Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №2

по дисциплине: "Разработка кроссплатформенных приложений"

на тему: " Работа с коллекциями объектов "

Выполнили:

Студенты группы 21ВВB1

Кирьянов В.Е.

Аляев А.О.

Приняли

К.т.н. Юрова О.В.

К.т.н. Карамышева Н.С.

Пенза 2024

Название работы: «Работа с коллекциями объектов».

**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 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. Оформление лабораторной работы должно быть выполнено в соответствии с требованиями, приведенными в Приложении 2.

Ход работы:

Реализовали класс «RecIntegral», способный хранить одну запись таблицы.

Создали две кнопки «Очистить» и «Заполнить». По кнопке «Заполнить» реализовали заполнение данными из LinkedList. По кнопке «Очистить» просто очищаем таблицу.

Листинг программы.

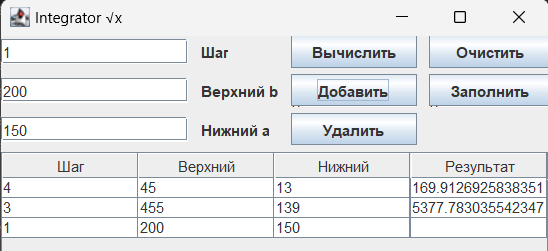
Class RecIntegral.

import java.util.Objects;  
public class RecIntegral {  
 public String step ;  
 public String b;  
 public String a;  
 public String res;  
 public RecIntegral (String step, String b, String a, String res){  
 this.step = step;  
 this.b = b;  
 this.a = a;  
 this.res = res;  
 }  
 public void getComp(String step, String b, String a, String res){  
 if(Objects.equals(step, this.step) && Objects.equals(b, this.b) && Objects.equals(a, this.a)){  
 this.res = res;  
 }  
 }  
}

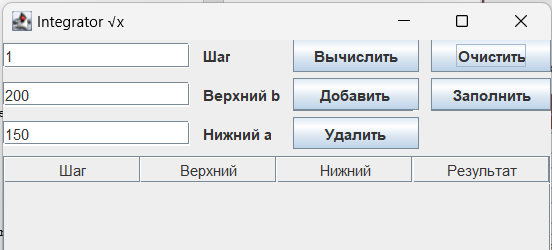
Class NewForm.

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;  
  
public class NewForm {  
 public LinkedList<RecIntegral> TableList = new LinkedList<>();  
 private JPanel panel1;  
 private JTextField textField1;  
 private JTextField textField2;  
 private JTextField textField3;  
 private JButton CalcButton;  
 private JButton AddButton;  
 private JButton DelButton;  
 private JTable table1;  
 private JScrollPane scrollPane1;  
 private JButton clearTableButton;  
 private JButton addTableButton;  
 DefaultTableModel model = (DefaultTableModel) table1.getModel();  
 //String[] columnNames = {"Шаг", "Верхний", "Нижний", "Результат"};  
 //String[][] data = { };  
 public NewForm() {  
 CalcButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 try {  
 double a = 0; //down  
 double b = 0; //upp  
 double step = 0; //step  
 double res = 0;  
 for (int row = 0; row < table1.getRowCount(); row++) {  
 if (table1.getValueAt(row, 0) != "" || table1.getValueAt(row, 1) != "" || table1.getValueAt(row, 2) != "") {  
 step = Double.*parseDouble*(table1.getValueAt(row, 0).toString());  
 b = Double.*parseDouble*(table1.getValueAt(row, 1).toString());  
 a = Double.*parseDouble*(table1.getValueAt(row, 2).toString());  
 res = Main.*Integral*(a, b, step);  
 table1.setValueAt(res, row, 3);  
 //TableList.equals(new Object[]{String.valueOf(step), String.valueOf(b), String.valueOf(a), String.valueOf(res)});  
 RecIntegral temp = TableList.get(row);  
 temp.res = String.*valueOf*(res);  
 TableList.set(row, temp);  
  
