Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №2

по курсу «Программирование на Java»

Выполнили:

студенты группы 20ВВП1

Кирюткин И. А.

Кодиров И.Н.

Приняли:

Юрова О.В.

**Цель работы:** научиться разрабатывать приложения, обладающие графическим интерфейсом пользователя, с использованием библиотеки Swing. Изучить библиотеку стандартных коллекций Java Collections Framework, позволяющую хранить различные структуры данных.

**Задание:** Модифицировать приложение из предыдущей лабораторной работы, реализовав хранение данных таблицы с использованием библиотеки коллекций. Для этого реализовать класс RecIntegral, способный хранить одну запись таблицы. Для нечетных вариантов в качестве класса-коллекции выбрать ArrayList, для четных - LinkedList. Кроме того, добавить пару кнопок: очистить / заполнить, которые будут очищать таблицу и заполнять ее данными из коллекции соответственно.

**Результат работы программы:**

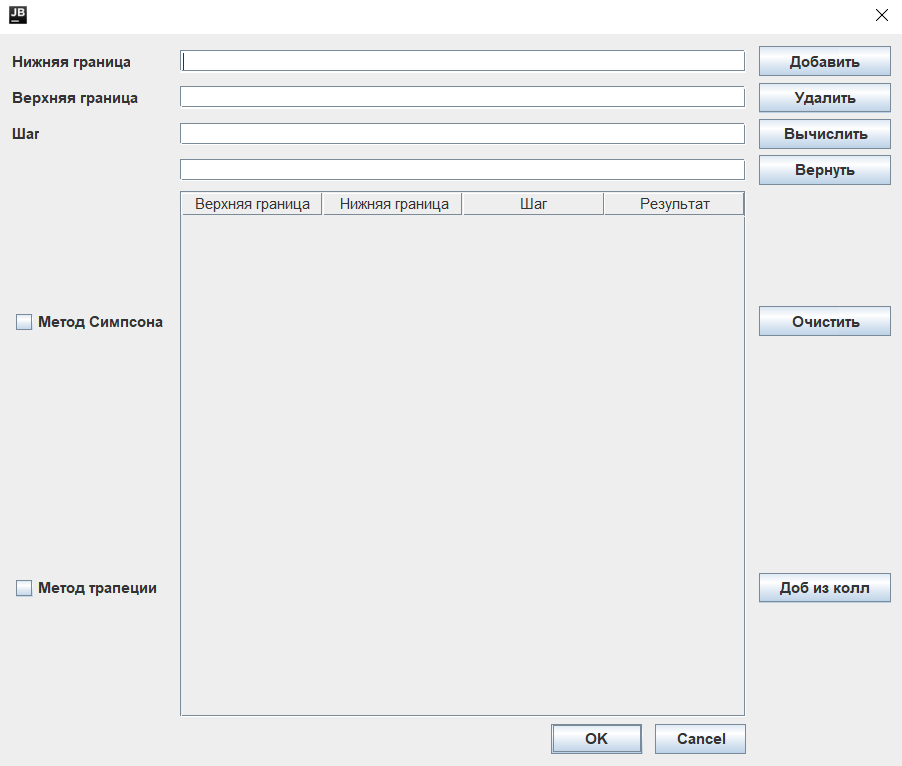


Рисунок 1 Программа

Листинг:

import javax.swing.\*;  
import javax.swing.table.DefaultTableModel;  
import java.awt.event.\*;  
import java.util.ArrayList;  
  
public class Form extends JDialog {  
 private JPanel contentPane;  
 private JButton buttonOK;  
 private JButton buttonCancel;  
 private JTable table1;  
 private JTextField textField1;  
 private JTextField textField2;  
 private JTextField textField3;  
 private JButton ButtonAdd;  
 private JButton ButtonDelete;  
 private JButton ButtonCalc;  
 private JTextField textField4;  
 private double[] tableCursor=new double[4];  
 private DataNumber dataNumber=new DataNumber();  
 private ArrayList<DataNumber> dataNumbers=new ArrayList();  
  
