Министерство образования Российской Федерации

Пензенский государственный университет Кафедра «Вычислительная техника»

ОТЧЁТ

По лабораторной работе №2

По курсу «Программирование на языке JAVA»

на тему: «Работа с коллекциями объектов»

Выполнили:

студенты группы 20ВВП1:

Сурков Максим

Пантелеев Иван

Приняли:

Юрова О.В.

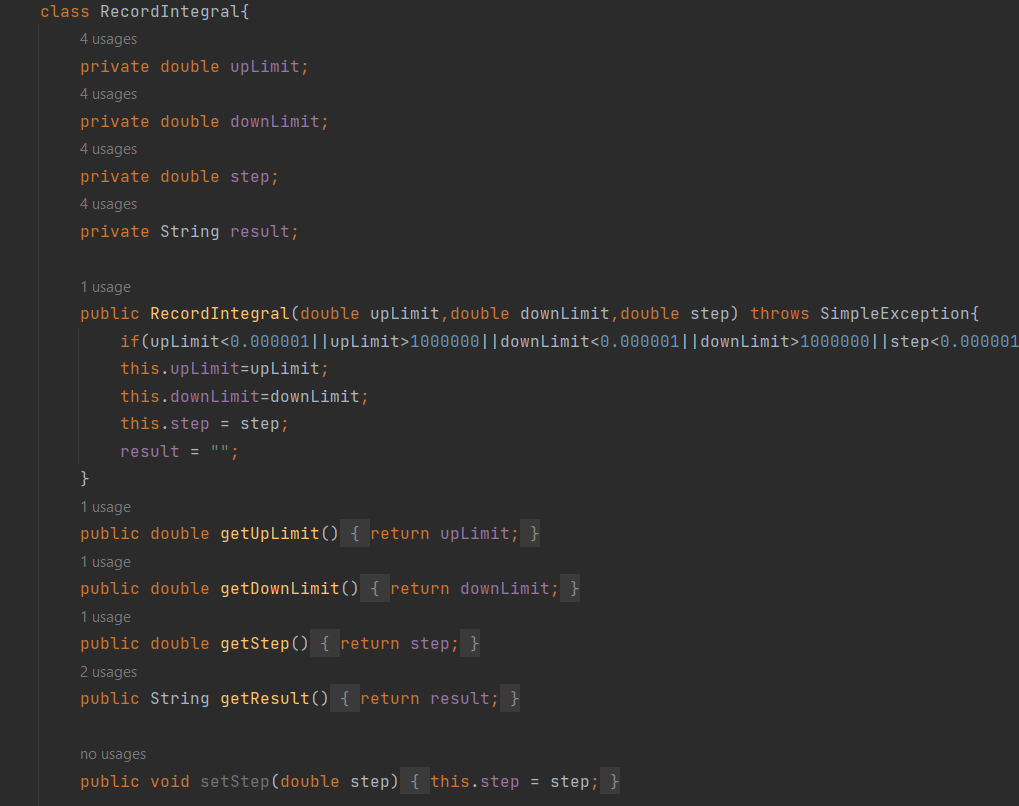
Карамышева Н.С.

