Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»  
  
  
  
  
  
  
**ОТЧЁТ**

по лабораторной работе № 4

по курсу «Программирование на языке Java»

на тему «Сетевые взаимодействия в Java»

Вариант №7

Выполнили:

Студенты группы 21ВВП2

Морозов К.В.

Самохвалов Я.Д.

Приняли:

К.т.н. доцент Юрова О.В.

К.т.н. доцент Карамышева Н.С.

Пенза 2024

**Название:** Сетевые взаимодействия в Java**.**

**Цель работы:** Научиться создавать клиент-серверное приложения с использованием стандартных классов Java.

**Лабораторное задание:** Модифицировать приложение из предыдущей лабораторной работы, реализовав клиент-серверную архитектуру, обеспечивающую распределённое вычисление определённого 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.

**Вариант:**

|  |  |  |
| --- | --- | --- |
| Номер варианта | Функция | Протокол |
| 7 | https://lh7-us.googleusercontent.com/ePJownX29XlkWquUWhSEBaI8gWuJdkgQ9ue_barwc39oj_1wf665ydc_sskLBBzvPcgm5uaqRbCGCiIQ2nNCbMogzWdwcK2e5Sx0cV_4Lgo3Gni9jHOu7BV9Lmr5YHz8Lfk3goZF1Hn6wCYOhJ3DEA | UDP |

**Ход работы**

**Листинг программы:**

**Server.java**

package my.numberaddition;

import java.io.IOException;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

import java.net.SocketException;

public class ServerLab6 {

public final static int SERVICE\_PORT=50001;

public static void main(String[] args) throws IOException{

JavaLabs6.main(args);

while (true) {

boolean k = JavaLabs6.kk;

while (k == true) {

try{

double a = JavaLabs6.rec.getLowerLimit();

double b = JavaLabs6.rec.getHigherLimit();

double h = JavaLabs6.rec.getIntegralStep();

DatagramSocket ds = new DatagramSocket(SERVICE\_PORT);

byte[] receivingDataBuffer = new byte[1024];

byte[] sendingDataBuffer = new byte[1024];

DatagramPacket inputPacket = new DatagramPacket(receivingDataBuffer, receivingDataBuffer.length);

System.out.println("Waiting for a client to connect...");

ds.receive(inputPacket);

InetAddress sendip = inputPacket.getAddress();

String receivedData = new String(inputPacket.getData());

System.out.println("Sent from the client: "+receivedData);

sendingDataBuffer = receivedData.toUpperCase().getBytes();

InetAddress senderAddress = inputPacket.getAddress();

int senderPort = inputPacket.getPort();

DatagramPacket outputPacket = new DatagramPacket(sendingDataBuffer, sendingDataBuffer.length, senderAddress,senderPort);

ds.send(outputPacket);

byte[] bufout = null;

byte[] buf = null;

DatagramPacket DpReceive = null;

DatagramPacket DpSend = null;

buf = new byte[65535];

String inp = String.valueOf(a);

buf = inp.getBytes();

DpSend = new DatagramPacket(buf, buf.length, sendip, 1234);

ds.send(DpSend);

buf = new byte[65535];

inp = String.valueOf(b);

System.out.println(inp);

buf = inp.getBytes();

DpSend = new DatagramPacket(buf, buf.length, sendip, 1234);

ds.send(DpSend);

buf = new byte[65535];

inp = String.valueOf(h);

System.out.println(inp);

buf = inp.getBytes();

DpSend = new DatagramPacket(buf, buf.length, sendip, 1234);

ds.send(DpSend);

bufout = new byte[65535];

DpReceive = new DatagramPacket(bufout, bufout.length);

ds.receive(DpReceive);

String out = new String(DpReceive.getData());

System.out.println(out);

double result = Double.parseDouble(out);

JavaLabs6.rec.setIntegralResult(result);

ds.close();

} catch (SocketException e){

e.printStackTrace();

