Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №6

по курсу «Программирование на языке JAVA»

на тему «Сетевое взаимодействие в Java»

Выполнили:

студенты группы 20ВВП1

Лазутин Д.Д.

Мартынов Н.В.

Приняли:

Юрова О.В.

Карамышева Н.С.

Пенза 2023

**Цель работы:** научиться создавать клиент-серверные приложения c использованием стандартных классов Java.

**Лабораторное задание:**

Модифицировать приложение из предыдущей лабораторной работы, реализовав клиент-серверную архитектуру, обеспечивающую распределенное вычисление определенного интеграла на нескольких вычислительных узлах (клиентах) при этом каждый узел использует несколько нитей, как в предыдущей работе. Сервер не занимается вычислениями, а лишь реализует взаимодействие с пользователем и агрегацию результатов вычислений от клиентов. Нечетные варианты используют протокол TCP, а четные UDP. Оформление лабораторной работы должно быть выполнено в соответствии с требованиями, приведенными в Приложении 2.

**Листинг (Client.java):**

package com.mycompany.testing;

import java.util.ArrayList;

import javax.swing.JOptionPane;

import javax.swing.table.DefaultTableModel;

import java.io.\*;

import javax.swing.JFileChooser;

import java.util.Scanner;

import java.util.Arrays;

import java.net.\*;

public class Lab1 extends javax.swing.JFrame implements Serializable {

public Lab1() {

initComponents();

}

ArrayList<RecIntegral> list = new ArrayList<RecIntegral>();

ArrayList<RecIntegral> listB = new ArrayList<RecIntegral>();

ArrayList <String> file\_values = new ArrayList<String>();

JFileChooser window;

ScannerException error = new ScannerException("");

File myfile;

FileWriter filewrite;

FileReader fileread;

Scanner scanner;

public static final String ANSI\_GREEN = "\u001B[32m";

public static final String ANSI\_BLUE = "\u001B[34m";

public static final String ANSI\_RED = "\u001B[31m";

public static final String ANSI\_RESET = "\u001B[0m";

public void InputText() {

window = new JFileChooser();

window.setCurrentDirectory(new File(".")); // установить директорию по умолчанию

window.setDialogTitle("Выберите файл");

window.setMultiSelectionEnabled(false); //возможность выбора нескольких файлов

window.setFileSelectionMode(JFileChooser.FILES\_ONLY); //Можно выбрать только файл

window.setSelectedFile(new File("Result\_Text.txt"));

window.showDialog(window, "Сохранить"); //открытие диалогового окна

myfile = window.getSelectedFile();

try {

filewrite = new FileWriter(myfile, true);

for (int i=0; i<jTable2.getRowCount(); i++) {

for (int j = 0; j<jTable2.getColumnCount(); j++) {

filewrite.write(String.valueOf(jTable2.getValueAt(i, j)));

filewrite.write(" ");

}

filewrite.write("\n");

}

filewrite.close();

} catch(IOException e){

System.out.println("Ошибка открытия файла!");

}

}

public void OutputText () {

window = new JFileChooser();

window.setCurrentDirectory(new File(".")); // установить директорию по умолчанию

window.setDialogTitle("Выберите файл");

window.setMultiSelectionEnabled(false); //возможность выбора нескольких файлов

window.setFileSelectionMode(JFileChooser.FILES\_ONLY); //Можно выбрать только файлы

window.setSelectedFile(new File("Result\_Text.txt"));

window.showDialog(window, "Открыть"); //открытие диалогового окна

myfile = window.getSelectedFile();

String[] values;

DefaultTableModel model2 = (DefaultTableModel)jTable2.getModel();

try {

//fileread = new FileReader(myfile);

scanner = new Scanner(myfile);

list.clear();

while (scanner.hasNextLine()) {

values = scanner.nextLine().split(" ");

model2.addRow(new Object[] {values[0], values[1], values[2], values[3]});

}

scanner.close();

} catch (IOException e) {

System.out.println("Ошибка открытия файла");

