Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

Пенза 2021

**ОТЧЁТ**

по лабораторной работе №2

по курсу «Программирование на языке JAVA»

на тему «Работа с коллекциями объектов»

Выполнили:

студенты группы 21ВВП1

Брянцев А. Г.

Сущев М. В.

Тюрин В. И.

Приняли:

Юрова О.В.

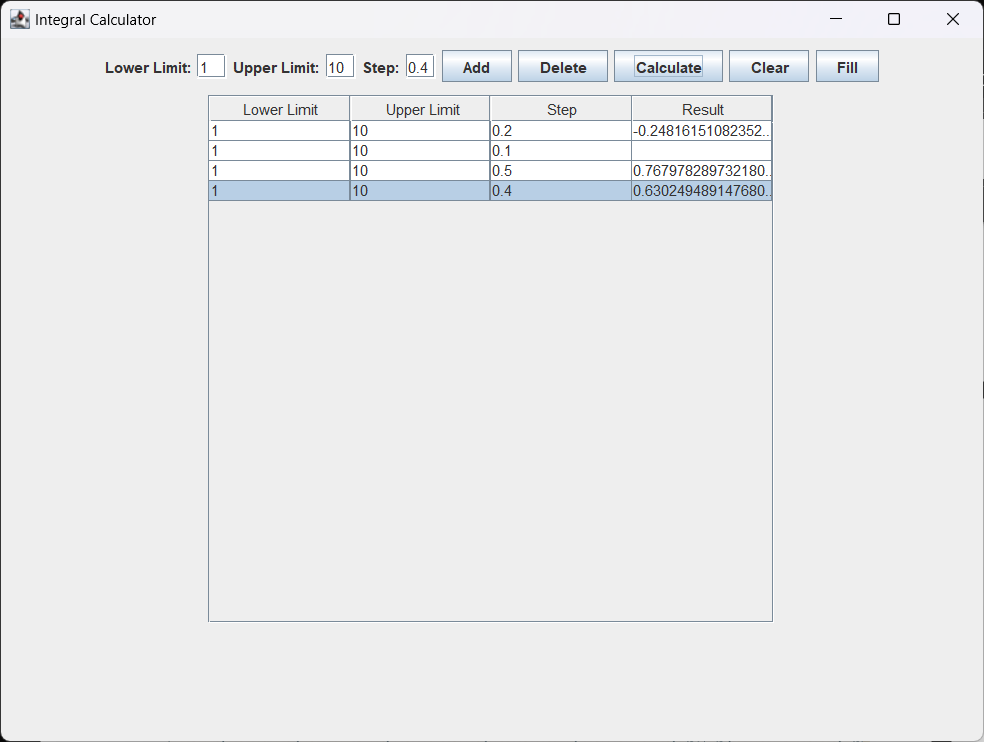
Карамышева Н.С.

Пенза 2024

**Цель работы:** изучить библиотеку стандартных коллекций Java Collections Framework, позволяющую хранить различные структуры данных.

Вариант – 9 ArrayList

**Результат работы программы:**

****

**Изображение выглядит как текст, снимок экрана, дисплей, компьютер

Автоматически созданное описание**

**Листинг**

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.ArrayList;

public class IntegralCalculatorGUI extends JFrame {

private DefaultTableModel tableModel;

private JTable table;

private JTextField lowerLimitField;

private JTextField upperLimitField;

private JTextField stepField;

private ArrayList<RecIntegral> integralList;

public IntegralCalculatorGUI() {

setTitle("Integral Calculator");

setSize(800, 600);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

tableModel = new DefaultTableModel();

tableModel.addColumn("Lower Limit");

tableModel.addColumn("Upper Limit");

tableModel.addColumn("Step");

tableModel.addColumn("Result");

table = new JTable(tableModel);

lowerLimitField = new JTextField(2);

upperLimitField = new JTextField(2);

stepField = new JTextField(2);

integralList = new ArrayList<>();

JButton addButton = new JButton("Add");

addButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

tableModel.addRow(new Object[]{lowerLimitField.getText(), upperLimitField.getText(), stepField.getText(), ""});

