Пензенский Государственный университет

Кафедра «Вычислительная техника»

**Отчёт**

По лабораторной работе №3

По дисциплине: «Программирование на языке JAVA»

По теме: «Обработка исключительных ситуаций»

Вариант 4

Выполнили:

Студенты группы 20ВВП2

Кожевникова А.В.

Баулин А.Д.

Приняли:

Юрова О.В.

Карамышева Н.С.

Пенза 2023

**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 выполнения работы**

Расширили функционал программы из предыдущей лабораторной работы, создав собственный класс, который производит обработку исключений при вводе данных.

**Листинг**

**Файл Form\_Integral.java**

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.MouseEvent;

import java.awt.event.MouseListener;

import java.util.LinkedList;

import java.util.function.Function;

import java.io.Serializable;

public class Form\_Integral extends JFrame{

private JButton addbtn = new JButton("Добавить"),

delbtn = new JButton("Удалить"),

countbtn = new JButton("Вычислить"),

cleanbtn = new JButton("Очистить"),

fillbtn = new JButton(("Заполнить"));

private JTextField highText = new JTextField("",10),

lowText = new JTextField("",10),

StepText = new JTextField("",10);

private JLabel highlabel = new JLabel("Введите верхнюю границу: "),

lowlabel = new JLabel("Введите нижнюю границу: "),

steplabel = new JLabel("Введите шаг интегрирования: ");

private JTable table;

// LinkedList

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

// JFrame

static JFrame f;

//Создаём отоброжаемую на экране форму

public Form\_Integral() {

setTitle("Вычисление интеграла");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setBounds(100,100,550,450);

Container c = getContentPane();

JPanel panel = new JPanel(),

btnpanel = new JPanel();

JScrollPane tablePanel = new JScrollPane();

GroupLayout layout = new GroupLayout(panel);

panel.setLayout(layout);

layout.setAutoCreateGaps(true);

layout.setAutoCreateContainerGaps(true);

layout.setHorizontalGroup(

layout.createSequentialGroup()

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

.addComponent(highlabel)

.addComponent(lowlabel)

.addComponent(steplabel)

)

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

.addComponent(highText)

.addComponent(lowText)

.addComponent(StepText)

)

);

layout.setVerticalGroup(

layout.createSequentialGroup()

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

.addComponent(highlabel)

.addComponent(highText))

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

.addComponent(lowlabel)

.addComponent(lowText))

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

.addComponent(steplabel)

.addComponent(StepText))

);

btnpanel.setLayout(new GridLayout(2,4,4,4));

//Панель с кнопками

countbtn.addActionListener(new BtnEventListener());

addbtn.addActionListener(new AddDataBtn());

delbtn.addActionListener(new DeleteDataBtn());

cleanbtn.addActionListener(new CleanDataBtn());

fillbtn.addActionListener(new FillDataBtn());

btnpanel.add(addbtn);

btnpanel.add(delbtn);

btnpanel.add(countbtn);

btnpanel.add(cleanbtn);

btnpanel.add(fillbtn);

btnpanel.setBorder(BorderFactory.createEmptyBorder(10,10,10,10));

//Панель с табличкой

String[] columnNames = { "Верхняя граница", "Нижняя граница", "Шаг интегрирования", "Результат" };

int numRows = 0;

DefaultTableModel model = new DefaultTableModel(numRows,columnNames.length);

model.setColumnIdentifiers(columnNames);

table = new JTable(model)

{

public boolean isCellEditable(int row, int column)

{

return column !=3;

}

};

table.setSize(530,250);

table.setBackground(new Color(164,191,220));

table.addMouseListener(new TableMouseClicked());

panel.setBackground(new Color(164,191,220));

//btnpanel.setBackground(Color.blue);

//tablePanel.setBackground(Color.green);

c.setLayout(new BoxLayout(c, BoxLayout.Y\_AXIS));

c.add(panel);

c.add(btnpanel);

c.add(new JScrollPane(table));