}

}

public void OutputBinary() {

window = new JFileChooser();

window.setCurrentDirectory(new File(".")); // установить директорию по умолчанию

window.setDialogTitle("Выберите файл");

window.setMultiSelectionEnabled(false); //возможность выбора нескольких файлов

window.setFileSelectionMode(JFileChooser.FILES\_ONLY); //Можно выбрать только файл

window.setSelectedFile(new File("Result.ser"));

window.showDialog(window, "Сохранить"); //открытие диалогового окна

//int check = JFileChooser.APPROVE\_OPTION;

myfile = window.getSelectedFile();

for (int i = 0; i < jTable2.getRowCount(); i++) {

for (int j = 0; j < jTable2.getColumnCount(); j++) {

file\_values.add(String.valueOf(jTable2.getValueAt(i, j)));

}

}

ObjectOutputStream out = null;

try {

out = new ObjectOutputStream(new BufferedOutputStream(new FileOutputStream(myfile)));

out.writeObject(file\_values);

out.close();

} catch(IOException e) {

e.printStackTrace();

}

}

public void InputBinary() {

window = new JFileChooser();

window.setCurrentDirectory(new File(".")); // установить директорию по умолчанию

window.setDialogTitle("Выберите файл");

window.setMultiSelectionEnabled(false); //возможность выбора нескольких файлов

window.setFileSelectionMode(JFileChooser.FILES\_ONLY); //Можно выбрать только файл

window.setSelectedFile(new File("Result.ser"));

window.showDialog(window, "Сохранить"); //открытие диалогового окна

myfile = window.getSelectedFile();

file\_values.clear();

//list.clear();

try {

ObjectInputStream in = null;

in = new ObjectInputStream(new BufferedInputStream(new FileInputStream(myfile)));

file\_values = (ArrayList<String>)in.readObject();

in.close();

} catch(IOException | ClassNotFoundException e) {

e.printStackTrace();

}

}

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jPanel1 = new javax.swing.JPanel();

jTextField1 = new javax.swing.JTextField();

jTextField2 = new javax.swing.JTextField();

jTextField3 = new javax.swing.JTextField();

jButton1 = new javax.swing.JButton();

jButton2 = new javax.swing.JButton();

jButton3 = new javax.swing.JButton();

jLabel1 = new javax.swing.JLabel();

jLabel2 = new javax.swing.JLabel();

jLabel3 = new javax.swing.JLabel();

jScrollPane2 = new javax.swing.JScrollPane();

jTable2 = new javax.swing.JTable();

jButton4 = new javax.swing.JButton();

jButton5 = new javax.swing.JButton();

jButton6 = new javax.swing.JButton();

jButton7 = new javax.swing.JButton();

jButton8 = new javax.swing.JButton();

jButton9 = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

jPanel1.setBackground(new java.awt.Color(242, 242, 229));

jTextField1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jTextField1ActionPerformed(evt);

}

});

jTextField2.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jTextField2ActionPerformed(evt);

}

});

jTextField3.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jTextField3ActionPerformed(evt);

}

});

jButton1.setText("Добавить");

jButton1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton1ActionPerformed(evt);

}

});

jButton2.setText("Удалить");

jButton2.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton2ActionPerformed(evt);

}

});

jButton3.setText("Вычислить");

jButton3.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton3ActionPerformed(evt);

}

});

jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);

jLabel1.setText("Шаг");

jLabel2.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);

jLabel2.setText("Верх. гр.");

jLabel3.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);

jLabel3.setText("Нижн. граница");

jTable2.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

"Нижняя граница", "Верхняя граница", "Шаг интегрирования", "Результат"

}

));

jScrollPane2.setViewportView(jTable2);

if (jTable2.getColumnModel().getColumnCount() > 0) {

jTable2.getColumnModel().getColumn(0).setResizable(false);

jTable2.getColumnModel().getColumn(1).setResizable(false);

jTable2.getColumnModel().getColumn(2).setResizable(false);

jTable2.getColumnModel().getColumn(3).setResizable(false);