}

}

}

}

}

**ClientLab6.java**

package my.numberaddition;

import java.io.IOException;

import java.io.Serializable;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

import java.net.SocketException;

import java.net.UnknownHostException;

import java.nio.charset.StandardCharsets;

import java.util.logging.Level;

import java.util.logging.Logger;

public class ClientLab6 {

public final static int SERVICE\_PORT = 50001;

public static void main(String args[]) throws SocketException, UnknownHostException, IOException {

class RecIntegral1 implements Serializable {

private double higherLimit;

private double lowerLimit;

private double integralStep;

private double integralResult;

public RecIntegral1(double low, double high, double step, double res) {

higherLimit = high;

lowerLimit = low;

integralStep = step;

integralResult = res;

}

public RecIntegral1(double low, double high, double step) {

higherLimit = high;

lowerLimit = low;

integralStep = step;

}

public double getHigherLimit() {

return higherLimit;

}

public double getLowerLimit() {

return lowerLimit;

}

public double getIntegralStep() {

return integralStep;

}

public double getIntegralResult() {

return integralResult;

}

public String[] getDataAsStringArray() {

String[] data = new String[4];

data[0] = String.valueOf(lowerLimit);

data[1] = String.valueOf(higherLimit);

data[2] = String.valueOf(integralStep);

data[3] = String.valueOf(integralResult);

return data;

}

}

class IntegralRun implements Runnable {

RecIntegral1 rec1;

public IntegralRun(RecIntegral1 rec11) {

this.rec1 = rec11;

}

@Override

public void run() {

synchronized(rec1) {

try {

double sum = 0.0;

double x = rec1.lowerLimit;

while (x < rec1.higherLimit) {

double fx1 = 1 / java.lang.Math.log(x);

double fx2 = 1 / java.lang.Math.log(Math.min(x + rec1.integralStep, rec1.higherLimit));

sum += (fx1 + fx2) \* (Math.min(rec1.integralStep, rec1.higherLimit - x) / 2);

x += rec1.integralStep;

}

int decimalPlaces = 4;

sum = Math.round(sum \* Math.pow(10, decimalPlaces)) / Math.pow(10, decimalPlaces);

rec1.integralResult = sum;

Thread.sleep(100);

} catch (InterruptedException ex) {

Logger.getLogger(JavaLabs6.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

}

try{

DatagramSocket ds = new DatagramSocket();

InetAddress ip = InetAddress.getByName("localhost");

byte[] sendingDataBuffer = new byte[1024];

byte[] receivingDataBuffer = new byte[1024];

String sentence = "Hello from UDP client";

sendingDataBuffer = sentence.getBytes();

DatagramPacket sendingPacket = new DatagramPacket(sendingDataBuffer,sendingDataBuffer.length,ip, SERVICE\_PORT);

ds.send(sendingPacket);

DatagramPacket receivingPacket = new DatagramPacket(receivingDataBuffer,receivingDataBuffer.length);

ds.receive(receivingPacket);

String receivedData = new String(receivingPacket.getData());

System.out.println("Sent from the server: "+receivedData);

byte buf[] = null;

byte bufout[] = null;

buf = new byte[65535];

receivingPacket = new DatagramPacket(buf, buf.length);

ds.receive(receivingPacket);

String inp = new String(receivingPacket.getData());

String[] values = inp.split(" ");

double a = Double.parseDouble(values[0]);

double b = Double.parseDouble(values[1]);

double h = Double.parseDouble(values[2]);

RecIntegral1 rec = new RecIntegral1(a, b, h);

for(int i = 1; i < 8; i++) {

Thread t;

t = new Thread(new IntegralRun(rec));

t.setName("Поток " + i);

t.start();

}

bufout = new byte[65535];

String out = Double.toString(rec.getIntegralResult());

bufout = out.getBytes();

DatagramPacket DpSend = new DatagramPacket(bufout, bufout.length, ip, SERVICE\_PORT);

ds.send(DpSend);

