**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информатика»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2

по дисциплине **«Разработка приложений для Интернет»**

на тему: **«Работа с файловой системой»**

Выполнил: студент гр. ИП-31

Кузнецова Е. А.

Принял: ст. преподаватель

Фролова Е. В.

Дата сдачи отчета: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата допуска к защите: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата защиты: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Гомель 2024

**Цель работы:** изучить основные возможности по работе с файловой системой и потоками в Node.js.

**Задание 1.** Написать приложение, которое логирует ошибки (вводятся пользователем) в текстовом файле. В файл добавляется дата ошибки и ее описание.

**Код программы:**

const fs = require('fs');

const readline = require('readline');

const rl = readline.createInterface({

input: process.stdin,

output: process.stdout

});

// Функция для логирования ошибок в файл

function logError(error, callback) {

const date = new Date().toISOString();

const logEntry = `[${date}] ${error}\n`;

fs.appendFile('error.log', logEntry, (err) => {

if (err) throw err;

callback(); // Вызываем колбэк после успешной записи в файл

});

}

// Функция для запроса описания ошибки

function promptForError() {

rl.question('Введите описание ошибки (или "exit" для завершения): ', (errorDescription) => {

if (errorDescription.toLowerCase() === 'exit') {

rl.close(); // Завершаем работу приложения если пользователь ввел "exit"

} else {

logError(errorDescription, () => {

console.log('Ошибка зарегистрирована в файле error.log');

promptForError(); // Запрашиваем следующую ошибку

});

}

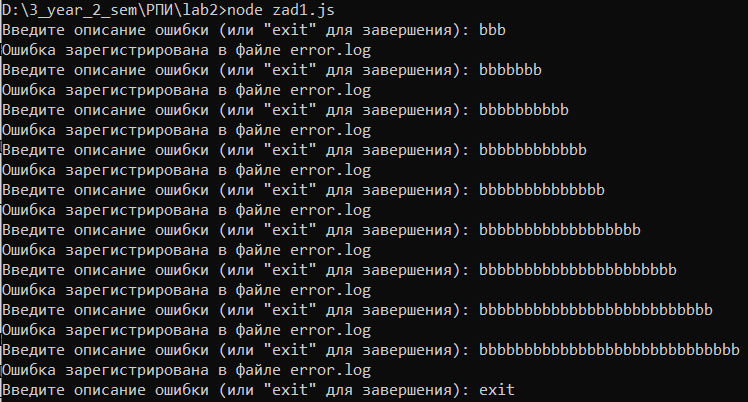
});

}

// Начинаем запрос ошибок

promptForError();

**Результат выполения:**

****

**Задание 2:** Создать массив объектов на основе класса в соответствии со своим вариантом. Записать этот массив в формате JSON в файл.

|  |  |  |
| --- | --- | --- |
| Вариант | Описание | Свойство |
| 10 | Видеокарта | Интерфейс  Производитель графического процессора  Графический процессор  Частота графического процессора  Видеопамять (объем) |

**Код программы:**

const fs = require('fs');

class GraphicsCard {

constructor(interfaceType, manufacturer, gpu, gpuFrequency, vram) {

this.interfaceType = interfaceType;

this.manufacturer = manufacturer;

this.gpu = gpu;

this.gpuFrequency = gpuFrequency;

this.vram = vram;

}

}

// Создаем несколько объектов графических карт

const graphicsCards = [

new GraphicsCard("PCIe 4.0", "NVIDIA", "RTX 3080", "1.71 GHz", "10 GB"),

new GraphicsCard("PCIe 4.0", "AMD", "Radeon RX 6900 XT", "2.25 GHz", "16 GB"),

new GraphicsCard("PCIe 4.0", "NVIDIA", "RTX 3070", "1.73 GHz", "8 GB")

];

// Преобразуем массив объектов в формат JSON

const jsonData = JSON.stringify(graphicsCards, null, 2);

// Записываем JSON данные в файл

fs.writeFile('graphics\_cards.json', jsonData, { flag: 'wx' }, (err) => {

if (err) {

if (err.code === 'EEXIST') {

console.error('Файл уже существует, данные не были записаны.');

return;

}

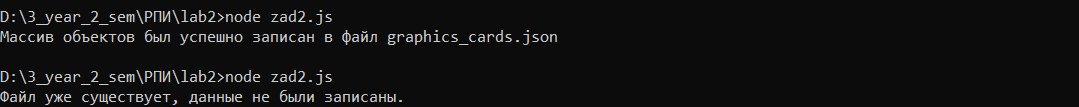
throw err;

}

console.log('Массив объектов был успешно записан в файл graphics\_cards.json');

});

**Результат выполнения:**



**Задание 3.**

Произвести считывание данных из созданного файла в новый массив. Вывести массив в консоль.