}

jButton4.setText("Очистить всё");

jButton4.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton4ActionPerformed(evt);

}

});

jButton5.setText("Заполнить из коллекции");

jButton5.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton5ActionPerformed(evt);

}

});

jButton6.setText("Сохранить в текстовом файле");

jButton6.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton6ActionPerformed(evt);

}

});

jButton7.setText("Заполнить из текстового файла");

jButton7.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton7ActionPerformed(evt);

}

});

jButton8.setText("Сохранить в бинарном файле");

jButton8.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton8ActionPerformed(evt);

}

});

jButton9.setText("Заполнить из бинарного файла");

jButton9.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton9ActionPerformed(evt);

}

});

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);

jPanel1.setLayout(jPanel1Layout);

jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addGap(31, 31, 31)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(jScrollPane2)

.addComponent(jTextField3, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.PREFERRED\_SIZE, 168, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(jTextField1)

.addComponent(jLabel3, javax.swing.GroupLayout.DEFAULT\_SIZE, 206, Short.MAX\_VALUE))

.addGap(18, 18, 18)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addComponent(jTextField2, javax.swing.GroupLayout.PREFERRED\_SIZE, 206, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 174, Short.MAX\_VALUE))

.addGroup(jPanel1Layout.createSequentialGroup()

.addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED\_SIZE, 206, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jLabel1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))))

.addGroup(jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)

.addGroup(javax.swing.GroupLayout.Alignment.LEADING, jPanel1Layout.createSequentialGroup()

.addComponent(jButton7)

.addGap(18, 18, 18)

.addComponent(jButton9, javax.swing.GroupLayout.PREFERRED\_SIZE, 1, Short.MAX\_VALUE))

.addGroup(jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(jButton1, javax.swing.GroupLayout.PREFERRED\_SIZE, 206, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jButton6, javax.swing.GroupLayout.PREFERRED\_SIZE, 206, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(18, 18, 18)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(jButton2, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jButton8, javax.swing.GroupLayout.DEFAULT\_SIZE, 206, Short.MAX\_VALUE))))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(jButton4, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jButton5, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jButton3, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))))

.addGap(0, 38, Short.MAX\_VALUE))

);

jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addGap(14, 14, 14)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel1)

.addComponent(jLabel3)

.addComponent(jLabel2))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jTextField2, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jTextField3, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jButton1)

.addComponent(jButton2)

.addComponent(jButton3))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jButton6)

.addComponent(jButton8)

.addComponent(jButton4))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jButton7)

.addComponent(jButton9)

.addComponent(jButton5))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED\_SIZE, 203, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(30, 30, 30))

);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addContainerGap())

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addContainerGap())

);

pack();

}// </editor-fold>

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

DefaultTableModel model = (DefaultTableModel) jTable2.getModel();

RecIntegral temp;

//listB.clear();

Socket socket;

DataOutputStream out;

DataInputStream in;

try {

socket = new Socket("localhost", 8000);

System.out.println(ANSI\_GREEN + "Connection to the server with host " +

socket.getRemoteSocketAddress() + " has been successfully completed!\n" + ANSI\_RESET);

out = new DataOutputStream(socket.getOutputStream());

in = new DataInputStream(socket.getInputStream());

int rowCount = model.getRowCount();

out.writeInt(rowCount);

for (int j = 0; j < model.getRowCount(); j++) {

double d1 = Double.parseDouble(String.valueOf(model.getValueAt(j, 0)));

double d2 = Double.parseDouble(String.valueOf(model.getValueAt(j, 1)));

double d3 = Double.parseDouble(String.valueOf(model.getValueAt(j, 2)));

out.writeDouble(d1);

out.writeDouble(d2);

out.writeDouble(d3);

System.out.println("The client sent the following data to the server: " + ANSI\_RED + d1 + ANSI\_RESET + ", " +

ANSI\_GREEN + d2 + ANSI\_RESET + ", " + ANSI\_BLUE + d3 + ANSI\_RESET);

double res = in.readDouble();