} catch(SocketException e) {

e.printStackTrace();

}

}

}

**ClientLab6.java**

package my.numberaddition;

import java.util.ArrayList;

import javax.swing.table.DefaultTableModel;

import java.io.Serializable;

import java.io.\*;

import javax.swing.\*;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

import java.net.SocketException;

import java.util.logging.Level;

import java.util.logging.Logger;

public class JavaLabs6 extends javax.swing.JFrame {

public final static int SERVICE\_PORT=50001;

/\*\*

\* Creates new form JavaLabsUI

\*/

public JavaLabs6() {

initComponents();

}

public class RecIntegral implements Serializable {

private static final long serialVersionUID = 1L;

private double higherLimit;

private double lowerLimit;

private double integralStep;

private double integralResult;

public RecIntegral(double low, double high, double step, double res) throws RecIntegralException {

this.higherLimit = high;

this.lowerLimit = low;

this.integralStep = step;

this.integralResult = res;

}

private RecIntegral(double low, double high, double step) {

this.higherLimit = high;

this.lowerLimit = low;

this.integralStep = step;

}

public double getHigherLimit() {

return higherLimit;

}

public double getLowerLimit() {

return lowerLimit;

}

public double getIntegralStep() {

return integralStep;

}

public double getIntegralResult() {

return integralResult;

}

public void setIntegralResult(double r) {

integralResult = r;

}

public String[] getDataAsStringArray() {

String[] data = new String[4];

data[0] = String.valueOf(lowerLimit);

data[1] = String.valueOf(higherLimit);

data[2] = String.valueOf(integralStep);

data[3] = String.valueOf(integralResult);

return data;

}

}

public class RecIntegralException extends Exception {

public RecIntegralException(String message) {

JOptionPane.showMessageDialog(null, message);

}

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

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

private void initComponents() {

fs = new javax.swing.JFileChooser();

jTextField1 = new javax.swing.JTextField();

jButton1 = new javax.swing.JButton();

jButton2 = new javax.swing.JButton();

jButton3 = new javax.swing.JButton();

jScrollPane1 = new javax.swing.JScrollPane();

jTable1 = new javax.swing.JTable();

jTextField2 = new javax.swing.JTextField();

jTextField3 = new javax.swing.JTextField();

jButton4 = new javax.swing.JButton();

jButton5 = new javax.swing.JButton();

jLabel1 = new javax.swing.JLabel();

jLabel3 = new javax.swing.JLabel();

jLabel4 = new javax.swing.JLabel();

jOptionPane1 = new javax.swing.JOptionPane();

jLabel2 = new javax.swing.JLabel();

jLabel5 = new javax.swing.JLabel();

jButton6 = new javax.swing.JButton();

jButton7 = new javax.swing.JButton();

jLabel6 = new javax.swing.JLabel();

jButton8 = new javax.swing.JButton();

jButton9 = new javax.swing.JButton();

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

setLocation(new java.awt.Point(450, 250));

setResizable(false);

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

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

jTextField1ActionPerformed(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);

}

});

jTable1.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

"Нижняя граница интегрирования", "Верхняя граница интегрирования", "Шаг интегрирования", "Резльтат вычисления"

}

));

jTable1.setCursor(new java.awt.Cursor(java.awt.Cursor.DEFAULT\_CURSOR));

jTable1.setEditingColumn(0);

jTable1.setEditingRow(0);

jTable1.setInheritsPopupMenu(true);

jTable1.setShowGrid(true);

jScrollPane1.setViewportView(jTable1);

jTable1.getColumnModel().getSelectionModel().setSelectionMode(javax.swing.ListSelectionModel.SINGLE\_INTERVAL\_SELECTION);

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);

}

});

jButton4.setText("Очистить");

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

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

jButton4ActionPerformed(evt);

}

});

jButton5.setText("Заполнить");

