



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Отчёт

по лабораторной работе № 2

Название: Формат JSON. Создание сервера.

Дисциплина: Архитектура ЭВМ

Студент

ИУ7-55Б

(Группа)

Д.О. Склифасовский

(Подпись, дата)

(И.О. Фамилия)

Преподаватель

А.Ю. Попов

(Подпись, дата)

(И.О. Фамилия)

Москва, 2020

Содержание

Введение	4
1 Задание 3.1	5
1.1 Условие	5
1.2 Решение	5
1.3 Тесты	6
2 Задание 3.2	7
2.1 Условие	7
2.2 Решение	7
2.3 Тесты	8
3 Задание 3.3	9
3.1 Условие	9
3.2 Решение	9
3.3 Тесты	10
4 Задание 3.4	11
4.1 Условие	11
4.2 Решение	11
4.3 Тесты	12
5 Задание 3.5	13
5.1 Условие	13
5.2 Решение	13
5.3 Тесты	14
6 Задание 3.6	15
6.1 Условие	15

6.2	Решение	15
6.3	Тесты	15
7	Задание 3.7	16
7.1	Условие	16
7.2	Решение	16
7.3	Тесты	17
8	Задание 4.1	19
8.1	Условие	19
8.2	Решение	19
8.3	Тесты	20
9	Задание 4.2	21
9.1	Условие	21
9.2	Решение	21
9.3	Тесты	22
10	Задание 4.3	23
10.1	Условие	23
10.2	Решение	23
10.3	Тесты	25
11	Задание 4.4	26
11.1	Условие	26
11.2	Решение	26
11.3	Тесты	27
	Вывод	27

Введение

Цель работы: познакомиться с форматом JSON. Запустить сервер, поработать с формами и HTML страницами.

1 Задание 3.1

1.1 Условие

С клавиатуры считывается число N. Далее считывается N строк. Необходимо создать массив и сохранять в него строки только с четной длиной. Получившийся массив необходимо преобразовать в строку JSON и сохранить в файл.

1.2 Решение

Листинг 1 – Task 3.1

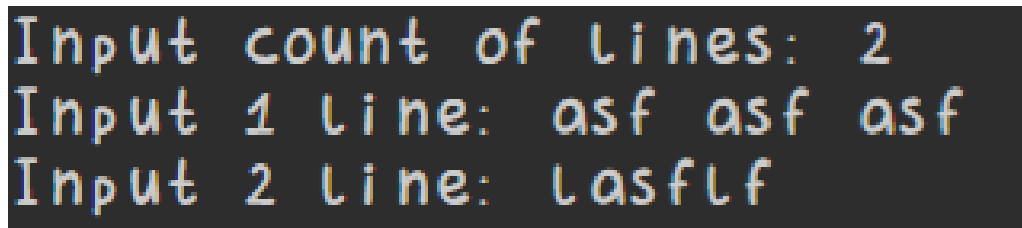
```
1  "use strict";
2
3  const readlineSync = require('readline-sync');
4  const fs = require("fs");
5  const fileName = "./files/firstTask.txt";
6
7  let arr = [];
8  let countOfLines = readlineSync.question("Input count of lines: ");
9  let curLine = "";
10 for (let i = 0; i < parseInt(countOfLines); i++) {
11     curLine = readlineSync.question("Input " + (i + 1) + " line: ");
12     if (curLine.length % 2 == 0) {
13         arr.push(curLine);
14     }
15 }
16 let jsonString = JSON.stringify(arr);
17 console.log(jsonString);
18 fs.writeFileSync(fileName, jsonString);
```

1.3 Тесты

Проводится эксперимент: вводится 2 строки:

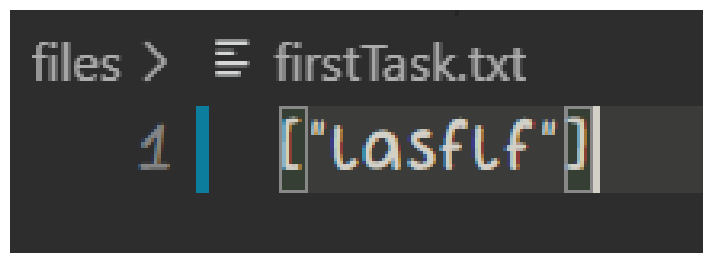
1. "asf asf asf" длиной 11
2. "lasflf" длиной 6

