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»

Task № 3

Дисциплина: Архитектура ЭВМ

Тема : Работа с Файлами.

Студент: Расколотов Д.Ю.

Группа: ИУ7-54Б

Преподаватель: Повов А.Ю.

Москва.
2020 г.

Цель работы: Научиться работать с JSON и Файлами.

Задание 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.

Код программы:

```
"use strict";

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

const arr = [];
const fs = require("fs");
const nameString = "a1.txt";

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

if (isNaN(a)) {
  console.log("Error input");
} else {
  console.log(a);
  for (let i = 0; i < a; i++) {
    let value = readlineSync.question("Input string: ");
    if (value.length % 2 == 0) {
      arr.push(value);
    }
  }
  const jsonString = JSON.stringify(arr);
  fs.writeFileSync(nameString, jsonString);
  console.log("Create File - OK");
}
```

Тестовая часть:

```
Input number n: 5
5
Input string: 763
Input string: 1234
Input string: 789634
Input string: 12345
Input string: 5
Create File - OK
```

a1.txt

```
1   ["1234","789634"]
```

Задание 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.

Код программы:

```
"use strict";


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

const arr = [];
const fs = require("fs");
const nameString = "a2.txt";

if (fs.existsSync(nameString)) {
  console.log("File exists");
  const MyStr = fs.readFileSync(nameString, "utf-8");
  const MyStrPs = JSON.parse(MyStr);
  console.log(MyStrPs);
  const simb = ["a", "e", "i", "o", "u", "y"];
  for (let i = 0; i < MyStrPs.length; i++) {
    let flag = true;
```

```
    for (let j = 0; j < MyStrPs[i].length; j++) {
      if (!simb.includes(MyStrPs[i][j].toLowerCase())) {
        flag = false;
        break;
      }
    }
    if (flag) {
      console.log(MyStrPs[i]);
    }
  }
} else {
  console.log("File was not found");
}
```

Тестовая часть:

a2.txt

```
["qw","qwer","qwerty","aaaooo","aabyy","aoeyu","eyuioa","qeyuioa"]
```

```
File exists
[
  'qw',      'qwer',
  'qwerty',  'aaaooo',
  'aabyy',   'aoeyu',
  'eyuioa',  'qeyuioa'
]
С гласной: aaaooo
С гласной: aoeyu
С гласной: eyuioa
```

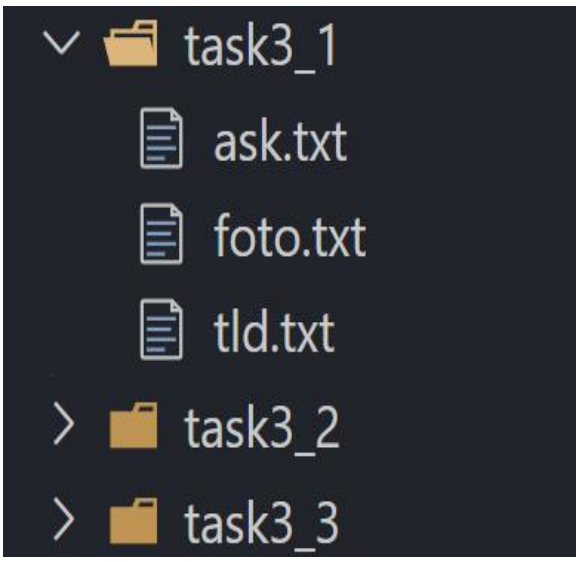
Задание 3:

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.

Код программы:

```
"use strict";

const readlineSync = require("readline-sync");
const fs = require("fs");

const a = readlineSync.question("Input expansion: ");
const folder = readlineSync.question("Input way: ");

const arr = fs.readdirSync(folder);
console.log(arr);
let dot = ".";
for (let i = 0; i < arr.length; i++) {
  if (arr[i].split(dot)[1] === a) {
    console.log(arr[i]);
    console.log(fs.readFileSync("./" + folder + "/"+ arr[i], "u
tf8"));
  }
}
```

Тестовая часть:



```
Input expansion: txt
Input way: task3_1
[ 'ask.txt', 'foto.txt', 'tld.txt' ]
ask.txt
asfdsaf
foto.txt
sfasgsgf
tld.txt
fasf
```

Задание 4:

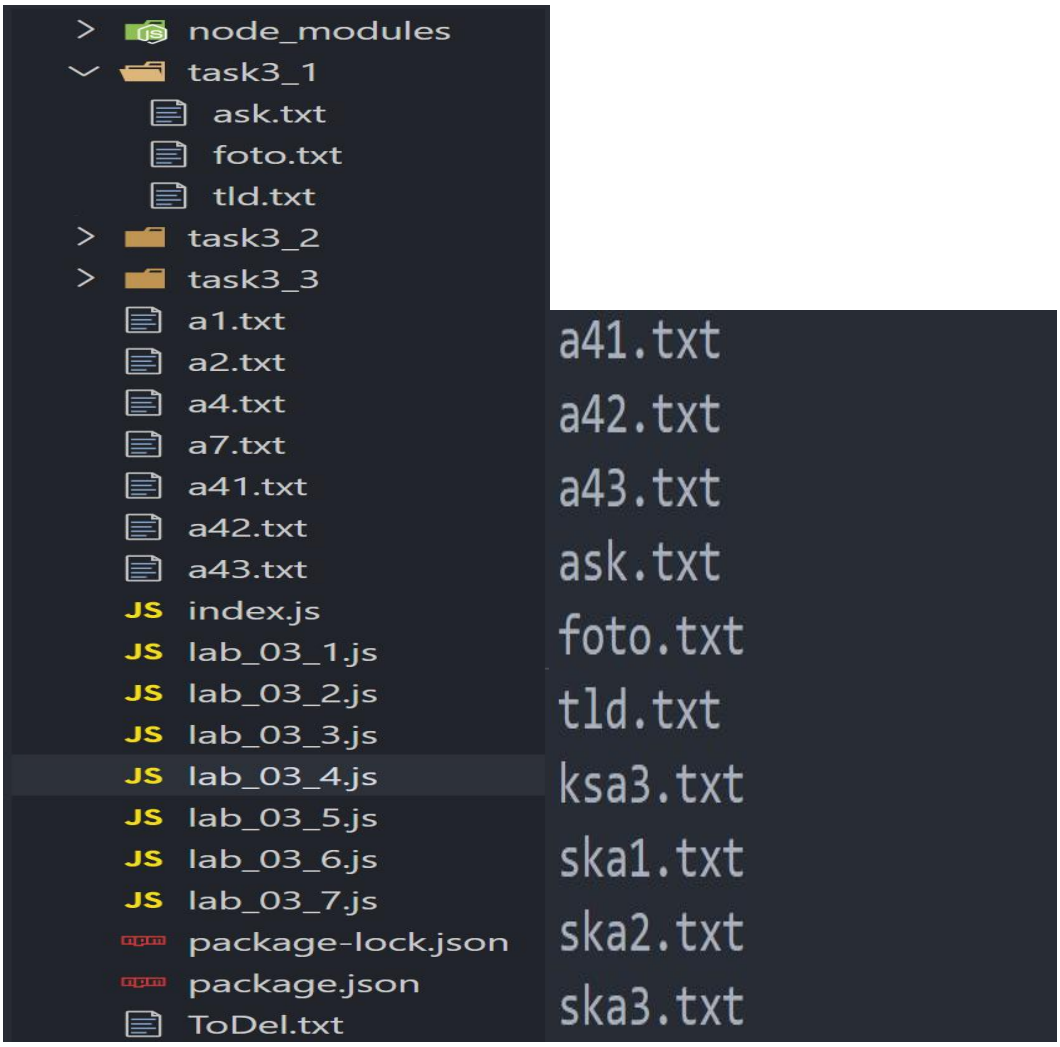
Дана вложенная структура файлов и папок. Все файлы имеют раширение "txt". Необходимо рекурсивно перебрать вложенную структуру и вывести имена файлов, у которых содержимое не превышает по длине 10 символов.

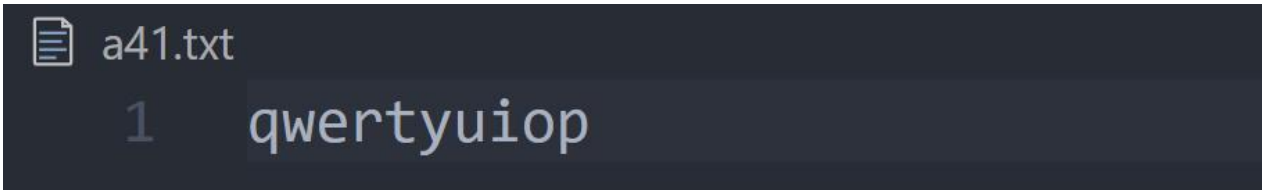
Код программы:

```
"use strict";

const readlineSync = require("readline-sync");
const fs = require("fs");
let dot = ".";

function searchFile(wayF) {
  let direct = fs.readdirSync(wayF);
  for (let file of direct) {
    let stat = fs.statSync(wayF + file);
    if (!stat.isFile()) {
      let nwayF = wayF + file + "/";
      searchFile(nwayF);
    } else {
      const fullContent = fs.readFileSync(wayF + file, "utf-
8");
      if (fullContent.length <= 10 && file.split(dot)[1] === "t
xt") {
        console.log(file);
      }
    }
  }
```

```
}

searchFile("./");
```

Тестовая часть:





Задание 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.

Код программы:

```
"use strict";

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

const fs = require("fs");
const nameString = "a4.txt";

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

if (isNaN(a)) {
  console.log("Error input");
} else {
  console.log(a);
  let flag = true;
  let ex = 0; //вхождения
  for (let i = 0; i < a; i++) {
    let value = readlineSync.question("Input string: ");
    if (!fs.existsSync(value)) {
      flag = false;
      break;
    }
    const fullContent = fs.readFileSync(value, "utf-8");
    if (ex === 0) {
      fs.writeFileSync(nameString, fullContent);
      ex++;
    } else {
      fs.appendFileSync(nameString, fullContent);
    }
  }
  if (!flag) {
    console.log("ERROR");
  }
}
```

Тестовая часть:

```
Input number n: 3
3
Input string: a41.txt
Input string: a42.txt
Input string: a43.txt
```

a41.txt

```
1  qwertyuiop
```

a42.txt

```
1  asdfghjkl
```

a43.txt

```
1  zxcvbnnm
```

a4.txt

```
1  qwertyuiopasdfghjklzxcvbnnm
```

Задание 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.