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

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

jButton5ActionPerformed(evt);

}

});

jLabel1.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N

jLabel1.setText("Нижняя граница интегрирования");

jLabel3.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N

jLabel3.setText("Верхняя граница интегрирования");

jLabel4.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N

jLabel4.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);

jLabel4.setText("Шаг интегрирования");

jLabel2.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N

jLabel2.setText("В текстовом виде");

jLabel5.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N

jLabel5.setText("Сохранение/загрузка в файл/из файла");

jButton6.setText("Сохранить");

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

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

jButton6ActionPerformed(evt);

}

});

jButton7.setText("Загрузить");

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

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

jButton7ActionPerformed(evt);

}

});

jLabel6.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N

jLabel6.setText("В двоичном виде");

jButton8.setText("Сохранить");

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

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

jButton8ActionPerformed(evt);

}

});

jButton9.setText("Загрузить");

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

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

jButton9ActionPerformed(evt);

}

});

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

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

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

.addGroup(layout.createSequentialGroup()

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

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

.addGroup(layout.createSequentialGroup()

.addGap(39, 39, 39)

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

.addComponent(jLabel3)

.addComponent(jLabel1)))

.addGroup(layout.createSequentialGroup()

.addGap(87, 87, 87)

.addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, 100, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createSequentialGroup()

.addGap(68, 68, 68)

.addComponent(jLabel4))

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jButton1)

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

.addComponent(jButton2)

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

.addComponent(jButton3))

.addGroup(layout.createSequentialGroup()

.addGap(86, 86, 86)

.addComponent(jTextField2, javax.swing.GroupLayout.PREFERRED\_SIZE, 100, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createSequentialGroup()

.addGap(86, 86, 86)

.addComponent(jTextField3, javax.swing.GroupLayout.PREFERRED\_SIZE, 100, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createSequentialGroup()

.addGap(40, 40, 40)

.addComponent(jButton4)

.addGap(9, 9, 9)

.addComponent(jButton5))

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

.addContainerGap()

.addComponent(jLabel5))

.addGroup(layout.createSequentialGroup()

.addGap(76, 76, 76)

.addComponent(jLabel2)))

.addGroup(layout.createSequentialGroup()

.addGap(77, 77, 77)

.addComponent(jLabel6))

.addGroup(layout.createSequentialGroup()

.addGap(36, 36, 36)

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

.addGroup(layout.createSequentialGroup()

.addComponent(jButton8)

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

.addComponent(jButton9))

.addGroup(layout.createSequentialGroup()

.addComponent(jButton6)

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

.addComponent(jButton7)))))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 53, Short.MAX\_VALUE)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 598, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap())

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

.addGroup(layout.createSequentialGroup()

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

.addComponent(jOptionPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

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

);

layout.setVerticalGroup(

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

.addGroup(layout.createSequentialGroup()

.addContainerGap()

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

.addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT\_SIZE, 378, Short.MAX\_VALUE)

.addGroup(layout.createSequentialGroup()

.addComponent(jLabel1)

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

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

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

.addComponent(jLabel3)

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

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

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

.addComponent(jLabel4)

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

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

.addGap(18, 18, 18)

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

.addComponent(jButton1)

.addComponent(jButton2)

.addComponent(jButton3))

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

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

.addComponent(jButton4)

.addComponent(jButton5))

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

.addComponent(jLabel5)

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

.addComponent(jLabel2)

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

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

.addComponent(jButton6)

.addComponent(jButton7))

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

.addComponent(jLabel6)

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

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

.addComponent(jButton8)

.addComponent(jButton9))

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

.addContainerGap())

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

.addGroup(layout.createSequentialGroup()

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

.addComponent(jOptionPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

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

);

pack();