В файл должна сохраниться вторая строка.



```
Input count of lines: 2
Input 1 line: asf asf asf
Input 2 line: lasflf
```

Рисунок 1 – Ввод данных



```
files > ≡ firstTask.txt
1 | ["lasflf"]
```

Рисунок 2 – Результат записи файла

2 Задание 3.2

2.1 Условие

Необходимо считать содержимое файла, в котором хранится массив строк в формате JSON. Нужно вывести только те строки на экран, в которых содержатся только гласные буквы.

2.2 Решение

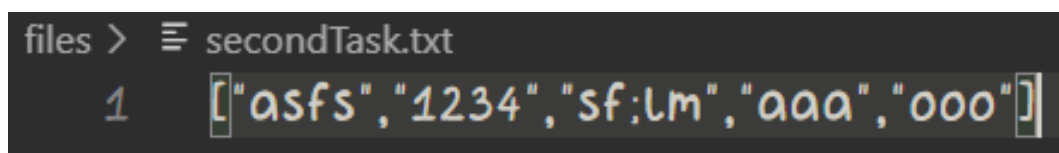
Листинг 2 – Task 3.2

```
1  "use strict";
2
3  function checkVowels(curString) {
4      let arrOfVowels = "AEIOUaeiou";
5      let check = 0;
6      for (let i = 0; i < curString.length; i++) {
7
8          for (let j = 0; j < arrOfVowels.length; j++) {
9              if (curString[i] == arrOfVowels[j]) {
10                 check = 1;
11                 break;
12             }
13         }
14         if (!check) {
15             return 0;
16         }
17         check = 0;
18     }
19     return 1;
20 }
21
22 const fs = require("fs");
23 const fileName = "./files/secondTask.txt";
24
25 const contentString = fs.readFileSync(fileName, "utf8");
26 let arr = JSON.parse(contentString);
27 for (let i = 0; i < arr.length; i++) {
28     if (checkVowels(arr[i])) {
```

```
29     console.log(arr[i]);  
30 }  
31 }
```

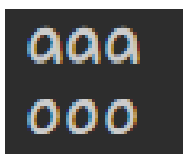
2.3 Тесты

Проводится эксперимент: в файле содержится массив строк:
["asfs "1234 "sf;lm "aaa "ooo"].



```
files > secondTask.txt  
1 ["asfs", "1234", "sf;lm", "aaa", "ooo"]
```

Рисунок 3 – Содержимое файла



```
aaa  
ooo
```

Рисунок 4 – Результат проверки

3 Задание 3.3

3.1 Условие

С клавиатуры считывается строка - название расширения файлов. Далее считывается строка - адрес папки. Необходимо перебрать все файлы в папке и вывести содержимое файлов, у которых расширение совпадает с введенным расширением.

3.2 Решение

Листинг 3 – Task 3.3

```
1  "use strict";
2
3  const fs = require("fs");
4  const readlineSync = require('readline-sync');
5
6  const expansion = readlineSync.question("Input expansion (without point)
   : ");
7  const folder = readlineSync.question("Input folder: ");
8  const arrOfFiles = fs.readdirSync(folder);
9
10 let curFile, curPath;
11 for (let i = 0; i < arrOfFiles.length; i++) {
12   curFile = arrOfFiles[i].split(".");
13   if (curFile[curFile.length - 1] == expansion) {
14     curPath = folder + arrOfFiles[i];
15     console.log(fs.readFileSync(curPath, "utf8"));
16   }
17 }
```

3.3 Тесты

Проводится эксперимент: вводится расширение "txt"и директорию
"./files/"

```
Input expansion (without point): txt
Input folder: ./files/
["\asf\lf"]
["asfs", "1234", "sf;lm", "aaa", "ooo"]
```

