|  |  |
| --- | --- |
| 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 **Компьютерные системы и сети (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.01 Информатика и вычислительная техника**

**Отчет**

|  |  |
| --- | --- |
| **по лабораторной работе №** | 2 |

**Дисциплина:** Архитектура ЭВМ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ7-52Б |  |  | В.А. Иванов |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | А.Ю. Попов |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2020

**Цель работы**

Целью лабораторной работы освоение методов работы с html страницами в JavaScript.

Задание 3.1

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.

**Программная реализация**

"use strict";

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

const fs = require("fs");

const nameString = "test.txt";

const n = parseInt(readlineSync.question("Input N: "));

let data = [];

for (let i=0; i < n; i++) {

    let str = readlineSync.question("Input string: ");

    if (str.length % 2 == 0) {

        data.push(str);

    }

}

console.log(data);

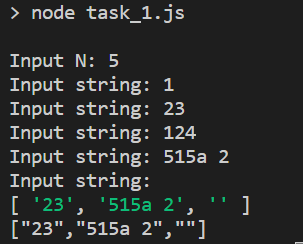
let parse\_data = JSON.stringify(data);

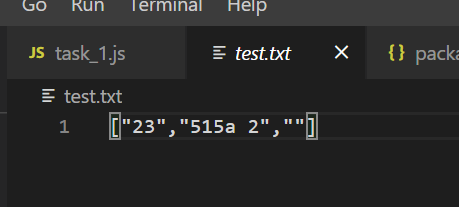
console.log(parse\_data);

fs.writeFileSync(nameString, parse\_data);

**Тесты**

Ввод:

  
Вывод:



Задание 3.2

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.

**Программная реализация**

"use strict";

const vowles = "AEIOU"

const fs = require("fs");

const nameString = "test.txt";

const str = fs.readFileSync(nameString, "utf-8");

console.log("Scaned string: " + str);

console.log("Vowles strings:");

let data = JSON.parse(str);

for (let i in data) {

    let str = data[i].toUpperCase();

    let j=0;

    for (; j < str.length; j++) {

        if (!(vowles.includes(str[j])))

            break;

    }

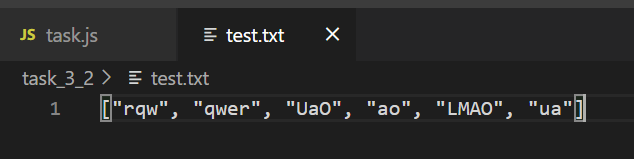
    if (j == str.length)

        console.log(data[i]);

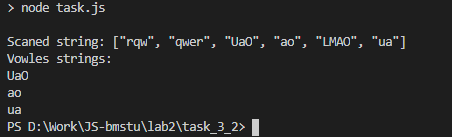
}

**Тесты**

Ввод:



Вывод:



Задание 3.3

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.

**Программная реализация**

"use strict";

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

const fs = require("fs");

const ext = readlineSync.question("Input file extension: ");

const addr = readlineSync.question("Input folder addres: ");

if (fs.existsSync(addr)) {

    const file\_arr = fs.readdirSync(addr);

    file\_arr.forEach(f\_name => {

        let parts = f\_name.split('.');

        if (parts[parts.length-1] == ext) {

            console.log(f\_name + ":");

            let str = fs.readFileSync(addr+"\\"+f\_name, "utf-8");

            if (str == "") {

                console.log("Empty file");

            } else {

                console.log(str);

            }

        }

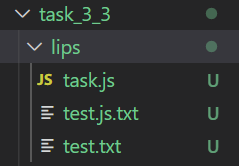
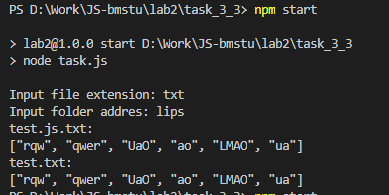
    });

} else {

    console.log("Folder not exists");

}

**Тесты**

Задание 3.4

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.

