Пензенский Государственный Университет

Кафедра «Вычислительная техника»

Отчет о выполнении работы по дисциплине

«Программирование на языке Java»

Лабораторная работа №5

Выполнили: ст. группы 20ВВП2

Педай Н.Д.

Мальков И.А.

Приняли:

Юрова О.В.

Карамышева Н.С.

Пенза, 2023

**Цель работы**: научиться создавать многопоточные приложения c

использованием стандартных средств языка Java.

**Заданиe:**

Модифицировать приложение из предыдущей лабораторной работы,

реализовав вычисление определённого интеграла в нескольких

дополнительных потоках (число потоков определяется номером варианта),

13

снимая нагрузку с основного потока и предотвращая "подвисание"

графического интерфейса. Варианты с номерами до 5 включительно

реализуют многопоточность путем наследования от класса Thread, остальные

реализуют интерфейс Runnable

**Ход работы:**

Реализовали параллельное вычисление интеграла на 7 потоках в соответствии с заданием используя интерфейс **Runnable.**

Для этого делим итерации сложения при вычислении интеграла методом трапеции на 7 частей, каждая из частей выполняется в своём потоке.

После завершения всех потоков (ожидаем завершения), результаты складываются в единый результат.

Провели проверку результатов полученных параллельным алгоритмом с ручным расчётом: результат верный.

**Листинг**:

**Файл формы**

import javax.swing.\*;  
import javax.swing.filechooser.FileNameExtensionFilter;  
import javax.swing.table.DefaultTableModel;  
import java.awt.event.\*;  
import java.io.\*;  
import java.util.ArrayList;  
  
public class GUIForm extends JDialog {  
 private int realColCnt;  
 private JMenuBar menuBar;  
 private JMenu menu, submenu;  
 private JMenuItem menuItem;  
 private JRadioButtonMenuItem rbMenuItem;  
 private JCheckBoxMenuItem cbMenuItem;  
 private JPanel contentPane;  
 private JButton buttonOK;  
 private JButton buttonCancel;  
 private JTextField textField1;  
 private JTextField textField2;  
 private JTextField textField3;  
 private JButton добавитьButton;  
 private JButton удалитьButton;  
 private JButton вычислитьButton;  
 private JTable table1;  
 private JButton заполнитьButton;  
 private JButton очиститьButton;  
 private JButton удалитьВсёButton;  
 private JButton сохранитьБинарныйButton;  
 private JButton загрузитьБинарныйButton;  
 private JButton открытьButton;  
 private JFileChooser chooser= new JFileChooser();  
 private File chosenFile;  
 ////////////////////////////////////ПЕРЕМЕННЫЕ/////////////////////////////////////////////////  
 private double[] dataT = new double[4];  
 int num =1;  
 //public RecIntegral[] dataListObject = new RecIntegral[10];  
 DefaultTableModel model = (DefaultTableModel) table1.getModel();  
 public ArrayList<RecIntegral> dataListObject= new ArrayList<RecIntegral>();  
 ////////////////////////////////////////////////////////////////////////////////////////////////  
 public GUIForm() {  
 setContentPane(contentPane);  
 setModal(true);  
 getRootPane().setDefaultButton(buttonOK);  
 setTitle("Лаба 1");  
 ////////////////  
 createMenuBar();  
  
 setTitle("Simple menu");  
 setSize(350, 250);  
 setLocationRelativeTo(null);  
 //setDefaultCloseOperation(EXIT\_ON\_CLOSE);  
 ////////////////  
 createTable();  
 buttonOK.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 onOK();  
 }  
 });  
  
 buttonCancel.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 onCancel();  
 }  
 });  
  
 добавитьButton.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 try {  
 добавитьButton();  
 } catch (NumberFormatException ex) {  
 throw new RuntimeException(ex);  
 }  
 }  
 });  
 удалитьButton.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
  
 try {  
 удалитьButton();  
 } catch (NumberFormatException ex) {  
 throw new RuntimeException(ex);  
 }  
  
 }  
 });  
  
 вычислитьButton.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 try {  
 вычислитьButton();  
 } catch (InterruptedException ex) {  
 throw new RuntimeException(ex);  
 }  
 }  
 });  
 заполнитьButton.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 заполнитьButton();  
 }  
 });  
 очиститьButton.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 очиститьButton();  
 }  
 });  
 удалитьВсёButton.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 удалитьВсёButton();  
 }  
 });  
  
  
  
