Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №2

по курсу «Разработка кроссплатформенных приложений»

на тему «**Работа с коллекциями объектов**»

**Выполнили студенты группы 21ввв1:**

Вартанов А.

Федулов Е.

**Приняли**

Юрова О.В.

Карамышева Н.С.

Пенза 2023

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

**Задание на лабораторную работу**

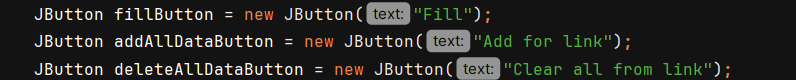
Модифицировать приложение из предыдущей лабораторной работы, реализовав хранение данных таблицы с использованием библиотеки коллекций. Для этого реализовать класс RecIntegral, способный хранить одну запись таблицы. Для в качестве класса-коллекции выбрать LinkedList. Кроме того, добавить пару кнопок: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

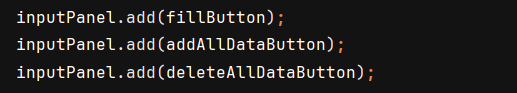
Вариант:

|  |  |
| --- | --- |
| Номер варианта | Функция |
| 2 |  |

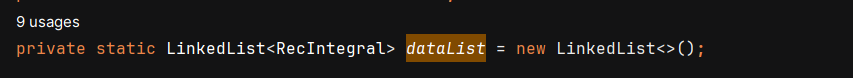
**Ход выполнения:**

Добавление новых кнопок:

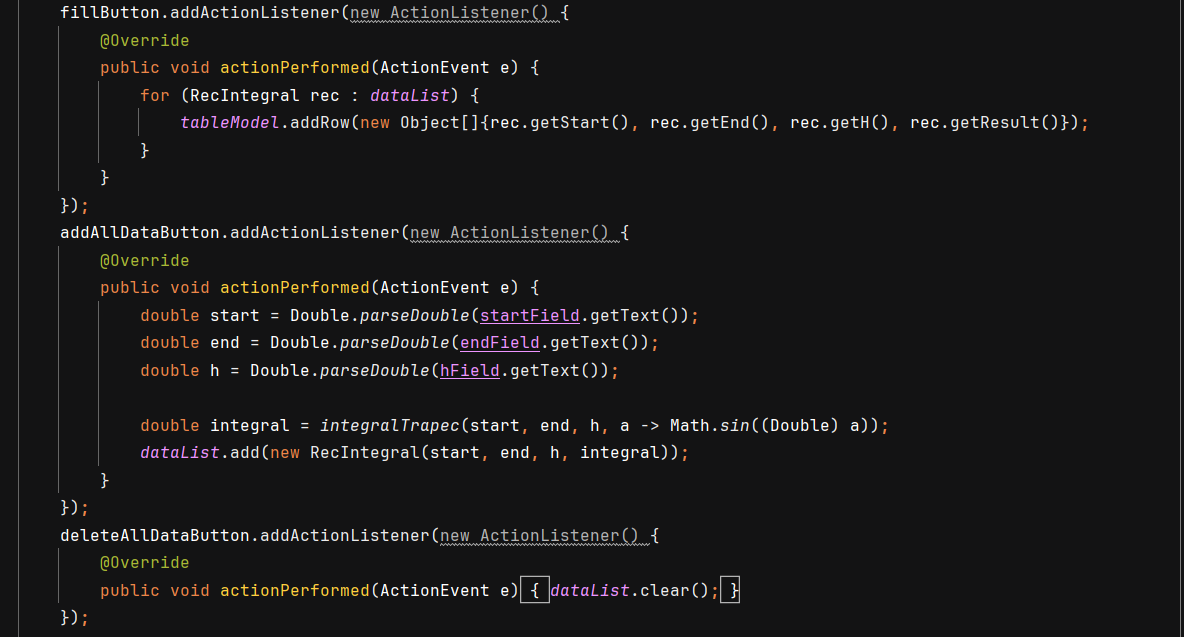




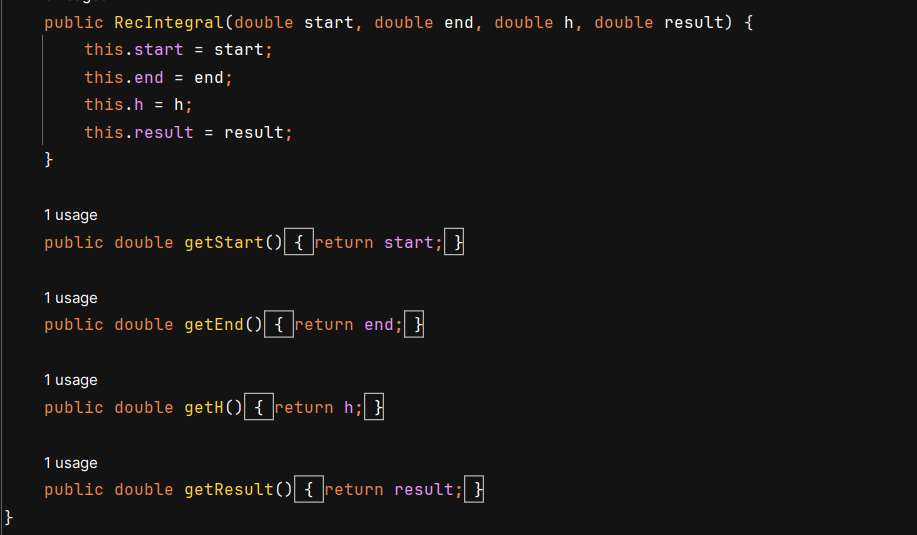
Создание LinkedList:



