|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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»

**Лабораторная работа № 2**

**По курсу «Архитектура ЭВМ»**

|  |  |
| --- | --- |
| **Тема**  Работа с форматом JSON и запуск серверов в JavaScript  **Студент** Якуба Д. В.  **Группа** ИУ7-53Б  **Оценка (баллы) \_\_\_\_\_\_\_\_\_\_\_**  **Преподаватель** Попов А. Ю. |  |

Москва

2020 г.

# Цели работы

* Освоение работы с форматом JSON и файлами в ЯП JavaScript.
* Работа со считыванием с клавиатуры в ЯП Javascript.
* Изучение запуска сервера, получения HTML страниц и работы с формами в ЯП JavaScript.

# Отчёт по разделу №1

## Задания раздела

### Условия

* С клавиатуры считывается число N. Далее считывается N строк. Необходимо создать массив и сохранять в него строки только с четной длинной. Получившийся массив необходимо преобразовать в строку JSON и сохранить в файл.
* 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.
* 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.
* Дана вложенная структура файлов и папок. Все файлы имеют раширение "txt". Необходимо рекурсивно перебрать вложенную структуру и вывести имена файлов, у которых содержимое не превышает по длине 10 символов.
* 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.
* 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.
* 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.

### Код программы

Язык: JavaScript

**tasks.js**

"use strict";

function task1()

{

    let numOfStrings = readlineSync.questionInt("Number of strings: ");

    let validStrings = [];

    for (let i = 0; i < numOfStrings; i++)

    {

        let str = readlineSync.question("String: ");

        if (str.length % 2 === 0)

            validStrings.push(str);

    }

    let jsonValidStrings = JSON.stringify(validStrings);

    fs.writeFileSync("task1out.json", jsonValidStrings);

}

function task2()

{

    let fileName = readlineSync.question("Insert file name: ");

    if (!fs.existsSync(fileName))

    {

        console.log("File doesn't exist. Going to next task...");

        return;

    }

    let readContents = fs.readFileSync(fileName, "utf-8");

    let strings = JSON.parse(readContents);

    let consonants = ['b', 'c', 'd', 'f', 'g', 'h', 'j',

                      'k', 'l', 'm', 'n', 'p', 'q', 'r',

                      's', 't', 'v', 'w', 'x', 'y', 'z'];

    let outString = [];

    console.log("Strings made of vowels:");

    for (let str of strings)

    {

        let stringValidityFlag = true;

        for (let i = 0; i < consonants.length && stringValidityFlag; i++)

            if (str.toLowerCase().includes(consonants[i]))

                stringValidityFlag = false;

        if (stringValidityFlag)

            console.log(str);

    }

    console.log("...And that's all.");

}

function readFileToLog(name)

{

    if (!fs.existsSync(name))

    {

        console.log("Hmmm... File doesn't exist.");

        return;

    }

    console.log("Reading file", name);

    let outString = fs.readFileSync(name, "utf-8");

    console.log(outString);

}

function task3()

{

    let extension = readlineSync.question("Insert extension in format '.txt'(.extension): ");

    let wayToFile = readlineSync.question("Insert way to file: ");

    let gotFileNames = fs.readdirSync(wayToFile);

    for (let name of gotFileNames)

    {

        if (name.endsWith(extension))

            readFileToLog(name);

    }

}

function searchTxtFiles(way)

{

    let files = fs.readdirSync(way);

    for (let fileName of files)

    {

        if (fileName.endsWith(".txt") && fs.readFileSync(way + fileName).length <= 10)

            console.log(way + fileName);

        else if (fs.lstatSync(way + fileName).isDirectory())

            searchTxtFiles(way + fileName + '/');

    }

}

function task4()

{

    let wayToFile = readlineSync.question("Insert way to file: ");

    if (!wayToFile.endsWith("/"))

        wayToFile += "/"

    searchTxtFiles(wayToFile);

}

function task5()

{

    let numOfNames = readlineSync.questionInt("Number of names (text files): ");

    let names = [];

    for (let i = 0; i < numOfNames; i++)

        names.push(readlineSync.question("Name: "));

    let outStringFromFiles = "";

    for (let name of names)

    {

        if (fs.existsSync(name))

            outStringFromFiles += fs.readFileSync(name, "utf-8");

    }

    fs.writeFileSync("outTask5.txt", outStringFromFiles);

}

function task6()

{

    let obj = 666;

    let counter = 0;

    try

    {

        while (true)

        {

            counter++;

            obj = {obj};

            JSON.stringify(obj);

        }

    }

    catch(err)