}

//Эта функция выводит на текстовые поля данные из строки таблицы,на которую мы нажали

class TableMouseClicked implements MouseListener{

public void mouseClicked(MouseEvent e) {

DefaultTableModel tblModel = (DefaultTableModel)table.getModel();

String tblHigh = tblModel.getValueAt(table.getSelectedRow(),0).toString();

String tblLow = tblModel.getValueAt(table.getSelectedRow(),1).toString();

String tblStep = tblModel.getValueAt(table.getSelectedRow(),2).toString();

highText.setText(tblHigh);

lowText.setText(tblLow);

StepText.setText(tblStep);

}

@Override

public void mousePressed(MouseEvent e) {

}

@Override

public void mouseReleased(MouseEvent e) {

}

@Override

public void mouseEntered(MouseEvent e) {

}

@Override

public void mouseExited(MouseEvent e) {

}

}

//В этой функции совершается подсчёт интеграла

class BtnEventListener extends Component implements ActionListener{

//Здесь будет выполняться вычисление интеграла

public void actionPerformed (ActionEvent e){

double b = Double.valueOf(highText.getText()),

a = Double.valueOf(lowText.getText()),

h = Double.valueOf(StepText.getText()),

integral = 0;

Function function;

for (double i = a; i < b; i+=h)

{

integral += h \* (0.5 \* (Math.tan(a) + Math.tan((i + h))));

}

//Записываем данные в таблицу

DefaultTableModel tblModel = (DefaultTableModel) table.getModel();

if (table.getSelectedColumnCount() == 1)

{

String high = highText.getText(),

low = lowText.getText(),

Step = StepText.getText();

tblModel.setValueAt(high, table.getSelectedRow(), 0);

tblModel.setValueAt(low, table.getSelectedRow(), 1);

tblModel.setValueAt(Step, table.getSelectedRow(), 2);

tblModel.setValueAt(String.valueOf(integral), table.getSelectedRow(), 3);

}

else

{

if (table.getRowCount() == 0)

{

JOptionPane.showMessageDialog(null, "Таблица пустая!");

}

else

{

JOptionPane.showMessageDialog(null, "Выберите строку,которую надо вычислить!");

}

}

}

}

//Добавление информации в таблицу

class AddDataBtn extends Component implements ActionListener

{

public void actionPerformed (ActionEvent e)

{

String data[] = {highText.getText(), lowText.getText(), StepText.getText()};

try

{

if(Double.valueOf(StepText.getText()) > Double.valueOf(highText.getText()))

{

throw new IntegralException("Шаг больше верхнего предела!");

}

else if (Double.valueOf(StepText.getText()) < 0.0)

{

throw new IntegralException("Шаг меньше 0!");

}

else if (Double.valueOf(lowText.getText()) > Double.valueOf(highText.getText()))

{

throw new IntegralException("Нижний предел больше верхнего!");

}

else if (Double.valueOf(highText.getText())<0.000001 || Double.valueOf(highText.getText())>1000000)

{

throw new IntegralException("Верхний предел имеет неправильное значение!");

}

else if (Double.valueOf(lowText.getText())<0.000001 || Double.valueOf(lowText.getText())>1000000)

{

throw new IntegralException("Нижний предел имеет неправильное значение!");

}

else if (Double.valueOf(StepText.getText())<0.000001 || Double.valueOf(StepText.getText())>1000000)

{

throw new IntegralException("Шаг имеет неправильное значение!");

}

}

catch (Exception ex)

{

return;

}

RecIntegral recIntegral = new RecIntegral();

recIntegral.setData(data);

dataList.add(recIntegral);

DefaultTableModel tblModel = (DefaultTableModel) table.getModel();

tblModel.addRow(data);

//очищаем поле для новых данных

highText.setText("");

lowText.setText("");

StepText.setText("");

}

}

//Функция очищающая таблицу

class CleanDataBtn extends Component implements ActionListener

{

public void actionPerformed(ActionEvent e){

DefaultTableModel tblModel = (DefaultTableModel)table.getModel();

tblModel.setRowCount(0);