Рисунок 5 – Результат работы программы

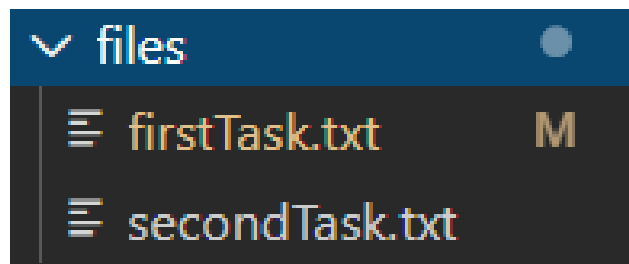


Рисунок 6 – Содержимое директории ./files/

4 Задание 3.4

4.1 Условие

Дана вложенная структура файлов и папок. Все файлы имеют расширение "txt". Необходимо рекурсивно перебрать вложенную структуру и вывести имена файлов, у которых содержимое не превышает по длине 10 символов.

4.2 Решение

Листинг 4 – Task 3.4

```
1  "use strict;"
2
3  const fs = require("fs");
4
5  function checkFolder(folder) {
6    fs.readdir(folder, (err, arr) => {
7      console.log("Folder: " + folder);
8      for(let i = 0; i < arr.length; i++) {
9        var line = arr[i].split(".");
10       if (line[line.length - 1] == "txt") {
11         const contentStr = fs.readFileSync(folder + "/" + arr[i], "utf8"
12           );
13         if (contentStr.length <= 10) {
14           console.log("Name of file: " + arr[i]);
15         }
16       }
17       else {
18         checkFolder(folder + "/" + arr[i]);
19       }
20     });
21     return;
22   }
23
24   var folder = "./fourthTask";
25   checkFolder(folder);
```

4.3 Тесты

Дана вложенная структура файлов и папок.

```
Folder: ./fourthTask
Name of file: 2.txt
Name of file: 3.txt
Folder: ./fourthTask/1
Name of file: 1.txt
Folder: ./fourthTask/1/2
Name of file: 1.txt
```

Рисунок 7 – Результат работы программы

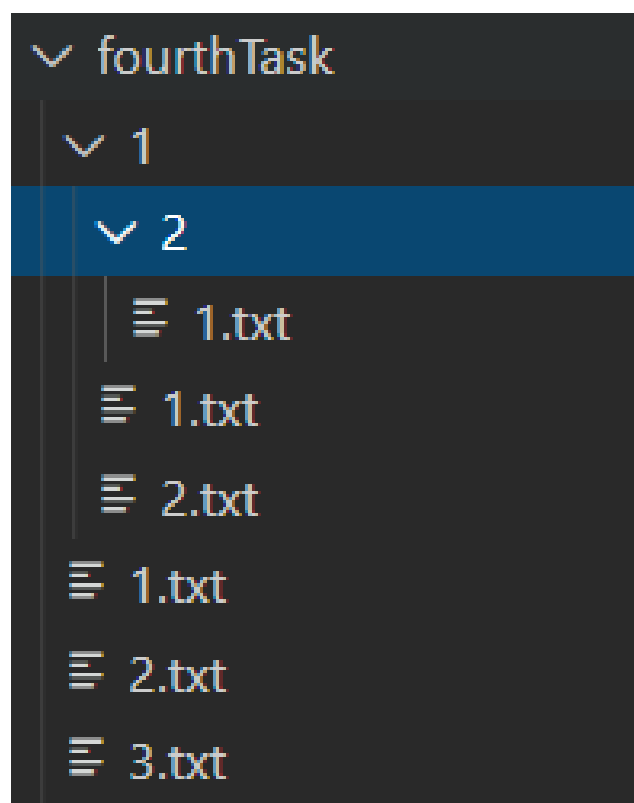


Рисунок 8 – Вложенная структура файлов и папок

5 Задание 3.5

5.1 Условие

С клавиатуры считывается число N. Далее считывается N строк - имена текстовых файлов. Необходимо склеить всё содержимое введенных файлов в одну большую строку и сохранить в новый файл.