**Программная реализация**

"use strict";

const fs = require("fs");

function lookup\_folder(addr){

    if (!fs.existsSync(addr)) { return; }

    const file\_arr = fs.readdirSync(addr);

    file\_arr.forEach(f\_name => {

        let parts = f\_name.split('.');

        if (parts.length == 1) {

            lookup\_folder(addr+"\\"+f\_name);

        } else {

            let str = fs.readFileSync(addr+"\\"+f\_name, "utf-8");

            if (str.length <= 10) {

                console.log(addr+"\\"+f\_name);

                console.log(str);

            }

        }

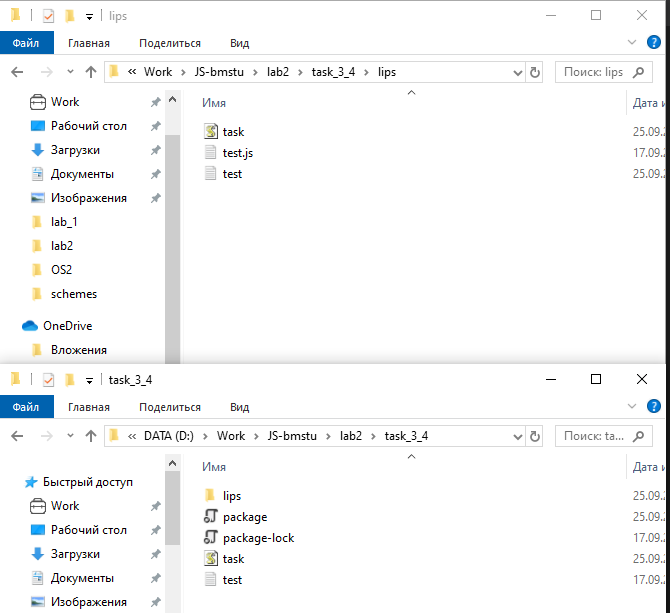
    });

}

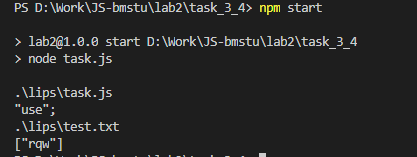
lookup\_folder(".");

**Тесты**

Ввод:



Вывод:



Задание 3.5

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.

**Программная реализация**

"use strict";

const fs = require("fs");

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

let result\_str = "";

const n = parseInt(readlineSync.question("Input N: "));

for (let i=0; i<n; i++){

    let f\_name = readlineSync.question("Input file name: ");

    if (fs.existsSync(f\_name)) {

        let str = fs.readFileSync(f\_name, "utf-8");

        result\_str += str;

    } else {

        console.log("File not exists");

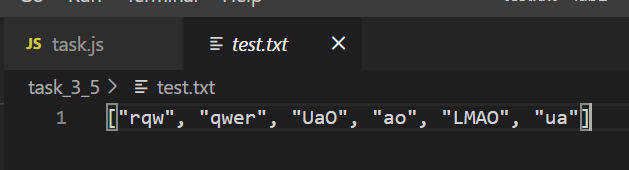
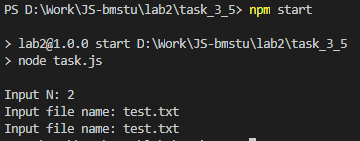
    }

}

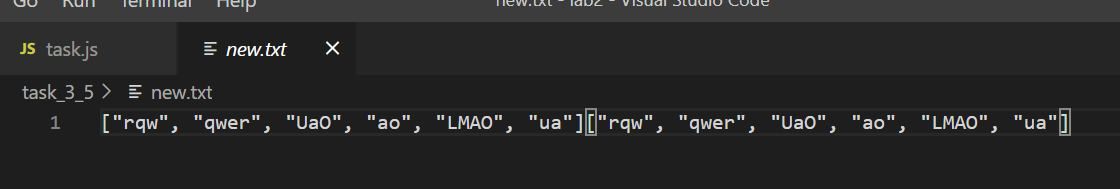
fs.writeFileSync("new.txt", result\_str);

**Тесты**

Ввод:

Вывод:



Задание 3.6

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.

