Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**по курсу «Программирование на языке JAVA»

по лабораторной работе №3

на тему: «Обработка исключительных ситуаций»  
Вариант 10.

Выполнил студент группы 20ВВП1:

Кодиров И. Н.

Приняла:

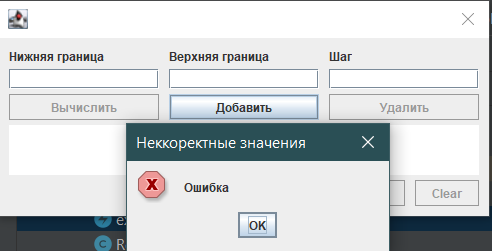
Юрова О. В.

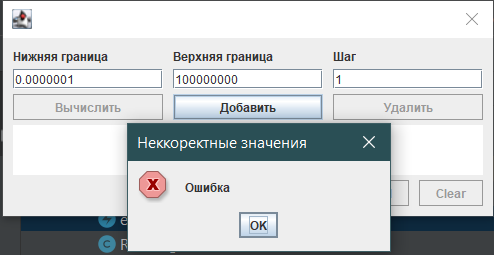
Пенза 2023

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

**Задание на лабораторную работу:** 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.

**Результаты выполнения программы:**

****

****

**Листинг:**

import javax.swing.\*;  
import javax.swing.table.DefaultTableModel;  
import java.awt.event.\*;  
import java.util.ArrayList;  
import java.util.Collection;  
  
public class gui extends JDialog {  
 private JPanel contentPane;  
 private JTextField textFieldlow;  
 private JTextField textFieldup;  
 private JTextField textFieldstep;  
 private JButton buttonadd;  
 private JButton buttondelete;  
 private JButton buttoncalc;  
 private JTable table;  
 private JButton fill;  
 private JButton clear;  
 private Recintegral recintegral = new Recintegral();  
 private ArrayList<Recintegral> digits = new ArrayList();  
 DefaultTableModel model = (DefaultTableModel) table.getModel();  
 boolean check;  
 public gui()  
 {  
 setContentPane(contentPane);  
 setModal(true);  
 clear.setEnabled(false);  
 fill.setEnabled(false);  
 buttoncalc.setEnabled(false);  
 buttondelete.setEnabled(false);  
  
 createtable();  
  
 buttonadd.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 add();  
 }  
 });  
  
 buttondelete.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 delete();  
 }  
 });  
  
 buttoncalc.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 try {  
 calc();  
 } catch (excep ex) {  
 throw new RuntimeException(ex);  
 }  
 }  
 });  
 fill.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e)  
 {  
 tabfill();  
 }  
 });  
 clear.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e)  
 {  
 tabclear();  
 }  
 });  
 // call onCancel() when cross is clicked  
 setDefaultCloseOperation(DO\_NOTHING\_ON\_CLOSE);  
 addWindowListener(new WindowAdapter() {  
 public void windowClosing(WindowEvent e) {  
 onCancel();  
 }  
 });  
  
 // call onCancel() on ESCAPE  
 contentPane.registerKeyboardAction(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 onCancel();  
 }  
 }, KeyStroke.getKeyStroke(KeyEvent.VK\_ESCAPE, 0), JComponent.WHEN\_ANCESTOR\_OF\_FOCUSED\_COMPONENT);  
 }  
  
 public void createtable()  
 {  
 model.addColumn("Верхняя граница интегрирования");  
 model.addColumn("Нижняя граница интегрирования");  
 model.addColumn("Шаг интегрирования");  
 model.addColumn("Результат");  
 }  
 private void tabfill(){  
 model.addRow(digits.get(0).addMod());  
 str++;  
 }  
 private void tabclear(){  
 model.setRowCount(0);  
 str=-1;  
 clear.setEnabled(false);  
 fill.setEnabled(false);  
 buttoncalc.setEnabled(false);  
 buttondelete.setEnabled(false);  
 }  
  