5.2 Решение

Листинг 5 – Task 3.5

```
1  "use strict";
2
3  const fs = require("fs");
4  const readlineSync = require('readline-sync');
5  let folder = "./fifthTask/";
6  const count = readlineSync.question("Input count of files: ");
7  let mainLine = ""
8  for (let _ = 0; _ < count; _++) {
9    let check = 0;
10   while (!check) {
11     const file = readlineSync.question("Input name of file " + (_ + 1) +
12       ": ");
13     if (fs.existsSync(folder + file)) {
14       let content = fs.readFileSync(folder + file, "utf8");
15       mainLine += content + "\n";
16       check = 1;
17     } else {
18       console.log("File was not found");
19     }
20   }
21 }
22 fs.writeFileSync(folder + "TaskFive.txt", mainLine);
```

5.3 Тесты

Проводится эксперимент: вводятся названия файлов: "1.txt" "2.txt" и "3.txt".

```
Input count of files: 3
Input name of file 1: 1.txt
Input name of file 2: 2.txt
Input name of file 3: 3.txt
```

Рисунок 9 – Ввод названий файлов

fifthTask > ≡ 1.txt	fifthTask > ≡ 2.txt	fifthTask > ≡ 3.txt
1 1 1 1	1 2 2 2 2	1 3 3 3 3

Рисунок 10 – Содержимое файлов

```
fifthTask > ≡ TaskFive.txt
1 1 1 1 1
2 2 2 2 2
3 3 3 3 3
```

Рисунок 11 – Результат

6 Задание 3.6

6.1 Условие

Написать код, который позволяет определить максимальный возможный уровень вложенности друг в друга полей в объекте, чтобы данный объект можно было преобразовать в строку формата JSON. Ответом является целое число.

6.2 Решение

Листинг 6 – Task 3.6

```
1  "use strict";
2
3  let mainObj = {};
4  let maxLevel = 0;
5  let check = true;
6  let curObj = mainObj;
7  while (check) {
8      curObj.newObj = {};
9      maxLevel++;
10     try {
11         const jsonStr = JSON.stringify(mainObj);
12         curObj = curObj.newObj;
13     } catch (error) {
14         check = false;
15     }
16 }
17
18 console.log(maxLevel);
```

6.3 Тесты

1846

Рисунок 12 – Результат

7 Задание 3.7

7.1 Условие

Из файла считывается строка в формате JSON. В этой строке информация об объекте, в котором находится большое количество вложенных друг в друга полей. Объект представляет из себя дерево. Необходимо рекурсивно обработать дерево и найти максимальную вложенность в дереве. Необходимо вывести на экран ветку с максимальной вложенностью.

7.2 Решение

Листинг 7 – Task 3.7

```
1  const fs = require("fs");
2  let arrCheck = [];
3  let maxArr = [];
4  function checkNesting(obj) {
5      let maxLevel = 1;
6      for (let key in obj) {
7          if (typeof (obj[key]) == "object") {
8              arrCheck.push(key);
9              let lvl = checkNesting(obj[key]) + 1;
10             if (lvl > maxLevel) {
11                 maxLevel = lvl;
12                 if (arrCheck.length > maxArr.length) {
13                     maxArr = arrCheck;
14                     arrCheck = [];
15                 }
16             }
17         }
18     }
19     return maxLevel;
20 }
21
22 const jsonStr = fs.readFileSync("./seventhTask/1.txt");
23 const obj = JSON.parse(jsonStr);
24 console.log(checkNesting(obj));
25 for (let key in maxArr) {
```



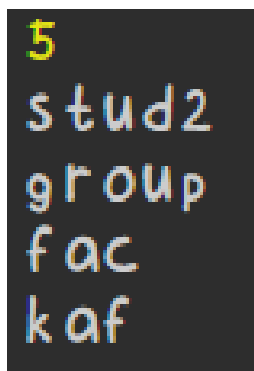
```
26     console.log(maxArr[key]);
27 }
```

7.3 Тесты

```
seventhTask > 1.txt
1  {
2      "stud1": {
3          "name": "George",
4          "age": 20,
5          "group": {
6              "group": 55,
7              "fac": {
8                  "fac": 7
9              }
10         }
11     },
12     "stud2": {
13         "name": "Danya",
14         "age": 19,
15         "group": {
16             "group1": 54,
17             "fac": {
18                 "fac1": "UY",
19                 "kaf": {
20                     "kaf1": 7
21                 }
22             }
23         }
24     },
25     "stud3": {
26         "name": "Denis",
27         "age": 20
28     },
29     "stud4": {
30         "name": "Sanya",
31         "age": 20
32     }
33 }
```