 // 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 createMenuBar() {  
  
 var menuBar = new JMenuBar();  
  
 var iconOpen = new ImageIcon("src/resources/open.png");  
 var iconSave = new ImageIcon("src/resources/save.png");  
  
 var exitIcon = new ImageIcon("src/resources/exit.png");  
  
 var fileMenu = new JMenu("File");  
 fileMenu.setMnemonic(KeyEvent.*VK\_F*);  
  
 var eMenuItem = new JMenuItem("Exit", exitIcon);  
 var openMenuItem = new JMenuItem("Open", iconOpen);  
 var saveMenuItem = new JMenuItem("Save", iconSave);  
 var openBinMenuItem = new JMenuItem("Open bin", iconOpen);  
 var saveBinMenuItem = new JMenuItem("Save bin", iconSave);  
 eMenuItem.setMnemonic(KeyEvent.*VK\_E*);  
 eMenuItem.setToolTipText("Exit application");  
 eMenuItem.addActionListener((event) -> System.*exit*(0));  
  
 fileMenu.add(eMenuItem);  
 fileMenu.add(openMenuItem);  
 fileMenu.add(saveMenuItem);  
 fileMenu.add(openBinMenuItem);  
 fileMenu.add(saveBinMenuItem);  
 menuBar.add(fileMenu);  
 setJMenuBar(menuBar);  
  
 saveBinMenuItem.addActionListener(new ActionListener() { ///////////////////// сохранить бинарный файл  
 public void actionPerformed(ActionEvent e) {  
 FileNameExtensionFilter filter = new FileNameExtensionFilter("Data List Object Type", "dlot");  
 chooser.setFileFilter(filter);  
 int choice = chooser.showSaveDialog(chooser);  
 //if (choice != JFileChooser.APPROVE\_OPTION) return;  
 chosenFile = chooser.getSelectedFile();  
  
 savebinFile();  
  
 }  
 });  
  
 openBinMenuItem.addActionListener(new ActionListener() { ///////////////////// открыть бинарный файл  
 public void actionPerformed(ActionEvent e) {  
 FileNameExtensionFilter filter = new FileNameExtensionFilter("Data List Object Type", "dlot");  
 chooser.setFileFilter(filter);  
 int choice = chooser.showOpenDialog(chooser);  
 if (choice != JFileChooser.*APPROVE\_OPTION*) return;  
 chosenFile = chooser.getSelectedFile();  
 loadbinFile();  
  
  
 }  
 });  
 saveMenuItem.addActionListener(new ActionListener() { ///////////////////// сохранить файл  
 public void actionPerformed(ActionEvent e) {  
 int choice = chooser.showSaveDialog(chooser);  
 //if (choice != JFileChooser.APPROVE\_OPTION) return;  
 chosenFile = chooser.getSelectedFile();  
 try {  
 saveFile();  
 } catch (IOException ex) {  
 throw new RuntimeException(ex);  
 }  
  
 }  
 });  
 openMenuItem.addActionListener(new ActionListener() { ///////////////////// сохранить файл  
 public void actionPerformed(ActionEvent e) {  
 FileNameExtensionFilter filter = new FileNameExtensionFilter("text", "txt");  
 chooser.setFileFilter(filter);  
 int choice = chooser.showOpenDialog(chooser);  
 if (choice != JFileChooser.*APPROVE\_OPTION*) return;  
 chosenFile = chooser.getSelectedFile();  
 try {  
 loadFile();  
 } catch (IOException ex) {  
 throw new RuntimeException(ex);  
 } catch (NumberFormatException ex) {  
 throw new RuntimeException(ex);  
 }  
 }  
 });  
  
 }  
 private void onOK() {  
 // add your code here  
 createTable();  
 }  
  
 private void onCancel() {  
 // add your code here if necessary  
 dispose();  
 }  
  
 ////////////////////////////////////РАБОТА С ДАННЫМИ/////////////////////////////////////////////////  
 private void добавитьButton() throws NumberFormatException {  
 dataListObject.add(num-1, new RecIntegral());  
  
 dataListObject.get(num-1).setDataA(Double.*valueOf*(  
 textField1.getText()));  
  
 dataListObject.get(num-1).setDataB(Double.*valueOf*(  
 textField3.getText()));  
  
 dataListObject.get(num-1).setDataC(Double.*valueOf*(  
 textField2.getText()));  
 if(dataListObject.get(num-1).getDataA()<dataListObject.get(num-1).getDataB()){  
 throw new NumberFormatException();  
 }  
 dataListObject.get(num-1).setDataD(0.0);  
  
 model.addRow(dataListObject.get(num-1).getDataList());  
  
 textField1.setText("");  
 textField2.setText("");  
 textField3.setText("");  
 }  
 private void удалитьButton() throws NumberFormatException {  
 int rowNamber;  
 rowNamber=table1.getSelectedRow();  
 model.removeRow(rowNamber);  
 //dataListObject.get(rowNamber);  
 dataListObject.remove(rowNamber);  
 num--;  
 }  
 private void удалитьВсёButton() {  
 int countData = dataListObject.size();  
 for(int i=0;i<countData;i++) {  
 model.removeRow(i);  
 dataListObject.remove(i);  
  