 private void onCancel() {  
 // add your code here if necessary  
 dispose();  
 }  
 int str=-1;  
 private void add(){  
 try  
 {  
 double a = Double.parseDouble(textFieldlow.getText());  
 double b = Double.parseDouble(textFieldup.getText());  
 double h = Double.parseDouble(textFieldstep.getText());  
 if(textFieldup.getText().equals("")) {throw new excep();}  
 if(textFieldlow.getText().equals("")){throw new excep();}  
 if(textFieldstep.getText().equals("")){throw new excep();}  
 if(a <= 0.000001 || a>=1000000)  
 throw new excep();  
 if(b <= 0.000001 || b>=1000000)  
 throw new excep();  
 if(h <= 0.000001 || h>=1000000)  
 throw new excep();  
 check = true;  
 model.addRow( new Object[]{textFieldlow.getText(), textFieldup.getText(), textFieldstep.getText()});  
 recintegral.setAllField(Double.valueOf(textFieldlow.getText()),Double.valueOf(textFieldup.getText()),Double.valueOf(textFieldstep.getText()));  
 digits.add(recintegral);  
 str++;  
 }  
 catch (excep ex)  
 {  
 throw new RuntimeException(ex);  
 }  
 if (check)  
 {  
 clear.setEnabled(true);  
 fill.setEnabled(true);  
 buttoncalc.setEnabled(true);  
 buttondelete.setEnabled(true);  
 }  
 }  
  
 private void delete(){  
 if(str == 0 && check)  
 {  
 clear.setEnabled(false);  
 fill.setEnabled(false);  
 buttoncalc.setEnabled(false);  
 buttondelete.setEnabled(false);  
 }  
 DefaultTableModel model = (DefaultTableModel)table.getModel();  
 model.removeRow(table.getSelectedRow());  
 str--;  
 }  
  
 private void calc() throws excep{  
 table.setValueAt(counted(), str, 3);  
 textFieldlow.setText("");  
 textFieldup.setText("");  
 textFieldstep.setText("");  
 }  
  
 public double counted() throws excep  
 {  
 double a = Double.parseDouble(textFieldlow.getText());  
 double b = Double.parseDouble(textFieldup.getText());  
 double h = Double.parseDouble(textFieldstep.getText());  
 return integral(a, b, h);  
 }  
 public double integral(double a, double b, double h)  
 {  
 double area = 0;  
 if (h == 0) return area;  
  
 for(int i = 0; i < (b-a)/h; i++){  
 area +=f(a + i\*h);  
 }  
 area += (f(a)+f(b))/2;  
 area \*= h;  
 return area;  
 }  
  
 public double f(double x){  
 return ((Math.pow(Math.E, x))/x);  
 }  
  
 public static void main(String[] args) {  
 gui dialog = new gui();  
 dialog.pack();  
 dialog.setVisible(true);  
 System.exit(0);  
 }  
  
}

**RecIntegral**

public class Recintegral {  
 private double Min;  
 private double Max;  
 private double Step;  
 private double Result;  
  
 public Recintegral(double min, double max, double step, double result) {  
 Min = min;  
 Max = max;  
 Step = step;  
 Result = result;  
 }  
 public Recintegral() {  
 Min = 0;  
 Max = 0;  
 Step = 0;  
 Result = 0;  
 }  
 public Recintegral(double[] temp) {  
  
 Min = temp[0];  
 Max = temp[1];  
 Step = temp[2];  
 Result = Trap(temp[0],temp[1],temp[2]);  
 }  
 public void setAllField(double a, double b,double step)  
 {  
 Min = a;  
 Max = b;  
 Step = step;  
 if(Min>Max)  
 {  
 setResultNull();  
 }else {  
 setResult();  
 }  
 }  
 public void setResult() {  
  
 Result = Trap(Min,Max,Step);  
 }  
 public void setResultNull() {  
  
 Result = 0;  
 }  
  
 public Object[] addMod() {  
 Object[] temp=new Object[4];  
 temp[0]=Min;  
 temp[1]=Max;  
 temp[2]=Step;  
 temp[3]=Result;  
 return temp;  
 }  
 public double Trap(double a,double b, double h){  
 double result = 0;  
 double num = 0;  
 for(double i=a;i<=b-(h\*2);i+=h)  
 {num=i+h;  
 if(i>b)  
 {  
 num=b;  
 }  
 result+=(InFunction(i)+InFunction(num))\*(b-i)/2;  
 }  
 return result;  
 }  
 public static double InFunction(double x) //Подынтегральная функция  
 {  
 return ((Math.pow(Math.E, x))/x);  
 }  
 @Override  
 public String toString() {  
 return "DataNumber{" +  
 "Min=" + Min +  
 ", Max=" + Max +  
 ", Step=" + Step +  
 ", Result=" + Result +  
 '}';  
 }  
}

**Excep**

public class excep extends Exception{  
public excep()  
{  
 showMessageDialog(null,"Ошибка","Неккоректные значения", JOptionPane.ERROR\_MESSAGE);  
}  
}

**Вывод:** научился разрабатывать приложения, обладающие графическим интерфейсом пользователя, а также изучил механизм обработки исключительных ситуаций.