Код программы:

```
"use strict";

const fs = require("fs");

function maxExition(obj) {
  let i = 0;
  let flag = true;

  while (i < 5000 && flag) {
    try {
      const myString = JSON.stringify(obj);
      obj = { deep: obj };
```

```
    } catch (err) {
      console.log(err.message);
      flag = false;
    }
    i++;
  }
  return i;
}

let obj = {};
let Qty = maxExition(obj);
console.log(Qty);
```

Тестовая часть:

```
PS D:\AVM\lab_03> node .\lab_03_6.js
Maximum call stack size exceeded
966
```

Задание 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.

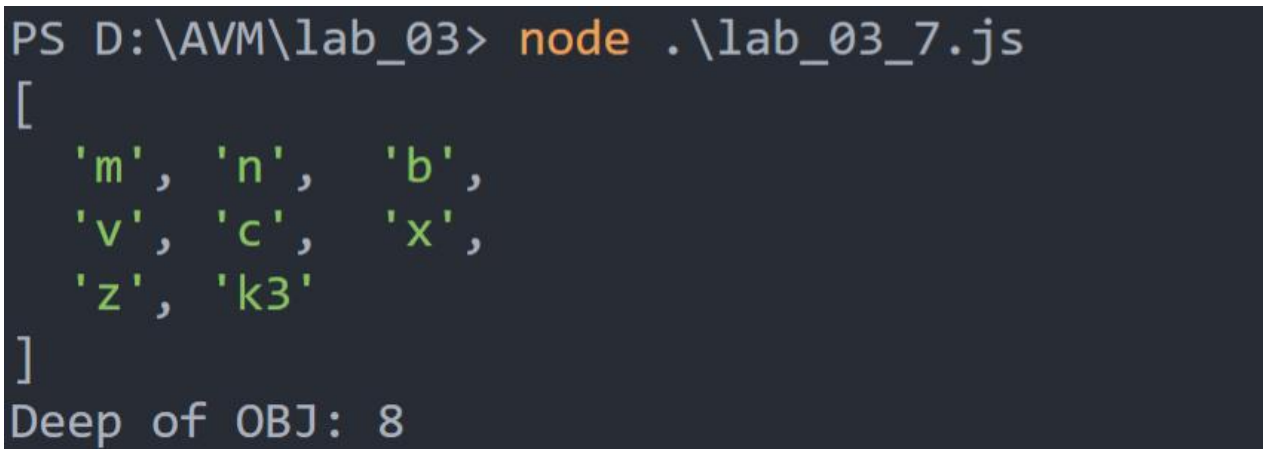
Код программы:

```
"use strict";

const fs = require("fs");

const king = {
  k1: {
    q: 43,
    qw: "female",
    h: {
      pop: 67,
      pip: {
        ip: {
          i: {
```

```
          tt: 228,
        },
        p: {
          tf: "angle",
        },
      },
      pi: {
        pp: {
          ytro: "dobroe",
          noch: "xolodnay",
        },
        ii: {
          rich: 25,
          marry: 28,
        },
      },
    },
  },
},
k2: {
  qwerty: {
    qwert: {
      qwer: {
        qwe: {
          gg: "end",
        },
      },
    },
  },
},
k3: {
  z: {
    x: {
      c: {
        v: {
          b: {
            n: {
              m: {
                kaef: "imenno",
              },
```

```
            },
          },
        },
      },
    },
  },
};
// console.log(king);

const myObjStr = JSON.stringify(king);
const nameFile = "a7.txt";

fs.writeFileSync(nameFile, myObjStr);

const readFile = fs.readFileSync(nameFile, "utf-8");
const mainObj = JSON.parse(readFile);

// console.log(mainObj);

let ways = []; // путь от самого глубокого до начала
function maxFieldEx(obj) {
  let col = 0;
  for (let key in obj) {
    if (typeof obj[key] === "object") {
      col = Math.max(maxFieldEx(obj[key]), col);
      ways[col - 1] = key;
    }
  }
  col++;
  return col;
}

maxFieldEx(mainObj);
console.log(ways);
console.log("Deep of OBJ: " + ways.length);
```

Тестовая часть:



Заключительная часть:

- Таким образом, мы научились преобразовывать объекты в формат строки **JSON.**
- Таким образом, если мы планируем преобразовать объект в строку **JSON**, надо проектировать структуру объекта с учетом запрета зацикленных структур.
- Таким образом, мы научились получать объект из строки **JSON.**
- Таким образом, мы разобрали базовые функции для взаимодействия с файловой системой.
- Таким образом, мы разобрались с возможностью ввода информации с клавиатуры