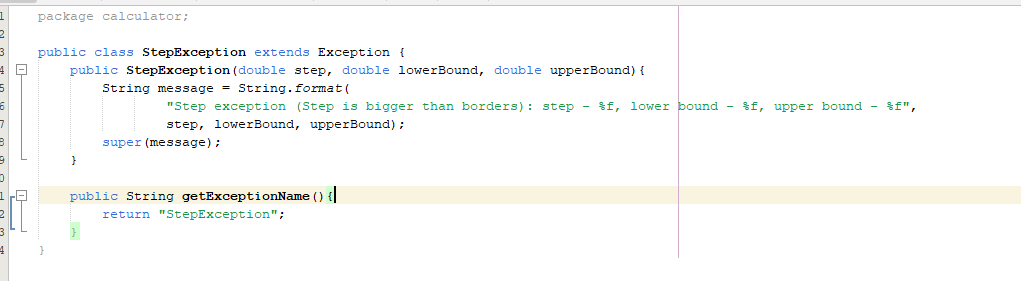
**Цель работы**

Изучить механизм обработки исключительных ситуаций

**Задание**

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 класс исключение  
   
2. Модифицировали основной класс, для обработки ошибок  
   
3. Модифицировали счетный класс  
   
4. Добавили StepException в качестве доп. задания  
   

**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;

import java.util.ArrayList;

import javax.swing.JOptionPane;

import javax.swing.JTable;

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";

private final ArrayList<RecIntegral> \_integrals;

/\*\*

\* Creates new form MainFrame

\*/

public MainFrame() {

initComponents();

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

\_integrals = new ArrayList<>();

}

/\*\*

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

\*/

@SuppressWarnings("unchecked")

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

private void initComponents() {

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();

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);

}

});

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(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())

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(DataTableScrollPanel)

.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

integral.calculateIntegral();

// 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 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.JButton FillTableButton;

private javax.swing.JLabel StepWidthLabel;

private javax.swing.JTextField StepWidthTextField;

private javax.swing.JLabel TopBorderLabel;

private javax.swing.JTextField TopBorderTextField;

// End of variables declaration

}

*RecIntegral.java*

package calculator;

public class RecIntegral {

final double MINIMAL\_DOUBLE\_VALUE = 0.000001;

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 {

\_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 = Double.NaN;

}

public double getTopBorder() { return \_topBorder; }

public double getBottomBorder() { return \_bottomBorder; }

public double getStepWidth() { return \_stepWidth; }

public double getResult() { return \_result; }

public void calculateIntegral() {

// Ensure that the step width is positive

if (\_stepWidth <= 0) {

throw new IllegalArgumentException("Step width must be positive");

}

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;

}

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;

}

}

*IntegralValueException.java*

package calculator;

public class IntegralValueException extends Exception {

public IntegralValueException(double lowerBound, double upperBound){

String message = String.format(

"Value exception: lower bound - %f, upper bound - %f",

lowerBound, upperBound);

super(message);

}

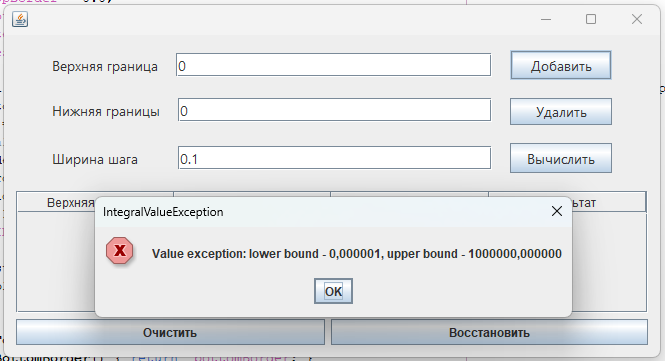
public String getExceptionName(){

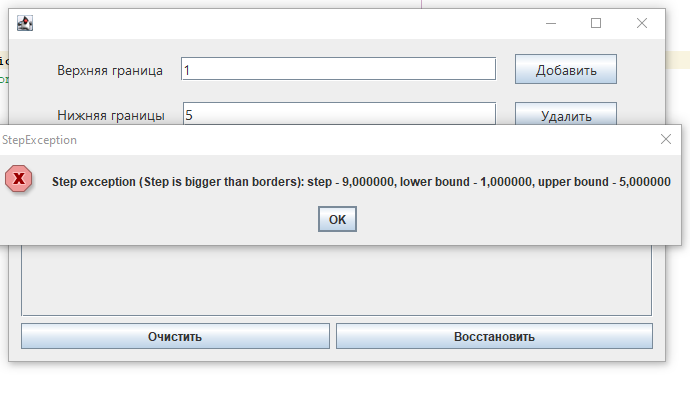
return "IntegralValueException";

}

}

**Результат выполнения программы**

****

****

**Вывод**

Изучили механизм обработки исключительных ситуаций