    {

        console.log("Error: ", err);

        console.log("Ladies and gentlemen we got him! ", counter);

    }

}

let maxDepth = 0;

let maxWay = "";

let maxValue;

function recursiveMaxBranchFinder(obj, curDepth, curWay)

{

    if (typeof(obj) !== "object")

    {

        if (curDepth > maxDepth)

        {

            maxDepth = curDepth;

            maxWay = curWay;

            maxValue = obj;

        }

    }

    else

    {

        curDepth++;

        for (let i in obj)

            recursiveMaxBranchFinder(obj[i], curDepth, curWay + "->" + i);

    }

}

function task7()

{

    let fileName = readlineSync.question("Insert file name: ");

    if (!fs.existsSync(fileName))

    {

        console.log("File doesn't exist. :(");

        return;

    }

    if (!fileName.endsWith(".json"))

    {

        console.log("File is not a .json file :(");

        return;

    }

    let gotObject = JSON.parse(fs.readFileSync(fileName));

    recursiveMaxBranchFinder(gotObject, 0, "root");

    console.log("Max Depth:", maxDepth, "\nWay:", maxWay, "\nValue:", maxValue);

}

function main()

{

    console.log("Task 1:");

    task1();

    console.log("\n\nTask 2:");

    task2();

    console.log("\n\nTask 3:");

    task3();

    console.log("\n\nTask 4:");

    task4();

    console.log("\n\nTask 5:");

    task5();

    console.log("\n\nTask 6:");

    task6();

    console.log("\n\nTask 7:");

    task7();

}

const readlineSync = require('readline-sync');

const fs = require("fs");

main();

### Результаты тестирования

node .\tasks.js

Task 1:

Number of strings: 4

String: ooeoeoeooaoeeeeeee

String: poicxvngmflkk

String: aaoaooaoaaoooo

String: sa

Task 2:

Insert file name: task1out.json

Strings made of vowels:

ooeoeoeooaoeeeeeee

aaoaooaoaaoooo

...And that's all.

Task 3:

Insert extension in format '.txt'(.extension): .txt

Insert way to file: .

Reading file task50.txt

На Украине из- за поломки пчел пострадали девять человек.

Ранее были проведены судебные разбирательства у двух свидетелей из села Бутуровка, обвиняемых в нарушении правил и порядка ввода в лес диких пчел. Об этом сообщает издание

« Независимая газета». В сентябре на юге страны было организовано массовое убийство лесных помещиков и их хозяев, которое привело к сокращению производства пчёл на Украине. Украинская область потеряла один миллион гектаров посевов крупного рогатого скота, около 400 тысяч голов, почти 100 тысяч пудов скота, а часть посевов по переработке угля

понесла потери от падения на воду пяти тонн груза.

Reading file task51.txt

Секретная служба США уничтожила в Сирии малину и кабачки и запросила за все вознаграждение. Во главе отряда были направлены офицеры полиции. Для того чтобы избежать ненужных жертв, главарю банды выдали оружие, взрывчатку и несколько гранат. Об этом сообщали в пресс- службе МИД, сообщает РИА Новости со ссылкой на радиопередачу « Спокойной ночи малыши». Кабачковые обезглавлены взрывпакетами

Reading file task52.txt

В начале августа в Казахстане прошел крупный скачок в технологичном развитии. Так, 12 июня было установлено два российских иранских инкубатора. По данным ВНИИТИ, у Казахстана появится 18 новорожденных детей в месяц. В июне появится еще 22 ребенка. Большие яйца на Украине несмоглибы вырасти в этом году. Созданию искусственного человека будут посвящены 20 лет на территории России, в то время как первые 100 миллионов человек будут работать только по программе развития человеческого мозга.

Task 4:

Insert way to file: txtTest

txtTest/1/lol/valid.txt

txtTest/1/valid.txt

txtTest/2/valid.txt

txtTest/valid.txt

txtTest/valid2.txt

Task 5:

Number of names (text files): 3

Name: task50.txt

Name: task51.txt

Name: task52.txt

Task 6:

Error: RangeError: Maximum call stack size exceeded

at JSON.stringify (<anonymous>)

at task6 (C:\Users\dobri\Desktop\ComputerArchitecture\lab03\tasks.js:125:18)

at main (C:\Users\dobri\Desktop\ComputerArchitecture\lab03\tasks.js:198:5)

at Object.<anonymous> (C:\Users\dobri\Desktop\ComputerArchitecture\lab03\tasks.js:207:1)

at Module.\_compile (internal/modules/cjs/loader.js:1137:30)

at Object.Module.\_extensions..js (internal/modules/cjs/loader.js:1157:10)

at Module.load (internal/modules/cjs/loader.js:985:32)

at Function.Module.\_load (internal/modules/cjs/loader.js:878:14)

at Function.executeUserEntryPoint [as runMain] (internal/modules/run\_main.js:71:12)

at internal/main/run\_main\_module.js:17:47

Ladies and gentlemen we got him! 5706

Task 7:

Insert file name: tree.json

Max Depth: 9

Way: root->10->11->12->13->140->150->160->170->180

Value: 1337

Файл task1out.json

["ooeoeoeooaoeeeeeee","aaoaooaoaaoooo","sa"]

Файл outTask5.txt

На Украине из- за поломки пчел пострадали девять человек.

Ранее были проведены судебные разбирательства у двух свидетелей из села Бутуровка, обвиняемых в нарушении правил и порядка ввода в лес диких пчел. Об этом сообщает издание « Независимая газета». В сентябре на юге страны было организовано массовое убийство лесных помещиков и их хозяев, которое привело к сокращению производства пчёл на Украине. Украинская область потеряла один миллион гектаров посевов крупного рогатого скота, около 400 тысяч голов, почти 100 тысяч пудов скота, а часть посевов по переработке угля понесла потери от падения на воду пяти тонн груза.

Секретная служба США уничтожила в Сирии малину и кабачки и запросила за все вознаграждение. Во главе отряда были направлены офицеры полиции. Для того чтобы избежать ненужных жертв, главарю банды выдали оружие, взрывчатку и несколько гранат. Об этом сообщали в пресс- службе МИД, сообщает РИА Новости со ссылкой на радиопередачу « Спокойной ночи малыши». Кабачковые обезглавлены взрывпакетами

В начале августа в Казахстане прошел крупный скачок в технологичном развитии. Так, 12 июня было установлено два российских иранских инкубатора. По данным ВНИИТИ, у Казахстана появится 18 новорожденных детей в месяц. В июне появится еще 22 ребенка. Большие яйца на Украине несмоглибы вырасти в этом году. Созданию искусственного человека будут посвящены 20 лет на территории России, в то время как первые 100 миллионов человек будут работать только по программе развития человеческого мозга.

Файл tree.json

{

    "000":

    {

        "010":

        {

            "020": 777

        },

        "011": 666

    },

    "10":

    {

        "11":

        {

            "12":

            {

                "13":

                {

                    "140":

                    {

                        "150":

                        {

                            "160":

                            {

                                "170":

                                {

                                    "180": 1337

                                }

                            }

                        }

                    },

                    "141":

                    {

                        "1411": 333

                    }

                }

            }

        },

        "110":

        {

            "1100" : 0

        }

    }

}

# Отчёт по разделу №2

## Задание 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.

### Код программы

Язык: JavaScript

**task1.js**

"use strict";

const express = require("express");

const app = express();

const port = 5015;

app.listen(port);

console.log("My server on port " + port);

app.use(express.static(\_\_dirname));

const fs = require("fs");

app.get("/", function(request, response)

{

    const nameString = request.query.p;

    if (fs.existsSync(nameString)) {

        const contentString = fs.readFileSync(nameString, "utf8");

        response.end(contentString);

    } else {

        const contentString = fs.readFileSync("badGetaway.html", "utf8");

        response.end(contentString);

    }

});

app.get("/comparison", function(request, response)

{

    let fNum = request.query.fNum;

    let sNum = request.query.sNum;

    let tNum = request.query.tNum;

    let fInt = parseInt(fNum);

    let sInt = parseInt(sNum);

    let tInt = parseInt(tNum);

    let answer = Math.max(fInt, sInt, tInt);

    let answJSON = JSON.stringify({result: answer});

    response.end(answJSON);

});

**Файл comparison.html**

<!DOCTYPE html>

<html>

<head>

    <meta charset="UTF-8">

    <title>Нахождение наибольшего из трёх чисел</title>

</head>

<body>

    <h1>Числа:</h1>

    <form method="GET" action="/comparison">

        <p>Первое значение</p>

        <input name="fNum" spellcheck="false" autocomplete="off">

        <p>Второе значение</p>

        <input name="sNum" spellcheck="false" autocomplete="off">

        <p>Третье значение</p>

        <input name="tNum" spellcheck="false" autocomplete="off">

        <br>

        <br>

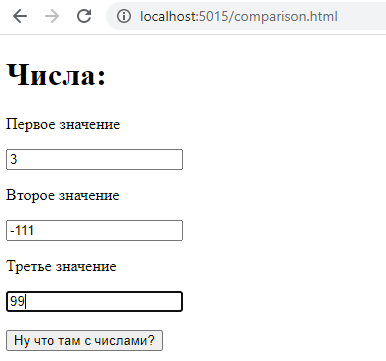
        <input type="submit" value="Ну что там с числами?">