Реализация кнопок:



Реализация RecIntegral:



**Листинг:**

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.LinkedList;

import java.util.function.Function;

public class IntegrationCalculator{

private static DefaultTableModel tableModel;

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

public static void main(String[] args) {

JFrame frame = new JFrame("Numerical Integration of sin(x)");

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setSize(500, 300);

tableModel = new DefaultTableModel();

tableModel.addColumn("Start");

tableModel.addColumn("End");

tableModel.addColumn("h");

tableModel.addColumn("Result");

// Заполнение коллекции dataList с предварительно заданными данными

double start1 = 0.0;

double end1 = Math.PI / 2;

double h1 = 0.1;

double result1 = integralTrapec(start1, end1, h1, a3 -> Math.sin((Double) a3));

dataList.add(new RecIntegral(start1, end1, h1, result1));

double start2 = Math.PI / 2;

double end2 = Math.PI;

double h2 = 0.1;

double result2 = integralTrapec(start2, end2, h2, a2 -> Math.sin((Double) a2));

dataList.add(new RecIntegral(start2, end2, h2, result2));

double start3 = Math.PI;

double end3 = 3 \* Math.PI / 2;

double h3 = 0.1;

double result3 = integralTrapec(start3, end3, h3, a1 -> Math.sin((Double) a1));

dataList.add(new RecIntegral(start3, end3, h3, result3));

double start4 = 3 \* Math.PI / 2;

double end4 = 2 \* Math.PI;

double h4 = 0.1;

double result4 = integralTrapec(start4, end4, h4, a -> Math.sin((Double) a));

dataList.add(new RecIntegral(start4, end4, h4, result4));

JTable table = new JTable(tableModel){

@Override

public boolean isCellEditable(int i, int i1){

return false;

}

};

JScrollPane scrollPane = new JScrollPane(table);

JPanel inputPanel = new JPanel(new GridLayout(6, 2));

JLabel startLabel = new JLabel("Start:");

JTextField startField = new JTextField();

JLabel endLabel = new JLabel("End:");

JTextField endField = new JTextField();

JLabel hLabel = new JLabel("h:");

JTextField hField = new JTextField();

JButton calculateButton = new JButton("Calculate");

JButton deleteButton = new JButton("Delete");

JButton clearButton = new JButton("Clear");

JButton fillButton = new JButton("Fill");

JButton addAllDataButton = new JButton("Add for link");

JButton deleteAllDataButton = new JButton("Clear all from link");

inputPanel.add(startLabel);

inputPanel.add(startField);

inputPanel.add(endLabel);

inputPanel.add(endField);

inputPanel.add(hLabel);

inputPanel.add(hField);

inputPanel.add(calculateButton);

inputPanel.add(deleteButton);

inputPanel.add(clearButton);

inputPanel.add(fillButton);

inputPanel.add(addAllDataButton);

inputPanel.add(deleteAllDataButton);

calculateButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

double a = Double.parseDouble(startField.getText());

double b = Double.parseDouble(endField.getText());

double h = Double.parseDouble(hField.getText());

double integral = integralTrapec(a, b, h, a1 -> Math.sin((Double) a1));

addToTable(a, b, h, integral);

}

});

deleteButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

int selectedRow = table.getSelectedRow();

if (selectedRow != -1) {

tableModel.removeRow(selectedRow);

dataList.remove(selectedRow);

}

}

});

clearButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

tableModel.setRowCount(0);

}

});

fillButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

for (RecIntegral rec : dataList) {

tableModel.addRow(new Object[]{rec.getStart(), rec.getEnd(), rec.getH(), rec.getResult()});

}

}

});

addAllDataButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

double start = Double.parseDouble(startField.getText());

double end = Double.parseDouble(endField.getText());

double h = Double.parseDouble(hField.getText());

double integral = integralTrapec(start, end, h, a -> Math.sin((Double) a));

dataList.add(new RecIntegral(start, end, h, integral));

}

});

deleteAllDataButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

dataList.clear();

}

});

frame.add(inputPanel, BorderLayout.NORTH);

frame.add(scrollPane, BorderLayout.CENTER);

frame.setVisible(true);