 }  
 }  
 private void заполнитьButton() {  
 int countData = dataListObject.size();  
  
 num=1;  
 for(int i=0; i<countData;i++) {  
 model.addRow(dataListObject.get(i).getDataList());  
 num++;  
 }  
 }  
  
 private void очиститьButton() {  
  
 model.setRowCount(0);  
 }  
 private void вычислитьButton() throws InterruptedException {  
  
 dataListObject.get(num-1).Trap();  
 // saveFile();  
 model.removeRow(num-1);  
 model.addRow(dataListObject.get(num-1).getDataList());  
  
 num++;  
  
 }  
  
 public void createTable(){  
  
 model.addColumn("Верхняя граница интегрирования");  
 model.addColumn("Нижняя граница интегрирования");  
 model.addColumn("Шаг интегрирования");  
 model.addColumn("Результат");  
 }  
 public void открытьButton(){  
  
 int choice = chooser.showOpenDialog(chooser);  
  
 if (choice != JFileChooser.*APPROVE\_OPTION*) return;  
 File chosenFile = chooser.getCurrentDirectory();  
  
 }  
  
 public void savebinFile(){  
  
 try(ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream(chosenFile)))  
 {  
 oos.writeObject(dataListObject);  
 System.*out*.println("File has been written");  
 }  
 catch(Exception ex){  
  
 System.*out*.println(ex.getMessage());  
 }  
  
  
 }  
 public void loadbinFile() {  
  
 try(ObjectInputStream ois = new ObjectInputStream(new FileInputStream(chosenFile)))  
 {  
  
 dataListObject=((ArrayList<RecIntegral>)ois.readObject());  
 }  
 catch(Exception ex){  
  
 System.*out*.println(ex.getMessage());  
 }  
 заполнитьButton();  
 }  
 public void saveFile() throws IOException {  
 int countData = dataListObject.size();  
  
 FileWriter myfile = new FileWriter(chosenFile);  
  
 for(int i=0;i<countData;i++) {  
 myfile.write(dataListObject.get(i).getDataA().toString() + "," +  
 dataListObject.get(i).getDataB().toString() + "," +  
 dataListObject.get(i).getDataC().toString() + "," +  
 dataListObject.get(i).getDataD().toString()+"\n");  
 }  
  
 myfile.flush();  
 myfile.close();  
 }  
 public void loadFile() throws IOException, NumberFormatException {  
 FileReader myfile = new FileReader(chosenFile);  
 BufferedReader reader = new BufferedReader(myfile);  
 int i=0;  
 dataListObject.add(i, new RecIntegral());  
 String line = reader.readLine();  
 String[] dblArray = line.split(",");  
  
 dataListObject.get(0).setDataA(Double.*valueOf*(dblArray[0]));  
 dataListObject.get(0).setDataB(Double.*valueOf*(dblArray[1]));  
 dataListObject.get(0).setDataC(Double.*valueOf*(dblArray[2]));  
 dataListObject.get(0).setDataD(Double.*valueOf*(dblArray[3]));  
  
  
 while (line != null) {  
 // считываем остальные строки в цикле  
 i++;  
 dataListObject.add(i, new RecIntegral());  
 line = reader.readLine();  
 if(line==null)break;  
 dblArray = line.split(",");  
 dataListObject.get(i).setDataA(Double.*valueOf*(dblArray[0]));  
 dataListObject.get(i).setDataB(Double.*valueOf*(dblArray[1]));  
 dataListObject.get(i).setDataC(Double.*valueOf*(dblArray[2]));  
 dataListObject.get(i).setDataD(Double.*valueOf*(dblArray[3]));  
  
 }  
  
 заполнитьButton();  
 }  
  
 /////////////////////////////////////////////////////////////////////////////////////////////////////////  
 public static void main(String[] args) {  
 GUIForm dialog = new GUIForm();  
 dialog.pack();  
 dialog.setVisible(true);  
 dialog.setName("laba1");  
  
 System.*exit*(0);  
 }  
}

**Вывод:** научились создавать многопоточные приложения c

использованием стандартных средств языка Java.