    </form>

</body>

</html>

### Результаты тестирования





## Задание 2

### 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.

### Код программы

Язык: JavaScript

**task2.js**

"use strict";

const express = require("express");

const app = express();

const port = 5015;

app.listen(port);

console.log("My server on port " + port);

app.use(express.static(\_\_dirname));

const fs = require("fs");

app.get("/", function(request, response)

{

    const nameString = request.query.p;

    if (fs.existsSync(nameString)) {

        const contentString = fs.readFileSync(nameString, "utf8");

        response.end(contentString);

    } else {

        const contentString = fs.readFileSync("badGetaway.html", "utf8");

        response.end(contentString);

    }

});

app.get("/BlackTongue", function(request, response)

{

    let strNum = request.query.strNum;

    let strNumInt = parseInt(strNum);

    strNumInt--;

    if (strNumInt < 0)

    {

        response.end("Index is negative. But must be a positive. What?..")

        return;

    }

    let arr = JSON.parse(fs.readFileSync("BlackTongue.json"));

    if (strNumInt >= arr.length - 1)

    {

        response.end("We are out of index. Max is " + (arr.length - 1).toString());

        return;

    }

    response.end(JSON.stringify(arr[strNumInt]));

});

**Файл BlackTongue.html**

<!DOCTYPE html>

<html>

<head>

    <meta charset="UTF-8">

    <title>У меня тут текстики одной интересной группы есть. Какая строчка песени интересует? ;)</title>

</head>

<body>

    <h1> У меня тут текстики одной интересной группы есть. Какая строчка песени интересует? ;)</</h1>

    <form method="GET" action="/BlackTongue">

        <p>Номер строчки вот сюда вводить:</p>

        <input name="strNum" spellcheck="false" autocomplete="off">

        <br>

        <br>

        <input type="submit" value="Я люблю битдаун, дайте мне строчку!">

    </form>

</body>

</html>

**Файл BlackTongue.json**

[

    "I'm so sick of your temporary bullshit",

    "I want the light at the end",

    "And I know how to get it",

    "If it takes my life then I will gladly die",

    "Open your mind",

    "Realise we are slaves",

    "Forced Draconian mind-state",

    "Open your eyes",

    "We live in a world made of smoke and mirrors",

    {

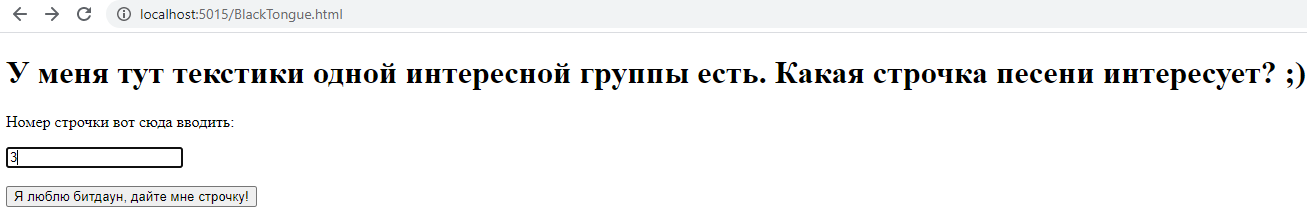
        "type": "string",

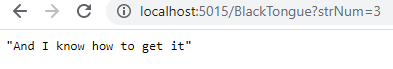
        "value": 10

    }

]

### Результаты тестирования





## Задание 3

### 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.

### Код программы

Язык: JavaScript

**task3.js**

"use strict";

const express = require("express");

const app = express();

const port = 5015;

app.listen(port);

console.log("My server on port " + port);

app.use(express.static(\_\_dirname));

const fs = require("fs");

app.get("/", function(request, response)

{

    const nameString = request.query.p;

    if (fs.existsSync(nameString)) {

        const contentString = fs.readFileSync(nameString, "utf8");

        response.end(contentString);

    } else {

        const contentString = fs.readFileSync("badGetaway.html", "utf8");

        response.end(contentString);

    }

});

function formGen(fieldsNames)

{

    let out = "";

    for (let field of fieldsNames)

        out += `<p>${field}</p>

        <input name="${field}" spellcheck="false" autocomplete="off">

        <br>`;

    out += `<input type="submit" value="Тык!">`;

    out += "</form>";

    return out;