}// </editor-fold>

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

// TODO add your handling code here:

try {

a = Double.parseDouble(jTextField1.getText());

b = Double.parseDouble(jTextField2.getText());

h = Double.parseDouble(jTextField3.getText());

model = (javax.swing.table.DefaultTableModel) jTable1.getModel();

if (a < 0.000001 || a > 1000000 || b < 0.000001 || b > 1000000 || h < 0.000001 || h > 1000000) {

throw new RecIntegralException("Values must be between 0.000001 and 1000000");

}

if (a >= b)throw new RecIntegralException("The lowerBound cannot be equal or be greater than the upperBound");

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

createServer();

rec = new RecIntegral(a, b, h, result);

arr.add(rec);

} catch (RecIntegralException ex) {

jOptionPane1.showMessageDialog(null, ex.getMessage(), "Некорректно введенные данные", jOptionPane1.ERROR\_MESSAGE);

}

count += 1;

}

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

// TODO add your handling code here:

}

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

// TODO add your handling code here:

model = (javax.swing.table.DefaultTableModel) jTable1.getModel();

int num = jTable1.getSelectedRow();

model.removeRow(num);

arr.remove(num);

count -= 1;

}

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

// TODO add your handling code here:

model = (DefaultTableModel) jTable1.getModel();

int num = jTable1.getSelectedRow();

a = (double) model.getValueAt(num, 0);

b = (double) model.getValueAt(num, 1);

h = (double) model.getValueAt(num, 2);

result = arr.get(num).getIntegralResult();

model.removeRow(num);

model.insertRow(num, new Object[]{a, b, h, result});

}

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

// TODO add your handling code here:

}

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

// TODO add your handling code here:

}

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

// TODO add your handling code here:

model = (javax.swing.table.DefaultTableModel) jTable1.getModel();

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

model.removeRow(0);

}

count = 0;

}

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

// TODO add your handling code here:

model = (javax.swing.table.DefaultTableModel) jTable1.getModel();

if (count != 0) {

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

model.removeRow(0);

}

count = 0;

}

count = arr.size();

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

model.addRow(new Object[]{arr.get(i).lowerLimit, arr.get(i).higherLimit, arr.get(i).integralStep, arr.get(i).integralResult});

}

}

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

// TODO add your handling code here:

fs.setDialogTitle("Сохранить файл");

fs.addChoosableFileFilter(new javax.swing.filechooser.FileFilter() {

public String getDescription() {

return "TXT Files (\*.txt)";

}

public boolean accept(File f) {

if (f.isDirectory()) {

return true;

} else {

return f.getName().toLowerCase().endsWith(".txt");

}

}

});

int returnVal = fs.showSaveDialog(null);

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

java.io.File file = fs.getSelectedFile();

saveTextToFile(file.getAbsolutePath(), arr);

}

}

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

// TODO add your handling code here:

fs.setDialogTitle("Загрузить файл");

fs.addChoosableFileFilter(new javax.swing.filechooser.FileFilter() {

public String getDescription() {

return "TXT Files (\*.txt)";

}

public boolean accept(File f) {

if (f.isDirectory()) {

return true;

} else {

return f.getName().toLowerCase().endsWith(".txt");

}

}

});

int returnVal = fs.showSaveDialog(null);

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

java.io.File file = fs.getSelectedFile();

loadTextFromFile(file.getAbsolutePath());

}

}

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

// TODO add your handling code here:

fs.setDialogTitle("Сохранить файл");

fs.addChoosableFileFilter(new javax.swing.filechooser.FileFilter() {

public String getDescription() {

return "TXT Files (\*.txt)";

}

public boolean accept(File f) {

if (f.isDirectory()) {

return true;

} else {

return f.getName().toLowerCase().endsWith(".txt");

}

}

});

int returnVal = fs.showSaveDialog(null);

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

java.io.File file = fs.getSelectedFile();

saveBinaryToFile(file.getAbsolutePath(), arr);

}

}

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

// TODO add your handling code here:

fs.setDialogTitle("Загрузить файл");

fs.addChoosableFileFilter(new javax.swing.filechooser.FileFilter() {

public String getDescription() {

return "TXT Files (\*.txt)";

