МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Факультет информационных технологий и робототехники (ФИТР)

Кафедра программного обеспечения информационных систем и технологий

**О Т Ч Е Т**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ №2**

# «Разработка веб-приложений»

**по курсу:**

***"Разработка веб-приложений"***

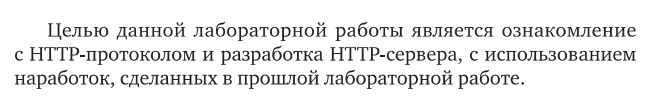
**Вариант 8**

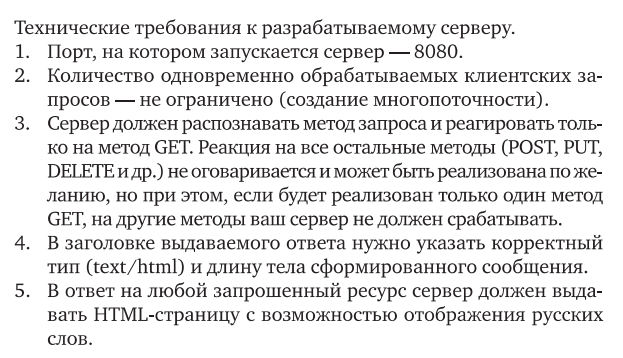
|  |  |
| --- | --- |
| Выполнили: | Студент: Журович К.С.  Гр. 10701121 |
| Проверил: | Пр. Кондратенок Е.В. |