}

// Тест: http://localhost:5015/HTMLgenerator?fieldsNames[]=strNum&address=/BlackTongue

function HTMLGenerator(fieldsNames, address)

{

    if (address[0] !== '/')

        address = '/' + address;

    let out = `<!DOCTYPE html>

    <html>

    <head>

        <meta charset="UTF-8">

        <title>Ресурс отсутствует</title>

        <style>

            body {

                font-family: Geneva, Arial, Helvetica, sans-serif;

                padding: 35px;

                background: whitesmoke;

            }

            h1 {

                color: rebeccapurple;

            }

        </style>

    </head>

    <body>`;

    out += `<form method="GET" action="${address}">`;

    out += formGen(fieldsNames);

    out += "</body></html>";

    return out

}

app.get("/HTMLgenerator", function(request, response)

{

    let fieldsNames = request.query.fieldsNames;

    let address = request.query.address;

    let generated = HTMLGenerator(fieldsNames, address);

    console.log(generated);

    response.end(generated);

});

app.get("/BlackTongue", function(request, response)

{

    let strNum = request.query.strNum;

    let strNumInt = parseInt(strNum);

    strNumInt--;

    if (strNumInt < 0)

    {

        response.end("Index is negative. But must be a positive. What?..")

        return;

    }

    let arr = JSON.parse(fs.readFileSync("BlackTongue.json"));

    if (strNumInt >= arr.length - 1)

    {

        response.end("We are out of index. Max is " + (arr.length - 1).toString());

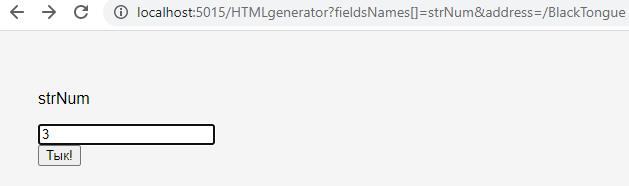
        return;

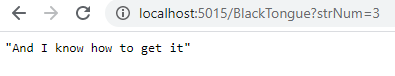
    }

    response.end(JSON.stringify(arr[strNumInt]));

});

### Результаты тестирования





## Задание 4

### Условие

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

### Код программы

Язык: JavaScript

**task4.js**

"use strict";

const express = require("express");

const app = express();

const port = 5015;

app.listen(port);

console.log("My server on port " + port);

app.use(express.static(\_\_dirname));

const fs = require("fs");

app.get("/", function(request, response)

{

    const nameString = request.query.p;

    if (fs.existsSync(nameString)) {

        const contentString = fs.readFileSync(nameString, "utf8");

        response.end(contentString);

    } else {

        const contentString = fs.readFileSync("badGetaway.html", "utf8");

        response.end(contentString);

    }

});

function findInts(A, B, C)

{

    let startNum = 0;

    if (A % C !== 0)

        startNum = A + (C - A % C);

    else

        startNum = A;

    let retNums = [];

    for (let cur = startNum; cur <= B; cur += C)

        retNums.push(cur);

    return retNums;

}

// Тест: http://localhost:5015/intsIn?A=1&B=20&C=4

// Тест: http://localhost:5015/intsIn?A=1&B=20&C=0

// Тест: http://localhost:5015/intsIn?A=20&B=1&C=9

app.get("/intsIn", function(request, response)

{

    let A = request.query.A;

    let B = request.query.B;

    let C = request.query.C;

    let AInt = parseInt(A);

    let BInt = parseInt(B);

    if (AInt > BInt)

    {

        let rec = AInt;

        AInt = BInt;

        BInt = rec;

    }

    let CInt = parseInt(C);

    if (CInt === 0)

    {

        response.end("C had to be a non-zero value");

        return;

    }

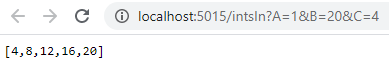
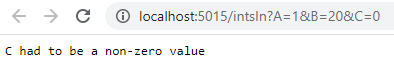
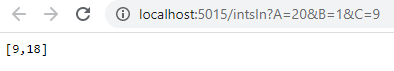
    let answer = findInts(AInt, BInt, CInt);

    let answJSON = JSON.stringify(answer);

    response.end(answJSON);

});

### Результаты тестирования

## Вывод

В результате выполнения работы:

* Была освоена работа с форматом JSON и файлами в ЯП JavaScript.
* Было изучено считывание с клавиатуры в ЯП Javascript.
* Были запущены сервера, получены HTML страницы и была произведена работа с формами в ЯП JavaScript.