}

public boolean accept(File f) {

if (f.isDirectory()) {

return true;

} else {

return f.getName().toLowerCase().endsWith(".txt");

}

}

});

int returnVal = fs.showSaveDialog(null);

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

java.io.File file = fs.getSelectedFile();

loadBinaryFromFile(file.getAbsolutePath());

}

}

public int n;

public double nnn;

public void createServer() {

try{

DatagramSocket ds = new DatagramSocket(SERVICE\_PORT);

byte[] receivingDataBuffer = new byte[1024];

byte[] sendingDataBuffer = new byte[1024];

DatagramPacket inputPacket = new DatagramPacket(receivingDataBuffer, receivingDataBuffer.length);

System.out.println("Waiting for a client to connect...");

ds.receive(inputPacket);

InetAddress sendip = inputPacket.getAddress();

String receivedData = new String(inputPacket.getData());

System.out.println("Sent from the client: "+receivedData);

sendingDataBuffer = receivedData.toUpperCase().getBytes();

InetAddress senderAddress = inputPacket.getAddress();

int senderPort = inputPacket.getPort();

DatagramPacket outputPacket = new DatagramPacket(sendingDataBuffer, sendingDataBuffer.length, senderAddress,senderPort);

ds.send(outputPacket);

byte[] bufout = null;

byte[] buf = null;

buf = new byte[65535];

String inp = String.valueOf(a);

inp = inp + " ";

inp = inp + String.valueOf(b);

inp = inp + " ";

inp = inp + String.valueOf(h);

buf = inp.getBytes();

outputPacket = new DatagramPacket(buf, buf.length, sendip, senderPort);

ds.send(outputPacket);

bufout = new byte[65535];

inputPacket = new DatagramPacket(bufout, bufout.length);

ds.receive(inputPacket);

String out = new String(inputPacket.getData());

result = Double.parseDouble(out);

ds.close();

} catch (SocketException e){

e.printStackTrace();

} catch (IOException ex) {

Logger.getLogger(JavaLabs6.class.getName()).log(Level.SEVERE, null, ex);

}

}

private void saveTextToFile(String file, ArrayList<RecIntegral> dataToSave) {

try (PrintWriter writer = new PrintWriter(file)) {

for (RecIntegral recIntegral : dataToSave) {

writer.println(recIntegral.getLowerLimit() + " " + recIntegral.getHigherLimit() + " " + recIntegral.getIntegralStep() + " " + recIntegral.getIntegralResult());

}

System.out.println("Data saved to text file.");

} catch (IOException e) {

System.err.println("Error saving to file: " + e.getMessage());

}

}

private void loadTextFromFile(String file) {

try (BufferedReader reader = new BufferedReader(new FileReader(file))) {

String line;

arr.clear();

while ((line = reader.readLine()) != null) {

String[] values = line.split(" ");

double lowerLimit = Double.parseDouble(values[0]);

double higherLimit = Double.parseDouble(values[1]);

double integralStep = Double.parseDouble(values[2]);

double integralResult = Double.parseDouble(values[3]);

try {

RecIntegral recIntegral = new RecIntegral(lowerLimit, higherLimit, integralStep, integralResult);

arr.add(recIntegral);

} catch (RecIntegralException e) {

System.out.println("Invalid values: " + e.getMessage());

}

}

model = (javax.swing.table.DefaultTableModel) jTable1.getModel();

if (count != 0) {

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

model.removeRow(0);

}

count = 0;

}

count = arr.size();

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

model.addRow(new Object[]{arr.get(i).lowerLimit, arr.get(i).higherLimit, arr.get(i).integralStep, arr.get(i).integralResult});

}

System.out.println("Data loaded from text file.");

} catch (IOException e) {

System.err.println("Error loading from file: " + e.getMessage());