**Программная реализация**

"use strict";

const e = require("express");

class Box {

    constructor (depth) {

        this.d = depth;

        if (depth > 0)

            this.next = new Box(depth-1);

        else

            this.next = NaN;

    }

}

let size = 100;

let step = 128;

while (step > 1) {

    try {

        let b = new Box(size);

        let jsonStr = JSON.stringify(b);

        size += step;

    } catch (RangeError) {

        size -= step;

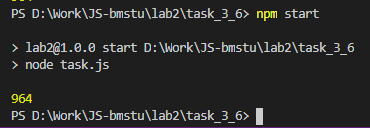
        step /= 2;

    }

}

console.log(size);

**Тесты**



Задание 3.7

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.

**Программная реализация**

"use strict";

const fs = require("fs");

const f\_name = "json-data.txt";

function create\_file() {

    const probability = 0.53;

    class Box {

        constructor (depth) {

            this.d = depth;

            this.leaf1 = NaN;

            this.leaf2 = NaN;

            if (Math.random() > 1 - probability)

                this.leaf1 = new Box(depth+1);

            if (Math.random() > 1 - probability)

                this.leaf2 = new Box(depth+1);

        }

    }

    let b = new Box(0);

    let jStr = JSON.stringify(b);

    console.log(jStr);

    fs.writeFileSync(f\_name, jStr);

}

function path\_lenght(path) {

    return (path.match(new RegExp("/", "g")) || []).length

}

function process\_tree(tree) {

    let path = "";

    for (let key in tree) {

        if (typeof(tree[key]) == "object" && tree[key] != null) {

            let t\_path = key + "/" + process\_tree(tree[key]);

            if (path\_lenght(t\_path) > path\_lenght(path))

                path = t\_path;

        }

    }

    return path;

}

// create\_file();

let jStr = fs.readFileSync(f\_name, "utf-8");

console.log("Scaned JSON-data: " + jStr);

let original\_tree = JSON.parse(jStr);

let path = process\_tree(original\_tree);

let path\_l = path\_lenght(path);

console.log("Path: " + path.slice(0, path.length-1));

console.log("Length: " + path\_l);

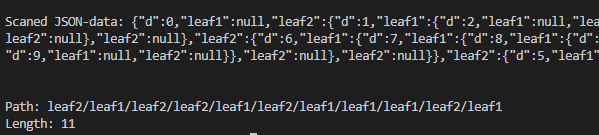
**Тесты**

Ввод:

Для проверки работоспособности программы была создана функция, записывающая в файл древовидную структуру в формате JSON. В тесте использована строка:

{"d":0,"leaf1":null,"leaf2":{"d":1,"leaf1":{"d":2,"leaf1":null,"leaf2":{"d":3,"leaf1":null,"leaf2":{"d":4,"leaf1":{"d":5,"leaf1":{"d":6,"leaf1":{"d":7,"leaf1":null,"leaf2":null},"leaf2":null},"leaf2":{"d":6,"leaf1":{"d":7,"leaf1":{"d":8,"leaf1":{"d":9,"leaf1":null,"leaf2":{"d":10,"leaf1":{"d":11,"leaf1":null,"leaf2":null},"leaf2":null}},"leaf2":{"d":9,"leaf1":null,"leaf2":null}},"leaf2":null},"leaf2":null}},"leaf2":{"d":5,"leaf1":null,"leaf2":null}}}},"leaf2":null}}

Вывод:



Задание 3.3

**Программная реализация**

**Тесты**

Ввод:

Вывод:

Задание 3.3

**Программная реализация**

**Тесты**

Ввод:

Вывод:

Задание 3.3

**Программная реализация**

**Тесты**

Ввод:

Вывод:

Задание 3.3

**Программная реализация**

**Тесты**

Ввод:

Вывод:

**Вывод**

В рамках лабораторной работы было выполнено ознакомление и практическое закрепление основ работы с JSON, потоками ввода/вывода, управления серверами в языке JS.