System.out.println("The client received the following data from the server: " + ANSI\_GREEN + res + ANSI\_RESET + "\n");

model.setValueAt(res, j, 3);

list.add(new RecIntegral(d1, d2, d3, res));

//System.out.println(res);

//socket.close();

}

socket.close();

} catch(IOException ex) {

System.out.println(ex);

}

// for (i=0; i<model.getRowCount(); i++) {

//temp = listB.get(i);

//new JThread(temp.getDownLimit(), temp.getUpLimit(), temp.getStep(), model, i, temp).start();

//}

}

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

if (jTextField1.getText().isEmpty() || jTextField2.getText().isEmpty() || jTextField3.getText().isEmpty()){

JOptionPane.showMessageDialog(this, "Пожалуйста, заполните все поля полностью!");

}

else {

DefaultTableModel model = (DefaultTableModel)jTable2.getModel();

Double num1= Double.valueOf(String.valueOf(jTextField1.getText()));

Double num2= Double.valueOf(String.valueOf(jTextField2.getText()));

Double num3= Double.valueOf(String.valueOf(jTextField3.getText()));

try {

error.CheckInputValues(num1, num2, num3);

error.CheckUpDowmLimits(num1, num2);

//listB.add(new RecIntegral(num1, num2, num3));

model.addRow(new Object[]{jTextField1.getText(), jTextField2.getText(), jTextField3.getText()});

jTextField1.setText("");

jTextField2.setText("");

jTextField3.setText("");

}

catch (ScannerException ex) {

String info = ex.getMessage();

JOptionPane.showMessageDialog(null, info);

}

}

}

private void jTextField3ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

private void jTextField2ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

private void jTextField1ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

try {

error.CheckBeforeDelete(jTable2.getSelectedRow());

DefaultTableModel model = (DefaultTableModel)jTable2.getModel();

list.remove(jTable2.getSelectedRow());

model.removeRow(jTable2.getSelectedRow());

}

catch (ScannerException ex) {

JOptionPane.showMessageDialog(this, ex.getMessage());

}

}

private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {

DefaultTableModel model = (DefaultTableModel)jTable2.getModel();

model.setRowCount(0);

}

private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

DefaultTableModel model = (DefaultTableModel)jTable2.getModel();

for(int i=0; i<list.size(); i++)

{

model.addRow(new Object[] {list.get(i).down, list.get(i).up, list.get(i).step, list.get(i).res});

}

}

private void jButton7ActionPerformed(java.awt.event.ActionEvent evt) {

OutputText();

int check = 0;

if(check == JFileChooser.APPROVE\_OPTION) {

System.out.println("-----------------------------------------------"

+ "-------------------------\nДанные успешно сохранены в текстовый файл!");

}

else if(check < JFileChooser.CANCEL\_OPTION) {

System.out.println("-----------------------------------------------"

+ "-------------------------\nВыбор текстового файла отменён!");

}

else if(check > JFileChooser.ERROR\_OPTION) {

System.out.println("-----------------------------------------------"

+ "-------------------------\nОшибка при открытии файла!");

}

}

private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {

InputText();

System.out.println("--------------------------------------------"

+ "----------------------------\nДанные успешно сохранены в текстовый файл!");

}

private void jButton8ActionPerformed(java.awt.event.ActionEvent evt) {

OutputBinary();

System.out.println("-----------------------------------------------"

+ "-------------------------\nДанные успешно сохранены в бинарный файл!");

}

private void jButton9ActionPerformed(java.awt.event.ActionEvent evt) {

InputBinary();

DefaultTableModel model = (DefaultTableModel)jTable2.getModel();

int p = 0;

while(p < file\_values.size())

{

model.addRow(new Object[] {file\_values.get(p), file\_values.get(p+1), file\_values.get(p+2), file\_values.get(p+3)});

p = p + 4;

}

System.out.println("-----------------------------------------------"

+ "-------------------------\nДанные успешно сохранены в бинарный файл!");