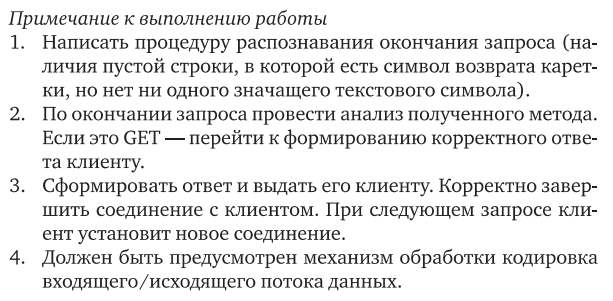
Минск – 2024

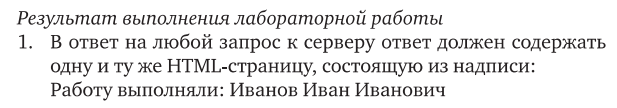
**ЛАБОРАТОРНАЯ РАБОТА 2**

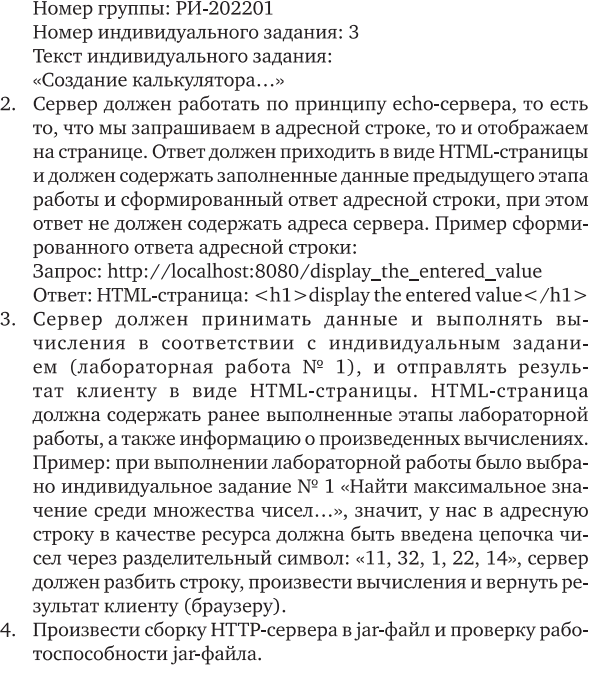
***Цель лабораторной работы:***











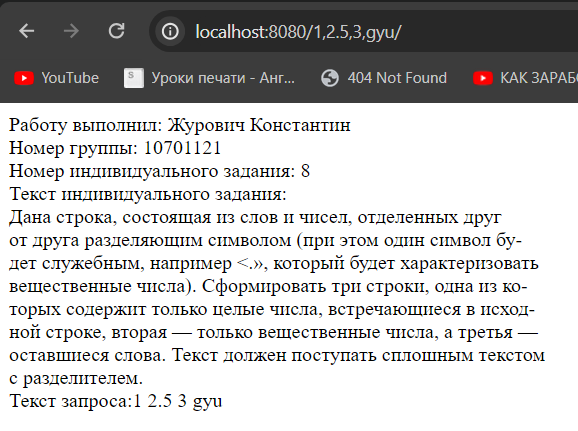


Рисунок 1 – Задание 1,2

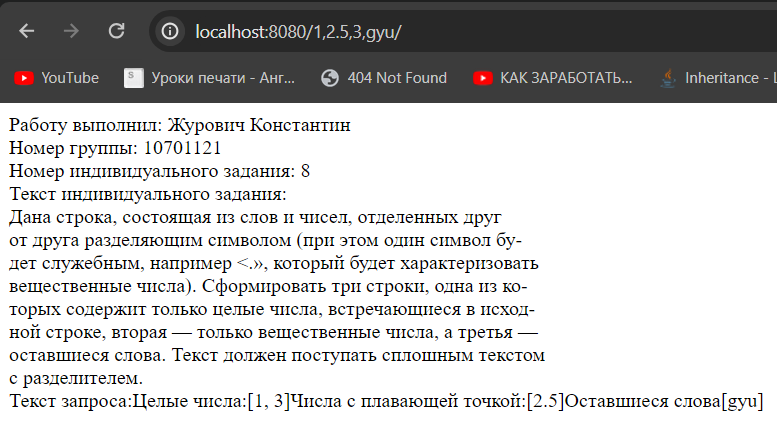


Рисунок 2 – Задание 3

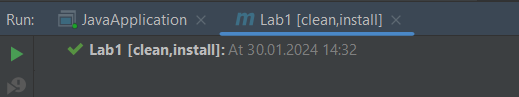


Рисунок 2 – Задание 4

**Листинг**

public class JavaApplication {  
 public static void main(String[] args) {  
 try (ServerSocket serverSocket = new ServerSocket(8080)){  
 while (true) {  
 Socket socket = serverSocket.accept();  
 System.*out*.println("Client connected");  
 new Thread(() -> *handleClient*(socket)).start();  
 }  
 } catch (IOException e) {  
 throw new RuntimeException(e);  
 }  
 }  
  
 private static void handleClient(Socket clientSocket) {  
 try (  
 BufferedReader reader = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));  
 OutputStream outputStream = clientSocket.getOutputStream()  
 ) {  
 // Чтение HTTP-запроса  
 StringBuilder request = new StringBuilder();  
 String line;  
 while ((line = reader.readLine()) != null && !line.isEmpty()) {  
 request.append(line).append("\r\n");  
 }  
  
 // Регулярное выражение для поиска строки после "GET /" и перед " HTTP/1.1"  
 Pattern pattern = Pattern.*compile*("GET /(.+?)/ HTTP/1\\.1");  
 Matcher matcher = pattern.matcher(request.toString());  
 String extractedString = new String("");  
 // Поиск соответствия  
 if (matcher.find()) {  
 // Извлечение найденной строки  
 extractedString = matcher.group(1);  
 }  
 Result result = new Result(extractedString);  
 String param = result.getResult();  
  
 // Распознавание метода запроса  
 if (request.toString().startsWith("GET")) {  
 String htmlContent = "<html>" +  
 "<body>" +  
 "<div>Работу выполнил: Журович Константин</div>" +  
 "<div>Номер группы: 10701121</div>" +  
 "<div>Номер индивидуального задания: 8</div>" +  
 "<div>Текст индивидуального задания:" + "<br>" +  
 "Дана строка, состоящая из слов и чисел, отделенных друг" + "<br>" +  
 "от друга разделяющим символом (при этом один символ бу-" + "<br>" +  
 "дет служебным, например <.», который будет характеризовать" + "<br>" +  
 "вещественные числа). Сформировать три строки, одна из ко-" + "<br>" +  
 "торых содержит только целые числа, встречающиеся в исход-" + "<br>" +  
 "ной строке, вторая — только вещественные числа, а третья —" + "<br>" +  
 "оставшиеся слова. Текст должен поступать сплошным текстом" + "<br>" +  
 "с разделителем.</div>" +  
 "<div>Текст запроса:" + param + "</div>" +  
 "</body>" +  
 "</html>";  
  
 int contentLength = htmlContent.getBytes("UTF-8").length;  
 // Формирование HTTP-ответа  
 String response = "HTTP/1.1 200 OK\r\nContent-Type: text/html; charset=UTF-8\r\nContent-Length: " +  
 contentLength + "\r\n\r\n" +  
 htmlContent;  
  
 // Отправка ответа клиенту  
 outputStream.write(response.getBytes());  
 outputStream.flush();  
 } else {  
 // В случае других методов, можно вернуть пустой ответ или что-то еще  
 String response = "HTTP/1.1 200 OK\r\nContent-Type: text/html\r\nContent-Length: 0\r\n\r\n";  
 outputStream.write(response.getBytes());  
 outputStream.flush();  
 }  
  
 // Завершение соединения с клиентом  
 clientSocket.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
}

public class Result implements ResultIMPL {  
 private List<Integer> intNumbers;  
 private List<Double> doubleNumbers;  
 private List<String> words;  
  
 public Result(String inputString) {  
  
 // Разделение строки по ","  
 String[] items = inputString.split(",");  
  
 this.intNumbers = new ArrayList<>();  
 this.doubleNumbers = new ArrayList<>();  
 this.words = new ArrayList<>();  
  
 for (String item : items) {  
 try {  
 intNumbers.add(Integer.*parseInt*(item));  
 } catch (NumberFormatException eInt) {  
 try {  
 doubleNumbers.add(Double.*parseDouble*(item));  
 } catch (NumberFormatException eDouble) {  
 words.add(item);  
 }  
 }  
 }  
 }  
 @Override  
 public String getResult() {  
 return "Целые числа:" + intNumbers.toString() +  
 "Числа с плавающей точкой:" + doubleNumbers +  
 "Оставшиеся слова" + words.toString();  
 }  
}

public interface ResultIMPL {  
 String getResult();  
}