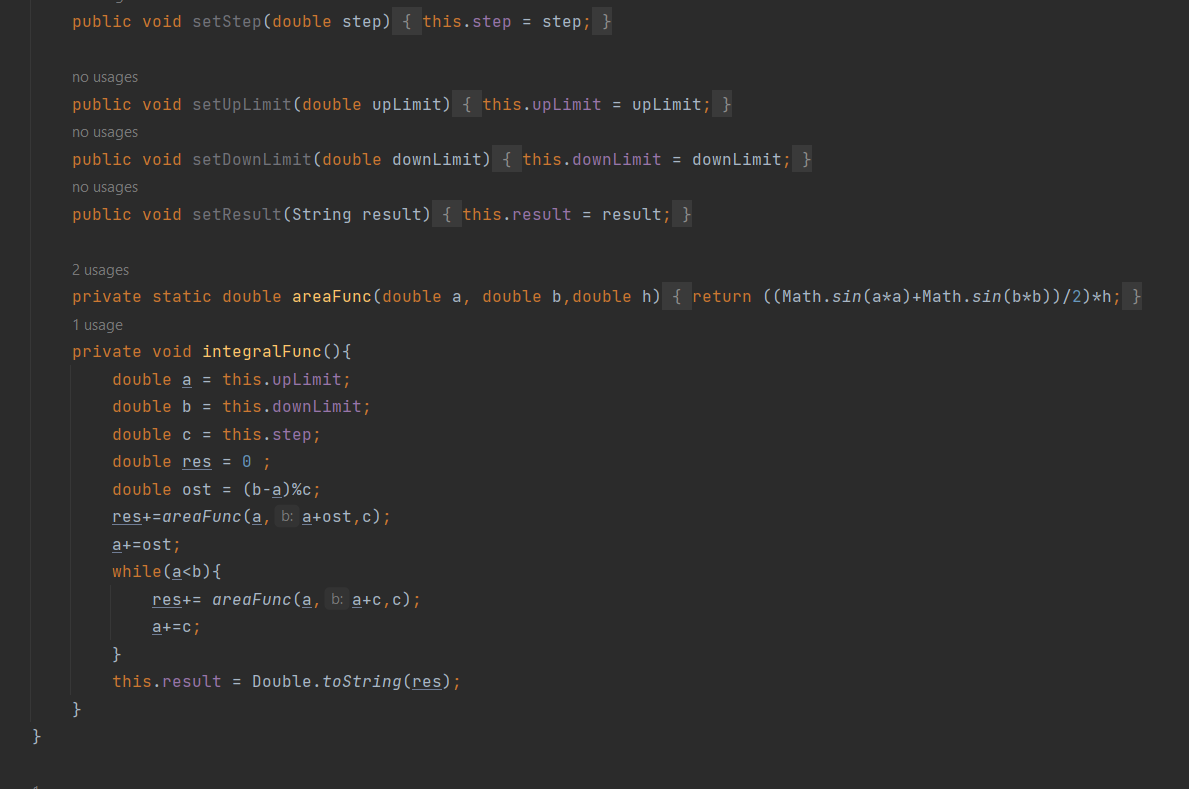
Пенза 2023

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

**Ход работы:**

Разработали класс RecIntergral содержащий одну запись таблицы





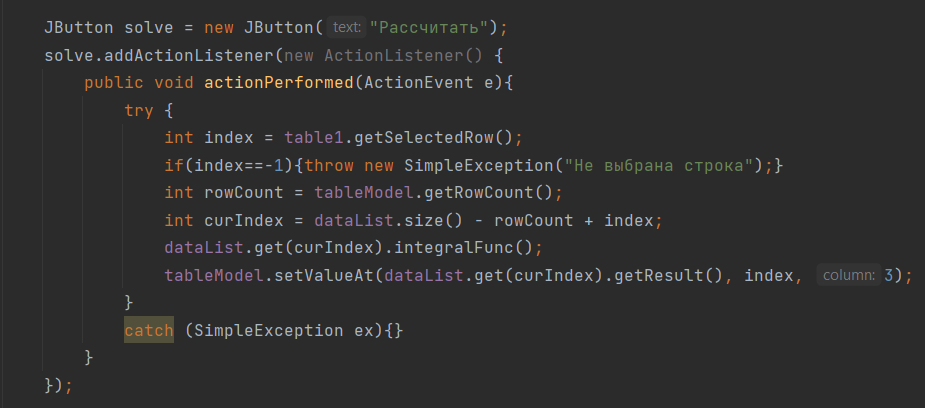
Создали LinkedList dataList



Изменили существующие методы кнопок и добавили 2 новых кнопки – «Очистить» очищающую все данные из талицы и «Заполнить» заполняющую таблицу данными из dataList