Рисунок 13 – Содержимое файла

Результат:



```
5  
stud2  
group  
fac  
kaf
```

Рисунок 14 – Результат

8 Задание 4.1

8.1 Условие

Запустить сервер. Реализовать на сервере функцию для сравнения трёх чисел и выдачи наибольшего из них. Реализовать страницу с формой ввода для отправки запроса на сервер.

8.2 Решение

Листинг 8 – Файл task1.js

```
1  "use strict";
2
3  const fs = require("fs");
4
5  const express = require("express");
6
7  const app = express();
8  const port = 5015;
9  app.listen(port);
10 console.log("My server on port " + port);
11
12 app.get("/me/page", function(request, response) {
13     const nameString = request.query.p;
14     if (fs.existsSync(nameString)) {
15         const contentString = fs.readFileSync(nameString, "utf8");
16         response.end(contentString);
17     } else {
18         const contentString = fs.readFileSync("bad.html", "utf8");
19         response.end(contentString);
20     }
21 });
22
23 app.get("/calculate/sum", function(request, response) {
24     const a = parseInt(request.query.a);
25     const b = parseInt(request.query.b);
26     const c = parseInt(request.query.c);
27     const answerJSON = JSON.stringify({result: Math.max(a, b, c)});
28     response.end(answerJSON);
```

8.3 Тесты

Task 1

Введите A

Введите B

Введите C

Отправить запрос

Рисунок 15 – Страница

```
{"result":5}
```

Рисунок 16 – Результат

9 Задание 4.2

9.1 Условие

Запустить сервер. На стороне сервера должен храниться файл, внутри которого находится JSON строка. В этой JSON строке хранится информация о массиве объектов. Реализовать на сервере функцию, которая принимает индекс и выдает содержимое ячейки массива по данному индексу. Реализовать страницу с формой ввода для отправки запроса на сервер.

9.2 Решение

Листинг 9 – Файл task2.js

```
1  const fs = require("fs");
2  const express = require("express");
3
4  const app = express();
5  const port = 5015;
6  app.listen(port);
7  console.log("My server on port " + port);
8
9  app.get("/me/page", function(request, response) {
10     const nameString = request.query.p;
11     if (fs.existsSync(nameString)) {
12         const contentString = fs.readFileSync(nameString, "utf8");
13         response.end(contentString);
14     } else {
15         const contentString = fs.readFileSync("bad.html", "utf8");
16         response.end(contentString);
17     }
18 });
19
20 app.get("/task2/index", function(request, response) {
21     let i = parseInt(request.query.i);
22     let groups = JSON.parse(fs.readFileSync("task2.txt", "utf8"));
23     if (i >= 0 && i < groups.length) {
24         const answerJSON = JSON.stringify({result: groups[i]});
25         response.end(answerJSON);
```

```
26     } else {  
27         const contentString = fs.readFileSync("bad.html", "utf8");  
28         response.end(contentString);  
29     }  
30 });
```

9.3 Тесты

Task 2

Введите индекс

Отправить запрос

Рисунок 17 – Страница

```
task2.txt  
1 [{"num":51,"count":24},{"num":52,"count":23},{"num":53,"count":29},{"num":54,"count":14},{"num":55,"count":28}]
```

Рисунок 18 – Содержимое файла на стороне сервера

```
{"result":{"num":52,"count":23}}
```

Рисунок 19 – Результат

10 Задание 4.3

10.1 Условие

Написать программу, которая на вход получает массив названий полей и адрес запроса (куда отправлять). Программа должна генерировать HTML разметку страницы, в которую встроена форма для отправки запроса.