}

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Lab1.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Lab1.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Lab1.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Lab1.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new Lab1().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton jButton1;

private javax.swing.JButton jButton2;

private javax.swing.JButton jButton3;

private javax.swing.JButton jButton4;

private javax.swing.JButton jButton5;

private javax.swing.JButton jButton6;

private javax.swing.JButton jButton7;

private javax.swing.JButton jButton8;

private javax.swing.JButton jButton9;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JPanel jPanel1;

private javax.swing.JScrollPane jScrollPane2;

private javax.swing.JTable jTable2;

private javax.swing.JTextField jTextField1;

private javax.swing.JTextField jTextField2;

private javax.swing.JTextField jTextField3;

// End of variables declaration

}

**Листинг (RecIntegral.java):**

package com.mycompany.testing;

import java.util.ArrayList;

import javax.swing.JOptionPane;

import java.io.Serializable;

public class RecIntegral extends Lab1 implements Serializable {

Double values[];

double down;

double up;

double step;

double res;

public RecIntegral (double a, double b, double h, double result) {

down = a;

up = b;

step = h;

res = result;

values = new Double[4];

values[0] = down;

values[1] = up;

values[2] = step;

values[3] = res;

}

public RecIntegral(double a, double b, double h) {

down = a;

up = b;

step = h;

}

public double getDownLimit() {

return down;

}

public double getUpLimit() {

return up;

}

public double getStep() {

return step;

}

public void setResult(double result) {

res = result;

}

public Double[] getValue() {

return values;

}

}

**Листинг (ScannerException.java):**

package com.mycompany.testing;

public class ScannerException extends Exception {

public ScannerException (String str) {

super(str);

}

void CheckInputValues (Double a, Double b, Double h) throws ScannerException {

if ((a>=0.000001 && a<=1000000) && (b>=0.000001 && b<=1000000) && (h>=0.000001 && h<=1000000)) {

System.out.println("Проверка #1 пройдена!");

}

else {

System.out.println("Проверка #1 НЕ пройдена!");

throw new ScannerException("Что-то пошло не так... Вы вышли за диапазон (0.000001 - 1000000)");

}

}

void CheckUpDowmLimits (Double a, Double b) throws ScannerException {

if (b<=a) {

System.out.println("Проверка #2 НЕ пройдена!");

System.out.println("\n-----------------------------");

throw new ScannerException("Упс... Верхняя граница не может меньше нижней!");

}

else {

System.out.println("Проверка #2 пройдена!");

System.out.println("\n-----------------------------");

}

}

void CheckBeforeDelete (int index) throws ScannerException {

if (index == -1) {

throw new ScannerException("Выберите строку для удаления и повторите попытку!");

}

else {

System.out.println("Строка успешно удалена!");

}

}

void CheckBeforeGettingResult (int index) throws ScannerException {

if (index == -1) {

throw new ScannerException("Не забудьте выбрать строку для вычисления :)");

}

else {

System.out.println("Результат получен!");

}

}

}

**Листинг (JThread.java):**

package com.mycompany.testing;

import javax.swing.JOptionPane;

import javax.swing.table.DefaultTableModel;

public class JThread extends Thread {

double sum=0;

double up;

double a1;

double b1;

double h1;

DefaultTableModel jmodel;

RecIntegral recIntegral;

int rowNumber;

JThread (double a, double b, double h, DefaultTableModel model, int rowNumber, RecIntegral rec) {

up=a;

a1 = a;

b1 = b;

h1 = h;

jmodel = model;

this.rowNumber = rowNumber;

recIntegral = rec;

}

public Double Integral (double x) {

return 1/x;

}

public void setValues(double a, double b, double h, DefaultTableModel model, int rowNumber, RecIntegral rec) {

up=a;

a1 = a;

b1 = b;

h1 = h;

jmodel = model;

this.rowNumber = rowNumber;

recIntegral = rec;

}

public double getResult() {

return this.sum;