  
 //for(RecIntegral iter: TableList) {  
 //iter.getComp(iter.step, iter.b, iter.a, String.valueOf(res));  
 //}  
 }  
 }  
 } catch (Exception ex){ System.*out*.printf("Ошибка! %s", ex);}  
 }  
 });  
 AddButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
// String[][] data = {  
// {"Шаг", "Верхний", "Нижний", "Результат"},  
// {"","","",""},  
// {"","","",""}  
// };  
 //table1.setModel(new DefaultTableModel().addRow(new Object[]{"","","",""}));  
 //table1.add  
 //DefaultTableModel model = (DefaultTableModel) table1.getModel();  
 model.addRow(new Object[]{textField1.getText(), textField2.getText(), textField3.getText()});  
 RecIntegral obj = new RecIntegral(textField1.getText(), textField2.getText(), textField3.getText(), "");  
 TableList.add(obj);  
 //table1.setValueAt(table1.getValueAt(1,0),2,0);  
 //table1.setValueAt(table1.getValueAt(1,1),2,1);  
 //table1.setValueAt(table1.getValueAt(1,2),2,2);  
 //table1.setValueAt(table1.getValueAt(1,3),2,3);  
  
 //table1.setValueAt(textField1.getText(),1,0);  
 //table1.setValueAt(textField2.getText(),1,1);  
 //table1.setValueAt(textField3.getText(),1,2);  
 //table1.setValueAt("",1,3);  
 }  
 });  
 DelButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 final int row = table1.getSelectedRow();  
 //final int column = table1.getSelectedColumn();  
 //for(int i = 0; i < 4; i++) {  
 //table1.setValueAt("", row, i);  
 try {  
 model.removeRow(row);  
 TableList.remove(row);  
 //}  
 } catch (Exception ex){}  
 }  
 });  
 clearTableButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 //TableList.clear();  
 model.getDataVector().removeAllElements();  
 model.fireTableDataChanged();  
 }  
 });  
 addTableButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 model.setRowCount(0);  
 //model.removeRow(model.getRowCount()-1);  
 //model.rowsRemoved(new DefaultTableModel(data,  
 //columnNames));  
 //System.out.println(TableList);  
 for(RecIntegral step: TableList){  
 //RecIntegral dt = new RecIntegral(step);  
 //RecIntegral Obj = new RecIntegral(step);  
 model.addRow(new Object[]{step.step,step.b,step.a,step.res});  
 //RecIntegral A = step.getStep();  
 //String value = new Object[]{step}.toString();  
 //model.addRow(new Object[]{value});  
 //model.addRow((Vector<?>) step);  
 //model.addRow(step.getStep(), step.getB(), step.getA(), step.get());  
 //step.getStep();  
 }  
 }  
 });  
 }  
 public static void main(String[] args) {  
 JFrame frame = new JFrame("Integrator √x");  
 frame.setLocationRelativeTo(null);  
 frame.setContentPane(new NewForm().panel1);  
 //frame.setContentPane(new NewForm().scrollPane1);  
 frame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 frame.pack();  
 frame.setVisible(true);  
 }  
 private void createUIComponents() {  
  
 scrollPane1 = new JScrollPane(table1);  
 //scrollPane1.max  
 scrollPane1.setPreferredSize(new Dimension(0, 150));  
  
 String[] columnNames = {"Шаг", "Верхний", "Нижний", "Результат"};  
 String[][] data = { };  
  
 table1 = new JTable(data, columnNames);  
 //table1.setBounds(30, 40, 200, 300);  
 table1.setModel(new DefaultTableModel(data,  
 columnNames  
 ) {public boolean isCellEditable(int row, int column){return (column == 0 || column == 1 || column == 2);}}  
 );  
 }  
}

Результаты программы.



Нажали кнопку «Очистить».



Нажали кнопку «Заполнить».



Вывод: изучили и научились работать с библиотекой стандартных коллекций Java Collections Framework, позволяющей хранить различные структуры данных.

.