10.2 Решение

Листинг 10 – Файл task3.js

```
1  const fs = require("fs");
2  const express = require("express");
3  const htmlStart = "<!DOCTYPE html>\n\
4  <html>\n\
5  <head>\n\
6  \t<meta charset='UTF-8'>\n\
7  \t<title>Task 3</title>\n\
8  </head>\n\
9  <body>\n\
10 \t<h1>Task 3</h1>\n"
11 const htmlEnd = "\t\t<br>\n\
12 \t\t<input type='submit' value='Send'>\n\
13 \t</form>\n\
14 </body>\n\
15 </html>\n";
16
17 function createArr(jsonArr) {
18   let arr = JSON.parse(jsonArr);
19   return arr;
20 }
21
22 function createPage(arr, address) {
23   fs.writeFileSync("task3End.html", htmlStart);
24   let adr = "\t\t<form method='GET' action=\"" + address + "\">\n";
25   fs.appendFileSync("task3End.html", adr);
26   for (let i = 0; i < arr.length; i++) {
27     let curStr = "\t\t<p>Input + arr[i] + "</p>\n" +
```

```

28     "\t\t<input name=\"\" + arr[i] + "\"" spellcheck='false' autocomplete
        ='off'>\n";
29     fs.appendFileSync("task3End.html", curStr);
30 }
31 fs.appendFileSync("task3End.html", htmlEnd);
32 }
33
34 const app = express();
35 const port = 5015;
36 app.listen(port);
37 console.log("My server on port " + port);
38
39 app.get("/me/page", function(request, response) {
40     const nameString = request.query.p;
41     if (fs.existsSync(nameString)) {
42         const contentString = fs.readFileSync(nameString, "utf8");
43         response.end(contentString);
44     } else {
45         const contentString = fs.readFileSync("bad.html", "utf8");
46         response.end(contentString);
47     }
48 });
49
50 app.get("/task3/createPage", function(request, response) {
51     let address = request.query.address;
52     try {
53         let arr = createArr(request.query.arr);
54         createPage(arr, address);
55         const contentString = fs.readFileSync("task3End.html", "utf8");
56         response.end(contentString);
57     } catch (error) {
58         const contentString = fs.readFileSync("bad.html", "utf8");
59         response.end(contentString);
60     }
61 });

```


10.3 Тесты

Task 3

Введите массив названий полей (JSON формат)

Введите адрес запроса

Рисунок 20 – Страница

Task 3

Введите поле First

Введите поле Second

Рисунок 21 – Результат

11 Задание 4.4

11.1 Условие

Запустить сервер. Реализовать на сервере функцию, которая принимает на вход числа А, В и С. Функция должна выдавать массив целых чисел на отрезке от А до В, которые делятся на С нацело.

11.2 Решение

Листинг 11 – Файл task4.js

```
1  "use strict";
2  const fs = require("fs");
3  const express = require("express");
4
5  const app = express();
6  const port = 5015;
7  app.listen(port);
8  console.log("My server on port " + port);
9
10 app.get("/me/page", function(request, response) {
11     const nameString = request.query.p;
12     if (fs.existsSync(nameString)) {
13         const contentString = fs.readFileSync(nameString, "utf8");
14         response.end(contentString);
15     } else {
16         const contentString = fs.readFileSync("bad.html", "utf8");
17         response.end(contentString);
18     }
19 });
20
21 app.get("/task4/section", function(request, response) {
22     let A = parseInt(request.query.A);
23     let B = parseInt(request.query.B);
24     let C = parseInt(request.query.C);
25     if (A < B) {
26         let arr = []
27         for (let i = A; i <= B; i++) {
28             if (i % C == 0) {
```

```
29     arr.push(i)
30   }
31 }
32 const answerJSON = JSON.stringify({result: arr});
33 response.end(answerJSON);
34 } else {
35   const contentString = fs.readFileSync("bad.html", "utf8");
36   response.end(contentString);
37 }
38 });
```

11.3 Тесты

Task 4

Введите А

Введите В

Введите С

Отправить запрос

Рисунок 22 – Страница

```
{"result": [0, 3, 6, 9]}
```

Рисунок 23 – Результат

Вывод

В ходе выполнения лабораторной работы я научился работать с строками и т.д. формата JSON. Познакомился с серверами, создал и поработал с ними.