****

****

**Листинг:**

import javax.swing.\*;  
import javax.swing.table.DefaultTableModel;  
import javax.swing.table.AbstractTableModel;  
import javax.swing.text.BadLocationException;  
import javax.swing.text.Document;  
  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.io.IOException;  
import java.sql.SQLOutput;  
import java.util.LinkedList;  
import java.util.SimpleTimeZone;  
  
public class LabApp extends JFrame  
{  
 private JTextField upLimitField;  
 private JTextField stepField;  
 private JTextField downLimitField;  
 private DefaultTableModel tableModel;  
 private AbstractTableModel abstractTableModel;  
 private JTable table1;  
 private JLabel label1;  
 private JLabel label2;  
 private JLabel label3;  
 private Object[] columnsHeader;  
 LinkedList<RecordIntegral>dataList;  
  
 class SimpleException extends Exception{  
 public SimpleException(){};  
 public SimpleException(String msg){  
 JOptionPane.*showMessageDialog*(null,msg,"Exception",JOptionPane.*PLAIN\_MESSAGE*);  
 }  
 }  
  
 class RecordIntegral{  
 private double upLimit;  
 private double downLimit;  
 private double step;  
 private String result;  
  
 public RecordIntegral(double upLimit,double downLimit,double step) throws SimpleException{  
 if(upLimit<0.000001||upLimit>1000000||downLimit<0.000001||downLimit>1000000||step<0.000001||step>1000000) throw new SimpleException("Введены некорректные данные");  
 this.upLimit=upLimit;  
 this.downLimit=downLimit;  
 this.step = step;  
 result = "";  
 }  
 public double getUpLimit() {  
 return upLimit;  
 }  
 public double getDownLimit() {  
 return downLimit;  
 }  
 public double getStep() {  
 return step;  
 }  
 public String getResult() {  
 return result;  
 }  
  
 public void setStep(double step) {  
 this.step = step;  
 }  
  
 public void setUpLimit(double upLimit) {  
 this.upLimit = upLimit;  
 }  
 public void setDownLimit(double downLimit) {  
 this.downLimit = downLimit;  
 }  
 public void setResult(String result) {  
 this.result = result;  
 }  
  
 private static double areaFunc(double a, double b,double h){  
 return ((Math.*sin*(a\*a)+Math.*sin*(b\*b))/2)\*h;  
 }  
 private void integralFunc(){  
 double a = this.upLimit;  
 double b = this.downLimit;  
 double c = this.step;  
 double res = 0 ;  
 double ost = (b-a)%c;  
 res+=*areaFunc*(a,a+ost,c);  
 a+=ost;  
 while(a<b){  
 res+= *areaFunc*(a,a+c,c);  
 a+=c;  
 }  
 this.result = Double.*toString*(res);  
 }  
 }  
  
 public LabApp()  
 {  
 super("Swing App");  
 setDefaultCloseOperation(*EXIT\_ON\_CLOSE*);  
  
 tableModel = new DefaultTableModel(){  
 @Override  
 public boolean isCellEditable(int i, int i1) {  
 return i1 != 3;  
 }  
 };  
 columnsHeader = new String[] {"Ниж. предел", "Верх. предел", "Шаг","Результат"};  
 tableModel.setColumnIdentifiers(columnsHeader);  
 upLimitField = new JTextField(10);  
 downLimitField = new JTextField(10);  
 stepField = new JTextField(10);  
 label1 = new JLabel("Нижний предел");  
 label2 = new JLabel("Верхний предел");  
 label3 = new JLabel("Шаг");  
 table1 = new JTable(tableModel);  
 dataList = new LinkedList<RecordIntegral>();  
  