}

public static double integralTrapec(double a, double b, double h, Function function) {

double area = 0;

for (int i = 0; a + i \* h < b; i++) {

area += h \* (0.5 \* (double)function.apply(a + i \* h) + (double)function.apply(a + (i + 1) \* h));

}

return area;

}

public static void addToTable(double a, double b, double h, double result) {

tableModel.addRow(new Object[]{a, b, h, result});

dataList.add(new RecIntegral(a, b, h, result));

}

}

class RecIntegral {

private double start;

private double end;

private double h;

private double result;

public RecIntegral(double start, double end, double h, double result) {

this.start = start;

this.end = end;

this.h = h;

this.result = result;

}

public double getStart() {

return start;

}

public double getEnd() {

return end;

}

public double getH() {

return h;

}

public double getResult() {

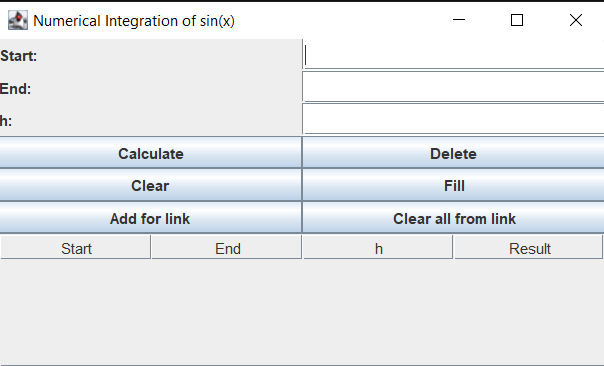
return result;

}

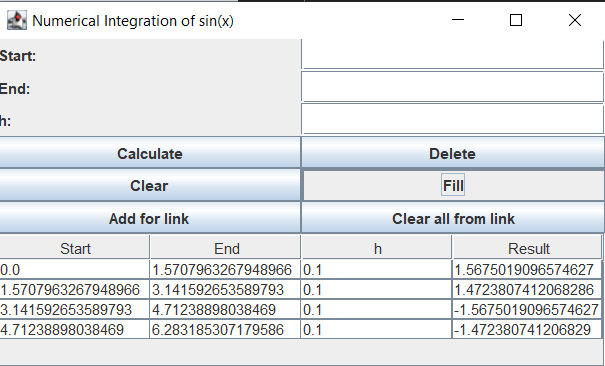
}

**Результаты работы программы:**

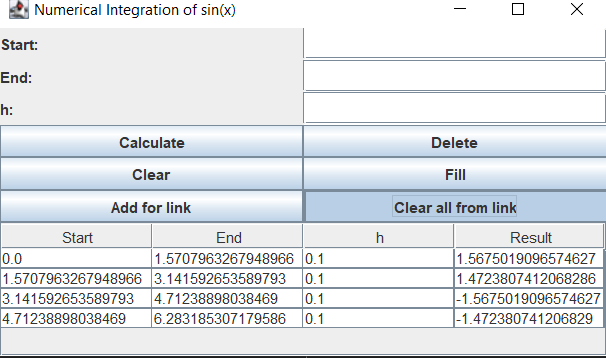
Обновлённая версия программы:

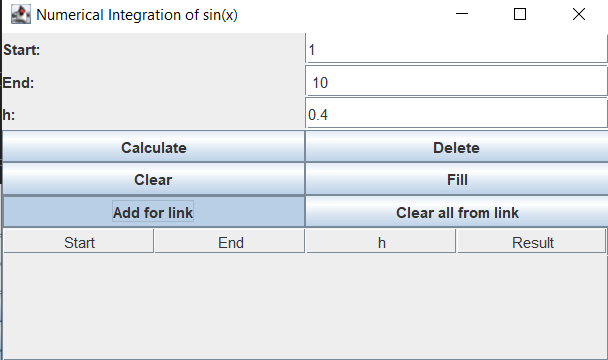


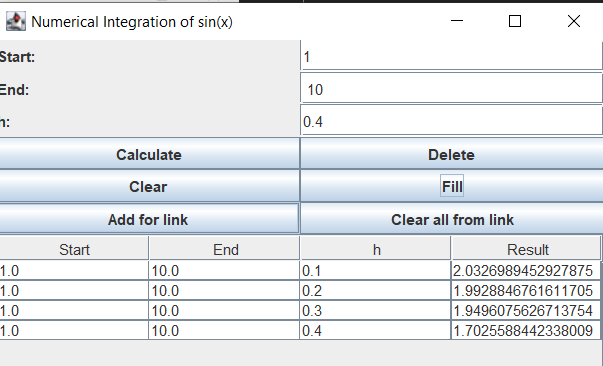
Вывели сохранённую при запуске программы сохранённую таблицу:



Очистив Linked list от данных мы заполнили новыми вычислениями таблицу:







**Вывод**: Мы модифицировали приложение, реализовав хранение данных таблицы с использованием библиотеки коллекции. Реализовали класс RecIntegral, способный хранить одну запись таблицы.