**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

###### ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

###### КЕМЕРОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

**ИНСТИТУТ ЦИФРЫ**

**ОТЧЁТ**

**О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

«Асинхронность»

Студентки 2 курса, ФИТ-211 группы

**Колесник Полины Олеговны**

Направление 02.03.02 – «Фундаментальная информатика и информационные технологии»

Руководитель:

Доцент Зимин А. И.

Работа защищена

« »

“ ” 2023 г.

Кемерово 2023 г.

**ОТЧЁТ О ПРОДЕЛАННОЙ РАБОТЕ**

**Объекты**

**1 задание**

function ask\_password(login, password, success, failure) {

    let vowels = ["a", "e", "i", "o", "u", "y"];

    let consonants = ["b", "c", "d", "f", "g", "h", "j", "k", "n", "p", "q", "r", "s", "t", "v", "w", "x", "z"];

    let countV = 0;

    let countC = 0;

    for (const key in password) {

        const letter = password[key];

        if(password.length === 6) {

            if (vowels.includes(letter) && login.includes(letter)){

                countV += 1;

            }

            else if (consonants.includes(letter) && login.includes(letter)){

                countC += 1;

            }

        }

    }

    if (countV === 3 && countC === 3) {

        success(login);

    }

    else if (countC != 3) {

        failure(login, "Wrong consonants");

    }

    else if (countV != 3) {

        failure(login, "Wrong number of vowels");

    } else {

        failure(login, "Everything is wrong");

    }

}

function main(login, password) {

    ask\_password(login, password, function(log) {

        console.log(`Привет, ${login}!`);

        },

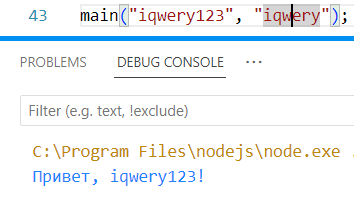
        function(log, msg) {

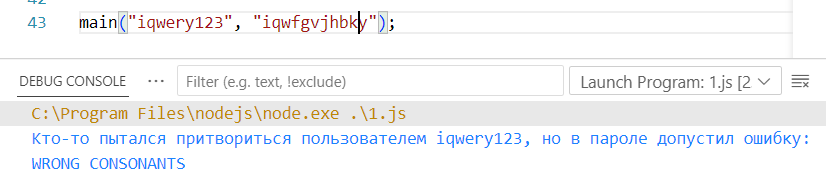
            console.log(`Кто-то пытался притвориться пользователем ${log}, но в пароле допустил ошибку: ${msg.toUpperCase()}`);

        });

}

main("iqwery123", "iqwery");





**2 задание**

// Асинхронные функции

function readConfig (name, callback) {

    setTimeout(() => {

        console.log('(1) config from ' + name + ' loaded')

        callback()

    }, Math.floor(Math.random() \* 1000))

}

function doQuery (statement, callback) {

    setTimeout(() => {

        console.log('(2) SQL query executed: ' + statement)

        callback()

    }, Math.floor(Math.random() \* 1000))

}

function httpGet (url, callback) {

    setTimeout(() => {

        console.log('(3) Page retrieved: ' + url)

        callback()

    }, Math.floor(Math.random() \* 1000))

}

function readFile (path, callback) {

    setTimeout(() => {

        console.log('(4) Readme file from ' + path + ' loaded')

        callback()

    }, Math.floor(Math.random() \* 1000))

}

function callback () {

    console.log('It is done!')

}

// Вызов функций

console.log('start')

//a

readConfig('myConfig', () => {

    doQuery('select \* from cities', () => {

        httpGet('http://google.com', () => {

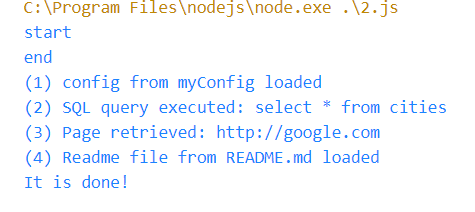
            readFile('README.md', callback)

        })

    })

})

console.log('end')



// Асинхронные функции

function readConfig (name, callback) {

    setTimeout(() => {

        console.log('(1) config from ' + name + ' loaded')

        callback()

    }, Math.floor(Math.random() \* 1000))

}

function readConfigNotify() {

    doQuery('select \* from cities', doQueryNotify);

}

function doQuery (statement, callback) {

    setTimeout(() => {

        console.log('(2) SQL query executed: ' + statement)

        callback()

    }, Math.floor(Math.random() \* 1000))

}

function doQueryNotify (){

    httpGet('http://google.com', httpGetNotify);

}

function httpGet (url, callback) {

    setTimeout(() => {

        console.log('(3) Page retrieved: ' + url)

        callback()

    }, Math.floor(Math.random() \* 1000))

}

function httpGetNotify() {

    readFile('README.md', callback)

}

function readFile (path, callback) {

    setTimeout(() => {

        console.log('(4) Readme file from ' + path + ' loaded')

        callback()

    }, Math.floor(Math.random() \* 1000))

}

function callback () {

    console.log('It is done!')

}

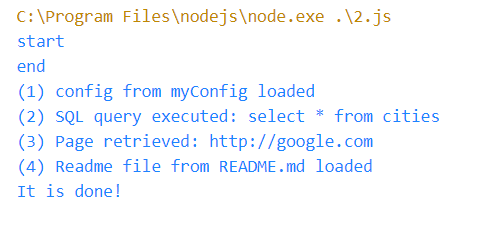
// Вызов функций

console.log('start')

//b

readConfig('myConfig', readConfigNotify);

console.log('end')



**3 задание**

function F(x)

{

    f1(x);

}

function f1(x) {

    setTimeout(() => {

        let result = x\*\*2;

        console.log(`Промежуточный результат ${result}`);

        f1Notify(x, result)

    }, 1000)

}

function f1Notify(x, result) {

    f2(x, result)

}

function f2(x, result) {

    setTimeout(() => {

        let res = result + x \* 2;

        console.log(`Промежуточный результат ${res}`)

        f2Notify(x, res)

    }, 1000)

}

function f2Notify(x, result) {

    console.log(`Pезультат при n = 2: ${result}`)

    f3(x, result)

}

function f3(x, result) {

    setTimeout(() => {

        let res = result - 2;

        console.log(`Промежуточный результат ${res}`)

        f3Notify(x, res)

    }, 1000)

}

function f3Notify(x, result) {

    f4(x, result)

}

function f4(x, result) {

    setTimeout(() => {

        let res = result \*\* 3

        console.log(`Промежуточный результат ${res}`)

        f4Notify(x, res)

    }, 1000)

}

function f4Notify(x, result) {

    console.log(`Pезультат при n = 4: ${result}`)

    f5(x, result)

}

function f5(x, result) {

    setTimeout(() => {

        let res = result \* 3

        console.log(`Промежуточный результат ${res}`)

        f5Notify(x, res)

    }, 1000)

}

function f5Notify(x, result) {

    f6(x, result)

}

function f6(x, result) {

    setTimeout(() => {

        let res = result - 3

        console.log(`Промежуточный результат ${res}`)

        f6Notify(x, res)

    }, 1000)

}

function f6Notify(x, result) {

    console.log(`Pезультат при n = 6: ${result}`)

}

console.log('start')

F(3);

console.log('end')