}

@Override

public void run() {

System.out.printf("%s is started...\n", Thread.currentThread().getName());

try {

Thread.sleep(3000);

}

catch (InterruptedException ex) {

System.out.println("Thread was interrupted!\n");

}

double ost=(b1-a1)%h1;

sum = h1\*(0.5\*(Integral(a1)+Integral(a1+ost)));

a1=a1+ost;

while (a1<=b1) {

sum = sum + h1\*(0.5\*(Integral(a1) + Integral(a1+h1)));

a1=a1+h1;

}

recIntegral.setResult(sum);

jmodel.setValueAt(sum, rowNumber, 3);

recIntegral.list.add(new RecIntegral(a1, b1, h1, sum));

//System.out.print("Array Size: ");

//System.out.print(rec.list.size());

System.out.printf("%s was finished!\n", Thread.currentThread().getName());

}

}

**Листинг (Server.java):**

package server;

import java.net.\*;

import java.io.\*;

public class Server extends Thread {

private ServerSocket serverSocket;

private double a;

private double b;

private double h;

private double result;

public static final String ANSI\_GREEN = "\u001B[32m";

public static final String ANSI\_BLUE = "\u001B[34m";

public static final String ANSI\_RED = "\u001B[31m";

public static final String ANSI\_RESET = "\u001B[0m";

public Server(int port) {

try {

serverSocket = new ServerSocket(port);

} catch (IOException ex) {

System.out.println(ex);

}

}

public double Integral(double x) {

return 1/x;

}

@Override

public void run() {

while (true) {

try {

System.out.println("Waiting for connection to port " + ANSI\_BLUE +

serverSocket.getLocalPort() + ANSI\_RESET + "...");

Socket socket = serverSocket.accept();

System.out.println(ANSI\_GREEN + "The client is successfully connected to the server!\n");

DataInputStream in = new DataInputStream(socket.getInputStream());

DataOutputStream out = new DataOutputStream(socket.getOutputStream());

int count = in.readInt();

//System.out.println("Iterations: " + count);

for (int j = 0; j < count; j++) {

a = in.readDouble();

b = in.readDouble();

h = in.readDouble();

System.out.println(ANSI\_RESET + "The server received the following data from the client: " + ANSI\_RED + a + ANSI\_RESET +

", " + ANSI\_GREEN + b + ANSI\_RESET + ", " + ANSI\_BLUE + h);

double ost=(b-a)%h;

double sum = h\*(0.5\*(Integral(a)+Integral(a+ost)));

a=a+ost;

while (a<=b) {

sum = sum + h\*(0.5\*(Integral(a) + Integral(a+h)));

a=a+h;

}

result = sum;

out.writeDouble(result);

System.out.println(ANSI\_RESET + "The server sent the result: " + ANSI\_GREEN + result + ANSI\_RESET + "\n");

}

socket.close();

} catch (IOException ex) {

System.out.println(ex);

}

}

}

public static void main(String[] args) {

int port = 8000;

Thread thread = new Server(port);

thread.start();

}

}

**Результаты программы:**

На рис. 1 изображены данные, которые мы будем отправлять на сервер для их последующей обработки.

