Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**по курсу «Программирование на языке JAVA»

по лабораторной работе №6

на тему: «Сетевое взаимодействие в Java»  
Вариант 10.

Выполнил студент группы 20ВВП1:

Кодиров И. Н.

Приняла:

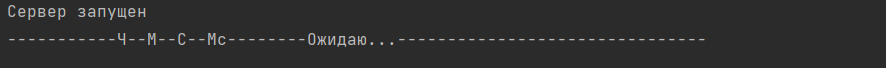
Юрова О. В.

Пенза 2023

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

**Задание на лабораторную работу:** 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.

**Результат работы программы:**

**Листинг:**

import java.io.IOException;  
import java.net.DatagramPacket;  
import java.net.DatagramSocket;  
import java.net.InetAddress;  
import java.text.SimpleDateFormat;  
import java.util.ArrayList;  
import java.util.Date;  
  
public class Server {  
 private String[] stlArray;  
 ArrayList<Double> dbArrayd = new ArrayList<Double>();  
 double dataD;  
 public Server() {  
 }  
 public void runi() throws IOException, InterruptedException {  
 ReceiveRequest();  
 Date date = new Date();  
 SimpleDateFormat formatter = new SimpleDateFormat("dd-MM-yyyy HH:mm:ss:SS");  
 System.out.println(formatter.format(date)+": Данные приняты");  
 dbArrayd.clear();  
 dbArrayd.add(Double.valueOf(stlArray[0]));  
 dbArrayd.add(Double.valueOf(stlArray[1]));  
 dbArrayd.add(Double.valueOf(stlArray[2]));  
 Trap();  
  
 Date date2 = new Date();  
 SimpleDateFormat formatter2 = new SimpleDateFormat("dd-MM-yyyy HH:mm:ss:SS");  
 System.out.println(formatter.format(date2)+": Интеграл вычислен");  
  
 SendResponse(dataD);  
 Date date1 = new Date();  
 SimpleDateFormat formatter1 = new SimpleDateFormat("dd-MM-yyyy HH:mm:ss:SS");  
 System.out.println(formatter1.format(date1)+": Данные отправлены");  
 }  
 public void ReceiveRequest() throws IOException, IOException {  
 DatagramSocket socket = new DatagramSocket(12345);  
 byte[] buffer = new byte[1024];  
 DatagramPacket packet = new DatagramPacket(buffer, buffer.length);  
 socket.receive(packet);  
 String message = new String(packet.getData(), 0, packet.getLength());  
 String requestData = new String(message);  
 socket.close();  
 stlArray = requestData.split(",");  
 }  
  
 public void SendResponse(double responseData) throws IOException {  
 DatagramSocket socket = new DatagramSocket();  
 byte[] data = String.valueOf(responseData).getBytes();  
 InetAddress clientAddress = InetAddress.getByName("127.0.0.1");  
 int clientPort = 12346;  
 DatagramPacket packet = new DatagramPacket(data, data.length, clientAddress, clientPort);  
 socket.send(packet);  
 socket.close();  
 }  
 static double InFunction(double x) //Подынтегральная функция  
 {  
 return 1/(Math.log(x));  
 }  
 public void Trap() throws InterruptedException {  
 final double[] result = {0};  
 int n = (int)((dbArrayd.get(0)-dbArrayd.get(2) - dbArrayd.get(1)) / dbArrayd.get(2));  
 result[0] += (InFunction(dbArrayd.get(0)) + InFunction(dbArrayd.get(1))) / 2;  
 int chunkSize = n / 7; // Размер частей  
 Thread[] threads = new Thread[7];  
 for (int i = 0; i < 7; i++) {  
  
 int startIndex = i \* chunkSize +1;  
 int endIndex = (i +1) \* chunkSize;  
  
 if (i == 6) {  
 endIndex = n;  
 }  
 int finalEndIndex = endIndex;  
 Runnable task = new Runnable() {  
 public void run() {  
 double localResult = 0;  
 for (int j = startIndex; j <= finalEndIndex; j++) {  
 localResult += InFunction(dbArrayd.get(1) + dbArrayd.get(2) \* j);  
 }  
  
 synchronized(this) {  
 result[0] += localResult;  
 }  
 }  
 };  
 threads[i] = new Thread(task);  
 threads[i].start();  
 // threads[i].join();  
 }  
 for (Thread thread : threads) {  
 try {  
 thread.join();  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 dataD=dbArrayd.get(2) \* result[0];  
 }  
  
}

main

import java.io.IOException;  
  
public class Main {  
  
 public Main() {  
 }  
 public static void main(String[] args) throws IOException, InterruptedException {  
 Server server = new Server();  
 System.out.println("Сервер запущен");  
  
 while (true) {  
 System.out.println("-----------Ч--М--С--Мс--------Ожидаю...-------------------------------");  
 server.runi();  
 }  
 }  
}

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