}

}

private void saveBinaryToFile(String file, ArrayList<RecIntegral> dataToSave) {

try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream(file))) {

oos.writeObject(dataToSave);

oos.close();

System.out.println("Data saved to binary file.");

} catch (IOException e) {

System.err.println("Error saving to file: " + e.getMessage());

}

}

private void loadBinaryFromFile(String file) {

arr.clear();

try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(file))) {

ArrayList<RecIntegral> loadedData = (ArrayList<RecIntegral>) ois.readObject();

arr.addAll(loadedData);

ois.close();

model = (javax.swing.table.DefaultTableModel) jTable1.getModel();

if (count != 0) {

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

model.removeRow(0);

}

count = 0;

}

count = arr.size();

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

model.addRow(new Object[]{arr.get(i).lowerLimit, arr.get(i).higherLimit, arr.get(i).integralStep, arr.get(i).integralResult});

}

System.out.println("Data loaded from binary file.");

} catch (IOException | ClassNotFoundException e) {

System.err.println("Error loading from file: " + e.getMessage());

}

}

/\*\*

\* @param args the command line arguments

\*/

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(JavaLabs6.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(JavaLabs6.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(JavaLabs6.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(JavaLabs6.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

/\* Create and display the form \*/

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

public void run() {

new JavaLabs6().setVisible(true);

}

});

}

public double a;

public double b;

public double h;

public double result;

public int count = 0;

public javax.swing.table.DefaultTableModel model;

public ArrayList<RecIntegral> arr = new ArrayList<>();

public static RecIntegral rec;

public static boolean kk = false;

// Variables declaration - do not modify

private javax.swing.JFileChooser fs;

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.JLabel jLabel4;

private javax.swing.JLabel jLabel5;

private javax.swing.JLabel jLabel6;

private javax.swing.JOptionPane jOptionPane1;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JTable jTable1;

private javax.swing.JTextField jTextField1;

private javax.swing.JTextField jTextField2;

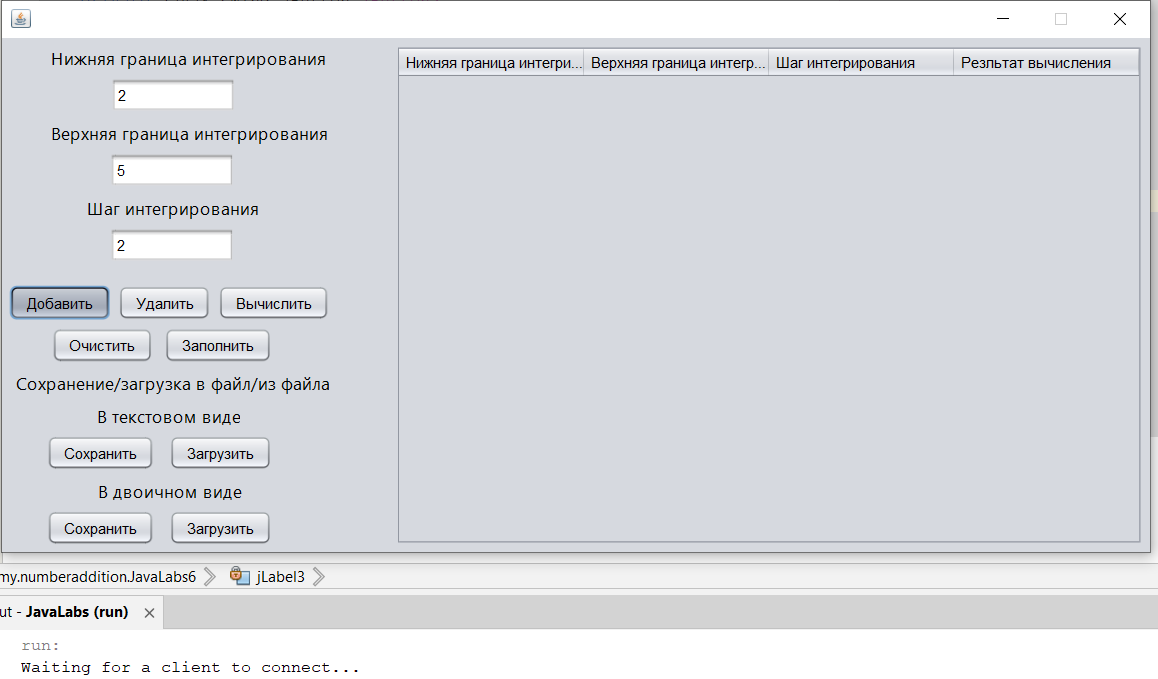
private javax.swing.JTextField jTextField3;