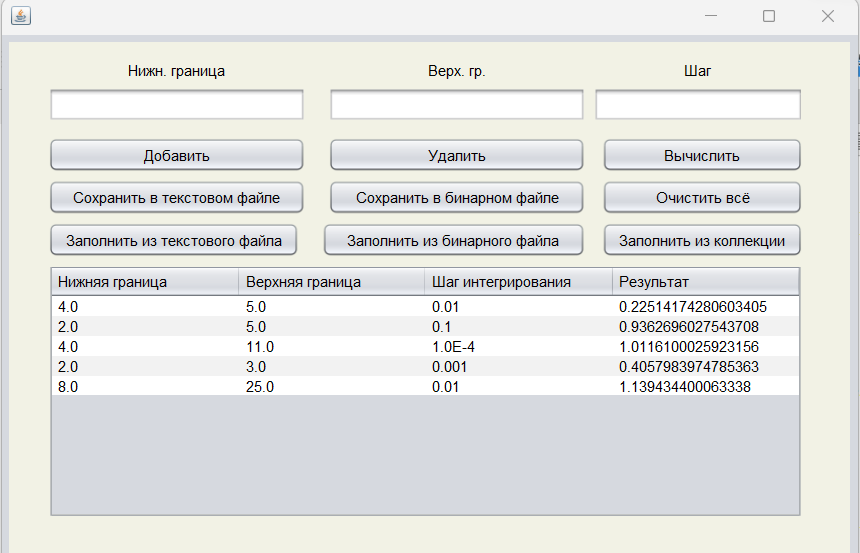


Рисунок 1 – Тестовые данные

При нажатии кнопки “Вычислить” осуществляется отправка данных на сервер, который сначала получает их, вычисляет значение определенного интеграла согласно полученной от клиента информации, а затем отправляет вычисленное значение обратно клиенту. Работа сервера проиллюстрирована на рисунке 2.

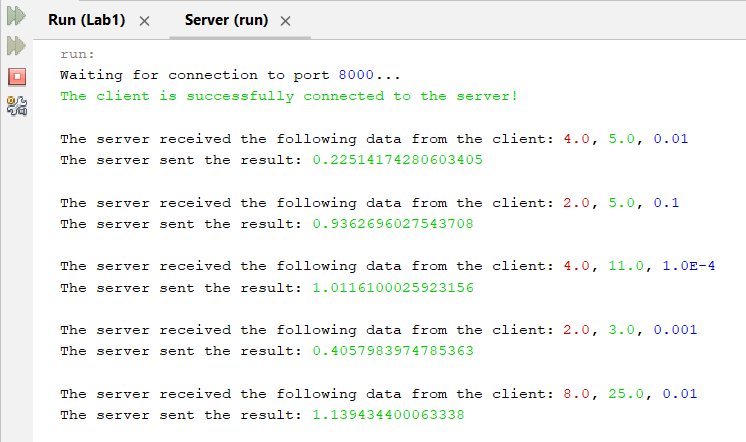


Рисунок 2 – Работа сервера

На рисунке 3 показана работа клиентской части программы в процессе передачи/получения информации.

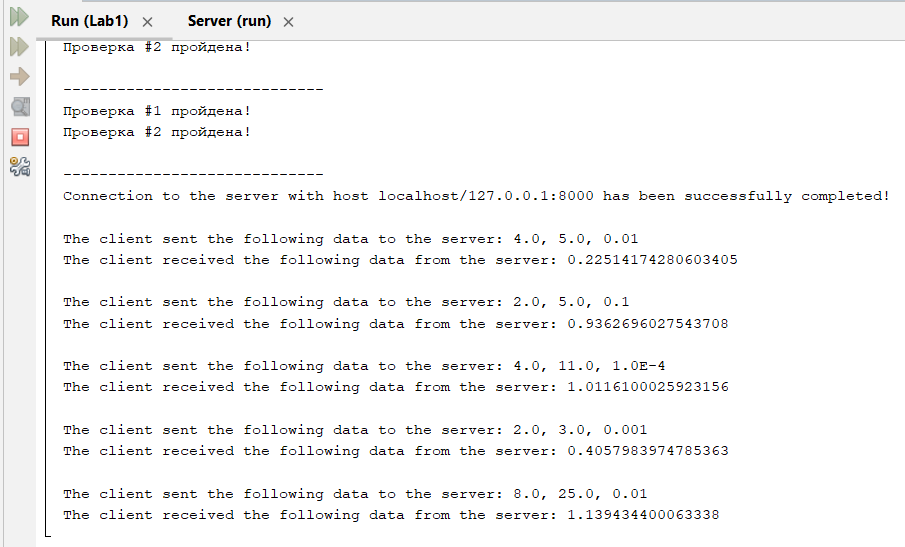


Рисунок 3 – Работа клиента

**Вывод:** научились создавать клиент-серверные приложения c использованием стандартных классов Java.