 DefaultTableModel modelData = (DefaultTableModel) table1.getModel();  
 private JButton buttonUp;  
 private JButton btnCleanTable;  
 private JButton buttonFromCollection;  
 private JCheckBox CheckBoxTrap;  
 private JCheckBox CheckBoxSimpson;  
 int num =0;  
 int realRow;  
 int realColumn;  
 boolean flagSave=false;  
 boolean flagPoint=false;  
  
 public Form() {  
 flagSave=false;  
 //contentPane.add(table1);  
 setContentPane(contentPane);  
 createTable();  
 setModal(true);  
 getRootPane().setDefaultButton(buttonOK);  
 buttonOK.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 onOK();  
 }  
 });  
  
 buttonCancel.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
  
 }  
 });  
 //добавление в таблицу  
 ButtonAdd.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 flagPoint=false;  
 AddTable();  
  
 }  
 });  
 //удаление  
 ButtonDelete.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 if(flagPoint){  
 delete();  
 }  
 flagPoint=false;  
  
 }  
 });  
 //вычисление  
 ButtonCalc.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 if(flagPoint) {  
 Calc();  
 }else{  
 CalcToField();  
 }  
 flagPoint=false;  
 }  
 });  
 buttonFromCollection.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 for(int i=0;i<num;i++) {  
 modelData.addRow(dataNumbers.get(i).addMod());  
 }  
 }});  
 //вернуть  
 buttonUp.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 if(flagPoint) {  
 textField1.setText("" + tableCursor[1]);  
 textField2.setText("" + tableCursor[0]);  
 textField3.setText("" + tableCursor[2]);  
 //textField4.setText("" + tableCursor[3]);  
 }  
 flagPoint=false;  
 }  
 });  
 btnCleanTable.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 modelData.setRowCount(0);  
 }  
 });  
 table1.addMouseListener(new MouseAdapter() {  
 @Override  
 public void mouseClicked(MouseEvent e) {  
 if (e.getClickCount() == 1) {//одинарный шелчок  
 int row = table1.rowAtPoint(e.getPoint());  
 int column = table1.columnAtPoint(e.getPoint());  
 //путь попроще без selectionMode  
 if (row > -1) {  
 flagPoint=true;  
 realRow = table1.convertRowIndexToModel(row);  
 realColumn = table1.convertColumnIndexToModel(column);  
 System.out.println("[" + realRow + "],[" + realColumn + "]");  
 tableCursor[0]=(double)table1.getValueAt(realRow, 0);  
 tableCursor[1]=(double)table1.getValueAt(realRow, 1);  
 tableCursor[2]=(double)table1.getValueAt(realRow, 2);  
 //tableCursor[3]=(double)table1.getValueAt(realRow, 3);  
 //System.out.println(tableCursor[0]);  
 //System.out.println(tableCursor[1]);  
 //System.out.println(tableCursor[2]);  
 // System.out.println(tableCursor[3]);  
 //номер строки из модели данных  
 //здесь должна быть выборка объекта из модели по номеру строки и его отображение  
 }  
 }  
 }  
 });  
 // 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);  
  
 }  
  
 private void delete()  
 {  
 modelData.removeRow(realRow);  
 dataNumbers.remove(realRow);  
 num--;  
 }  
 private void Calc() {  
 flagSave=true;  
 tableCursor[0]=(double)modelData.getValueAt(realRow,0);  
 tableCursor[1]=(double)modelData.getValueAt(realRow,1);  
 tableCursor[2]=(double)modelData.getValueAt(realRow,2);  
 tableCursor[3]=Trap(tableCursor[1],tableCursor[0],tableCursor[2]);  
 modelData.setValueAt(tableCursor[3],realRow,3);//4  
  