}

}

//Функция, заполняющая таблицу данными из LinkedList`а

class FillDataBtn extends Component implements ActionListener

{

public void actionPerformed(ActionEvent e){

DefaultTableModel tblModel = (DefaultTableModel) table.getModel();

tblModel.setRowCount(0);

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

{

tblModel.addRow(dataList.get(i).dataFromList);

}

}

}

//Удаление данных из таблицы

class DeleteDataBtn extends Component implements ActionListener{

public void actionPerformed(ActionEvent e){

DefaultTableModel tblModel = (DefaultTableModel)table.getModel();

if (table.getSelectedColumnCount() == 1){

tblModel.removeRow(table.getSelectedRow());

}else{

if (table.getRowCount() == 0){

JOptionPane.showMessageDialog(null, "Таблица пустая!");

}

else{

//если таблица не пустая, но ничего не выбрано

JOptionPane.showMessageDialog(null, "Вы ничего не выбрали!");

}

}

}

}

//Промежуточный класс для хранение одной строки данных

class RecIntegral implements Serializable

{

String dataFromList[] = new String[3];

public void setData(String new\_data[])

{

dataFromList=new\_data;

}

}

//Класс обработки исключений значений класса RecIntegral

class IntegralException extends Exception

{

public IntegralException(String message)

{

JOptionPane.showMessageDialog(null, message);

}

}

}

**Файл Main.java**

public class Main {

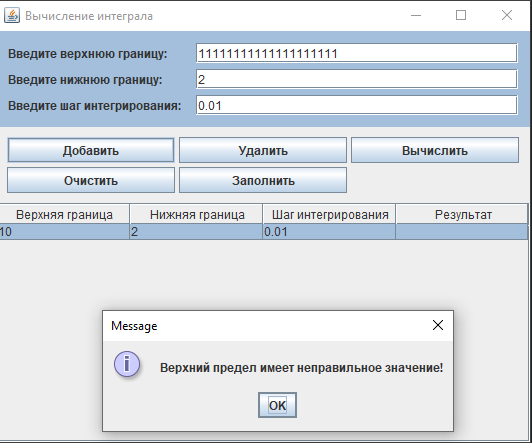
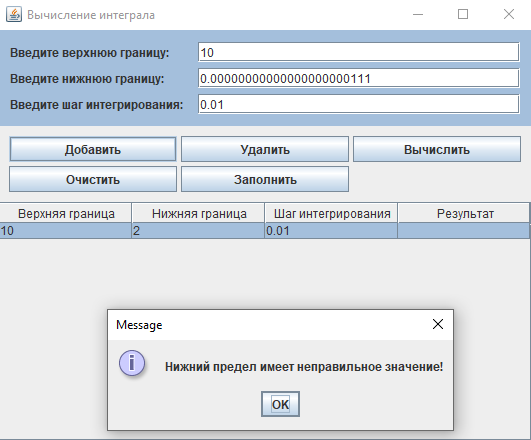
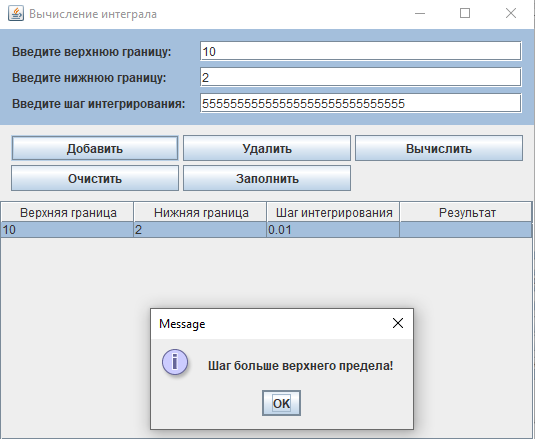
public static void main(String[] args) {

Form\_Integral app = new Form\_Integral();

app.setVisible(true);

}

}

**Результат работ**

**Вывод**: изучили механизм обработки исключительных ситуаций.