// End of variables declaration

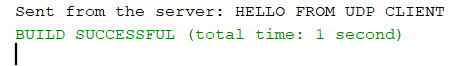
}

**Результат программы:**

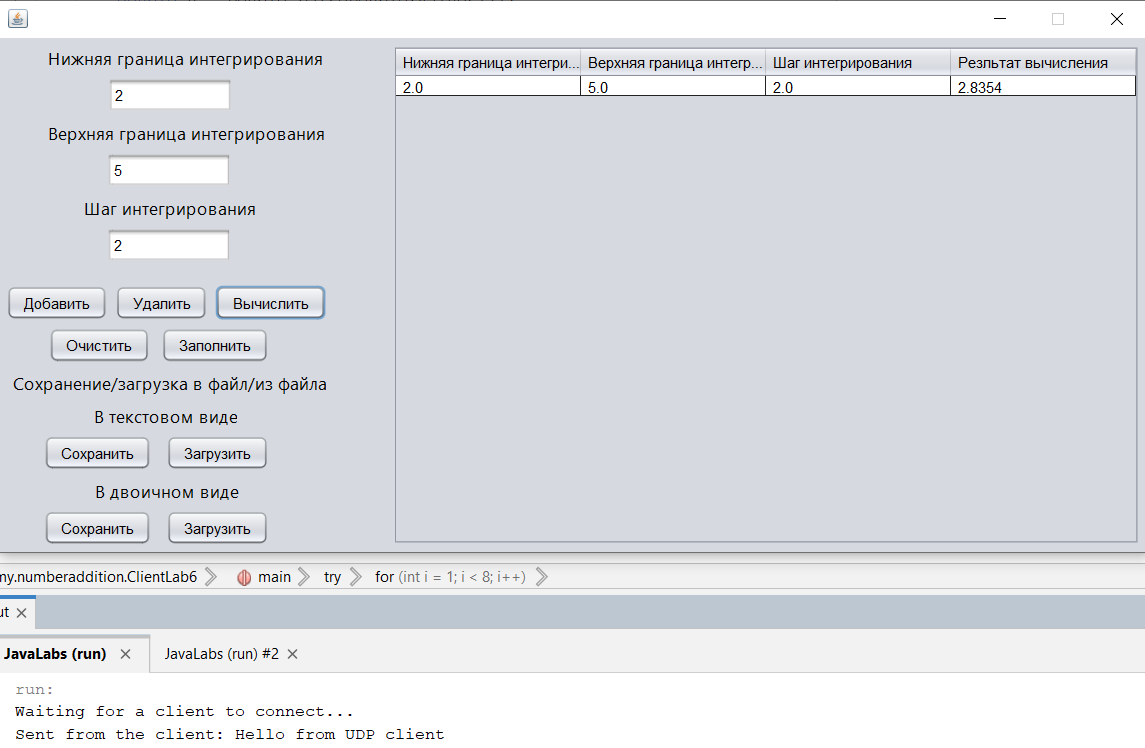
Ожидание подключения клиента:

****

Вход клиента на сервер:

****

Результат после входа клиента на сервер:

****

**Вывод:** В ходе выполнения лабораторной работы было создано клиент-серверное приложение с использованием стандартных классов Java.