 }  
 private void CalcToField() {  
 textField4.setText(""+Trap(Double.valueOf(textField1.getText()),Double.valueOf(textField2.getText()),Double.valueOf(textField3.getText())));  
 }  
 private void AddTable() {  
 dataNumber.setAllField(Double.valueOf(textField1.getText()),Double.valueOf(textField2.getText()),Double.valueOf(textField3.getText()));  
 dataNumbers.add(dataNumber);  
 modelData.addRow(dataNumbers.get(num).addMod());  
 num++;  
 textField1.setText("");  
 textField2.setText("");  
 textField3.setText("");  
 textField4.setText("");  
 }  
 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;  
 }  
 //double result=0;  
 // int n = (int)((a-b)/h);  
 //result += (InFunction(a)+InFunction(b))/2;  
 //for(int i = 1; i < n; i++) {  
 // result += InFunction(b + h \* i);  
 //}  
 //}  
 return result;  
 }  
 public static double InFunction(double x) //Подынтегральная функция  
 {  
 return Math.sin(Math.pow(x,2));  
 }  
 public void createTable(){  
 modelData.addColumn("Верхняя граница");  
 modelData.addColumn("Нижняя граница");  
 modelData.addColumn("Шаг");  
 modelData.addColumn("Результат");  
  
 }  
  
  
 private void onOK() {  
 // add your code here  
 dispose();  
 }  
  
 private void onCancel() {  
 // add your code here if necessary  
 dispose();  
 }  
  
 public static void main(String[] args) {  
 Form dialog = new Form();  
 dialog.pack();  
 dialog.setVisible(true);  
 System.exit(0);  
 }  
}

//  
// Source code recreated from a .class file by IntelliJ IDEA  
// (powered by FernFlower decompiler)  
//  
  
public class DataNumber {  
 private double Min;  
 private double Max;  
 private double Step;  
 private double Result;  
  
 public DataNumber(double min, double max, double step, double result) {  
 this.Min = min;  
 this.Max = max;  
 this.Step = step;  
 this.Result = result;  
 }  
  
 public DataNumber() {  
 this.Min = 0.0;  
 this.Max = 0.0;  
 this.Step = 0.0;  
 this.Result = 0.0;  
 }  
  
 public DataNumber(double[] temp) {  
 this.Min = temp[0];  
 this.Max = temp[1];  
 this.Step = temp[2];  
 this.Result = this.Trap(temp[0], temp[1], temp[2]);  
 }  
  
 public double getMin() {  
 return this.Min;  
 }  
  
 public double getMax() {  
 return this.Max;  
 }  
  
 public double getStep() {  
 return this.Step;  
 }  
  
 public double getResult() {  
 return this.Result;  
 }  
  
 public void setMin(double min) {  
 this.Min = min;  
 }  
  
 public void setMax(double max) {  
 this.Max = max;  
 }  
  
 public void setStep(double step) {  
 this.Step = step;  
 }  
  
 public void setAllField(double a, double b, double step) {  
 this.Min = a;  
 this.Max = b;  
 this.Step = step;  
 if (this.Min > this.Max) {  
 this.setResultNull();  
 } else {  
 this.setResult();  
 }  
  
 }  
  
 public void setResult() {  
 this.Result = this.Trap(this.Min, this.Max, this.Step);  
 }  
  
 public void setResultNull() {  
 this.Result = 0.0;  
 }  
  
 public Object[] addMod() {  
 Object[] temp = new Object[]{this.Min, this.Max, this.Step, this.Result};  
 return temp;  
 }  
  
 public double Trap(double a, double b, double h) {  
 double result = 0.0;  
 double num = 0.0;  
  
 for(double i = a; i <= b - h \* 2.0; i += h) {  
 num = i + h;  
 if (i > b) {  
 num = b;  
 }  
  
 result += (InFunction(i) + InFunction(num)) \* (b - i) / 2.0;  
 }  
  
 return result;  
 }  
  
 public static double InFunction(double x) {  
 return Math.sin(Math.pow(x, 2.0));  
 }  
  
 public String toString() {  
 return "DataNumber{Min=" + this.Min + ", Max=" + this.Max + ", Step=" + this.Step + ", Result=" + this.Result + "}";  
 }  
}

**Вывод:** изучили основные формы библиотеки свинг, реализовали простое приложение для вычисления интеграла.