// Создаем объект RecIntegral с заданными значениями и добавляем его в коллекцию

RecIntegral recIntegral = new RecIntegral(0, 0, 0);

recIntegral.setLowerLimit(lowerLimitField.getText());

recIntegral.setUpperLimit(upperLimitField.getText());

recIntegral.setStep(stepField.getText());

integralList.add(recIntegral);

}

});

JButton deleteButton = new JButton("Delete");

deleteButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

int selectedRow = table.getSelectedRow();

if (selectedRow != -1) {

tableModel.removeRow(selectedRow);

integralList.remove(selectedRow);

}

}

});

JButton calculateButton = new JButton("Calculate");

calculateButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

double lowerLimit, upperLimit, step;

try {

int selectedRow = table.getSelectedRow();

lowerLimit = Double.parseDouble(tableModel.getValueAt(selectedRow, 0).toString());

upperLimit = Double.parseDouble(tableModel.getValueAt(selectedRow, 1).toString());

step = Double.parseDouble(tableModel.getValueAt(selectedRow, 2).toString());

double integralResult = calculateIntegral(lowerLimit, upperLimit, step);

if (selectedRow != -1) {

tableModel.setValueAt(integralResult, selectedRow, 3);

}

} catch (NumberFormatException ex) {

JOptionPane.showMessageDialog(null, "Please enter valid numerical values.");

}

}

});

JButton clearButton = new JButton("Clear");

clearButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

tableModel.setRowCount(0); // Очистка таблицы

// integralList.clear(); // Очистка коллекции

}

});

JButton fillButton = new JButton("Fill");

fillButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

// Очищаем таблицу перед заполнением

tableModel.setRowCount(0);

// Заполняем таблицу данными из объектов RecIntegral в коллекции integralList

for (RecIntegral integral : integralList) {

tableModel.addRow(new Object[]{integral.getLowerLimit(), integral.getUpperLimit(), integral.getStep()});

}

}

});

JPanel inputPanel = new JPanel();

inputPanel.add(new JLabel("Lower Limit:"));

inputPanel.add(lowerLimitField);

inputPanel.add(new JLabel("Upper Limit:"));

inputPanel.add(upperLimitField);

inputPanel.add(new JLabel("Step:"));

inputPanel.add(stepField);

inputPanel.add(addButton);

inputPanel.add(deleteButton);

inputPanel.add(calculateButton);

inputPanel.add(clearButton);

inputPanel.add(fillButton);

JPanel mainPanel = new JPanel();

mainPanel.add(inputPanel);

mainPanel.add(new JScrollPane(table));

getContentPane().add(mainPanel);

}

public static double calculateIntegral(double lowerLimit, double upperLimit, double step) {

double x1, x2, sum = 0;

int amountSteps = (int) ((upperLimit - lowerLimit) / step); //округляется в меньшую сторону

x1 = lowerLimit;

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

x2 = x1 + step;

sum += 0.5 \* step \* (Math.cos(x1 \* x1) + Math.cos(x2 \* x2));

x1 = x2;

}

if ((upperLimit - lowerLimit) % step != 0)

sum += 0.5 \* (upperLimit - x1) \* (Math.cos(x1 \* x1) + Math.cos(upperLimit \* upperLimit));

return sum;

}

}

class RecIntegral {

private double lowerLimit;

private double upperLimit;

private double step;

public RecIntegral(double lowerLimit, double upperLimit, double step) {

this.lowerLimit = lowerLimit;

this.upperLimit = upperLimit;

this.step = step;

}

public double getLowerLimit() {

return lowerLimit;

}

public double getUpperLimit() {

return upperLimit;

}

public double getStep() {

return step;

}

public void setLowerLimit(String lowerLimit) {

this.lowerLimit = Double.parseDouble(lowerLimit);

}

public void setUpperLimit(String upperLimit) {

this.upperLimit = Double.parseDouble(upperLimit);

}

public void setStep(String step) {

this.step = Double.parseDouble(step);

}

}

**Вывод:** в ходе данной лабораторной работы изучили библиотеку стандартных коллекций Java Collections Framework, позволяющую хранить различные структуры данных.