 JButton add = new JButton("Добавить");  
 add.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 try {  
 double a = Double.*parseDouble*(upLimitField.getText());  
 double b = Double.*parseDouble*(downLimitField.getText());  
 double c = Double.*parseDouble*(stepField.getText());  
 dataList.add(new RecordIntegral(a,b,c));  
 if((b-a)<c)throw new SimpleException("Шаг превышает длинну отрезка интегрирования");  
 tableModel.addRow(new Object [] {a,b,c});  
 }  
 catch (SimpleException ex){}  
 catch (NumberFormatException ex){  
 JOptionPane.*showMessageDialog*(null,"Не все поля заполнены","Exception",JOptionPane.*PLAIN\_MESSAGE*);  
 }  
 }  
 });  
  
 JButton remove = new JButton("Удалить");  
 remove.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 try{  
 int index = table1.getSelectedRow();  
 if(index==-1){throw new SimpleException("Не выбрана строка");}  
 int rowCount = tableModel.getRowCount();  
 int curIndex = dataList.size()-rowCount+index;  
 tableModel.removeRow(index);  
 dataList.remove(curIndex);  
 }catch (SimpleException ex){}  
 }  
 });  
  
 JButton solve = new JButton("Рассчитать");  
 solve.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e){  
 try {  
 int index = table1.getSelectedRow();  
 if(index==-1){throw new SimpleException("Не выбрана строка");}  
 int rowCount = tableModel.getRowCount();  
 int curIndex = dataList.size() - rowCount + index;  
 dataList.get(curIndex).integralFunc();  
 tableModel.setValueAt(dataList.get(curIndex).getResult(), index, 3);  
 }  
 catch (SimpleException ex){}  
 }  
 });  
  
 JButton clear = new JButton("Очистить");  
 clear.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 while (tableModel.getRowCount()!=0){  
 tableModel.removeRow(0);  
 }  
 }  
 });  
  
 JButton fill = new JButton("Заполнить");  
 fill.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 while (tableModel.getRowCount()!=0){  
 tableModel.removeRow(0);  
 }  
 for(RecordIntegral i : dataList){  
 tableModel.addRow(new Object[]{i.getUpLimit(), i.getDownLimit(), i.getStep(),i.getResult()});  
 }  
 }  
 });  
  
 JPanel mainPanel = new JPanel();  
 mainPanel.setLayout(new BorderLayout());  
  
 JPanel title = new JPanel();  
 title.setLayout(new BoxLayout(title,BoxLayout.*X\_AXIS*));  
  
 JPanel first = new JPanel();  
 JPanel second = new JPanel();  
 JPanel third = new JPanel();  
 first.setLayout(new BoxLayout(first,BoxLayout.*Y\_AXIS*));  
 second.setLayout(new BoxLayout(second,BoxLayout.*Y\_AXIS*));  
 third.setLayout(new BoxLayout(third,BoxLayout.*Y\_AXIS*));  
 first.add(label1);  
 first.add(upLimitField);  
 second.add(label2);  
 second.add(downLimitField);  
 third.add(label3);  
 third.add(stepField);  
 title.add(first);  
 title.add(second);  
 title.add(third);  
  
 JPanel tablePanel = new JPanel();  
 tablePanel.setLayout(new BorderLayout());  
 tablePanel.add(new JScrollPane(table1),BorderLayout.*CENTER*);  
  
 JPanel buttons = new JPanel();  
 buttons.add(add);  
 buttons.add(remove);  
 buttons.add(fill);  
 buttons.add(clear);  
 buttons.add(solve);  
  
 mainPanel.add(title,BorderLayout.*NORTH*);  
 mainPanel.add(tablePanel,BorderLayout.*CENTER*);  
 mainPanel.add(buttons,BorderLayout.*SOUTH*);  
 getContentPane().add(mainPanel);  
 setSize(510,400);  
 setVisible(true);  
 }  
 public static void main(String[] args) {  
 new LabApp();  
 }  
}