**Код программы:**

const fs = require('fs');

// Считываем данные из файла

fs.readFile('graphics\_cards.json', 'utf8', (err, data) => {

if (err) {

console.error('Ошибка чтения файла:', err);

return;

}

// Преобразуем данные из JSON в объекты JavaScript

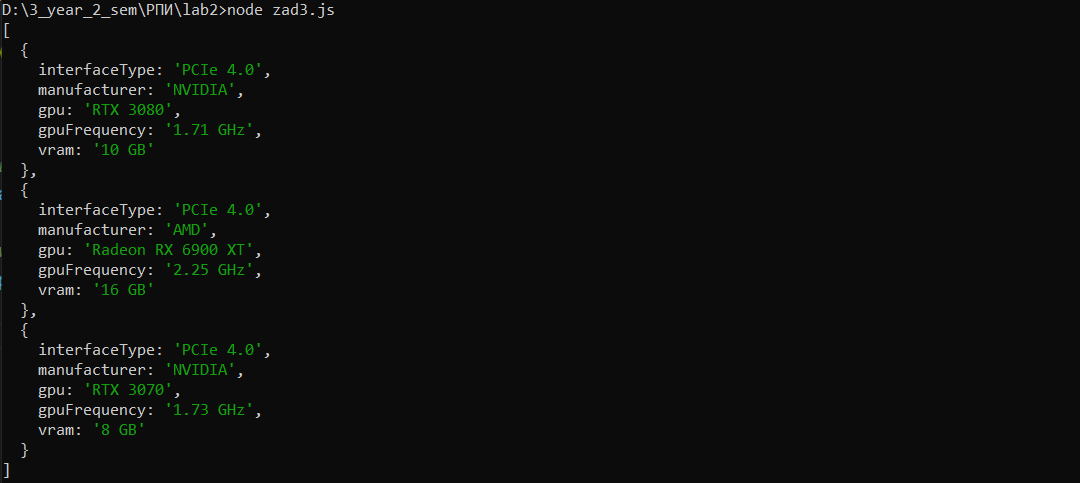
const graphicsCards = JSON.parse(data);

// Выводим массив в консоль

console.log(graphicsCards);

});

**Результат выполнения:**



**Задание 4:** Создать в папке проекта две папки src и dist. В папку src скопировать исходные файлы приложений (\*.js). В папку dist скопировать созданные файлы.

**Код программы:**

const fs = require('fs');

const path = require('path');

// Пути к исходной папке lab2 и папкам src и dist

const lab2Dir = './';

const srcDir = './src';

const distDir = './dist';

// Создаем папки src и dist, если они не существуют

if (!fs.existsSync(srcDir)) {

fs.mkdirSync(srcDir);

console.log('Папка src была успешно создана.');

}

if (!fs.existsSync(distDir)) {

fs.mkdirSync(distDir);

console.log('Папка dist была успешно создана.');

}

// Читаем содержимое папки lab2

fs.readdir(lab2Dir, (err, files) => {

if (err) {

console.error('Ошибка чтения директории lab2:', err);

return;

}

// Перебираем файлы в папке lab2

files.forEach(file => {

const srcFilePath = path.join(lab2Dir, file);

// Определяем расширение файла

const ext = path.extname(file);

// Копируем файлы с расширением .js в папку src

if (ext === '.js') {

const distFilePath = path.join(srcDir, file);

fs.copyFile(srcFilePath, distFilePath, (err) => {

if (err) {

console.error(`Ошибка копирования файла ${srcFilePath} в ${distFilePath}:`, err);

return;

}

console.log(`Файл ${srcFilePath} успешно скопирован в ${distFilePath}`);

});

}

// Копируем файлы с расширениями .json и .log в папку dist

else if (ext === '.json' || ext === '.log') {

const distFilePath = path.join(distDir, file);

fs.copyFile(srcFilePath, distFilePath, (err) => {

if (err) {

console.error(`Ошибка копирования файла ${srcFilePath} в ${distFilePath}:`, err);

return;

}

console.log(`Файл ${srcFilePath} успешно скопирован в ${distFilePath}`);

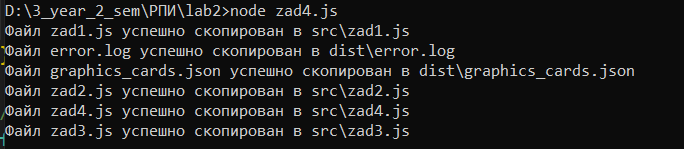
});

}

});

});

**Результат выполнения:**



**Вывод:** в результате выполнения лабораторной работы мноюизучены основные возможности по работе с файловой системой и потоками в Node.js.