**Практическая работа**

*Задание 1.*

Цель задания: написать Node.js скрипт, который читает содержимое файла input.txt построчно и выводит каждую строку в консоль, добавляя перед каждой строкой её номер.

1. Создаем файл input.txt со следующим содержимым. (Рисунок 1)

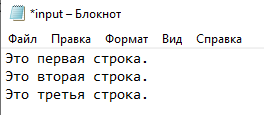


Рисунок 1 – Создание текстового файла

1. Создаем скрипт task1.js, в котором реализуется следующий код (Рисунок 2). В этом коде используется модель fs для чтения файла, далее содержимое файла разбивается на строки с помощью метода split('\n'), а после каждая строка выводится в консоль с её номером.

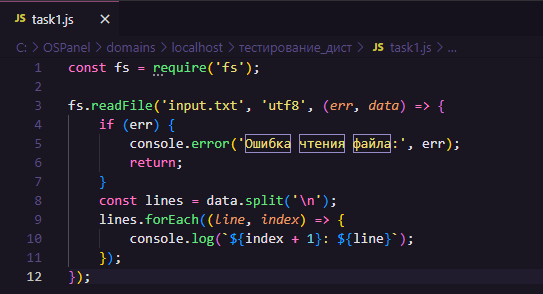


Рисунок 2 – Скрипт

1. Далее запускаем скрипт командой «node task1.js», предварительно перейдя в путь, где лежит данный файл с помощью «cd».
2. В результате выполнения скрипта в консоль был выведен следующий результат. (Рисунок 3)

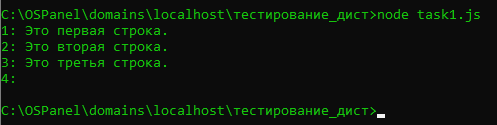


Рисунок 3 – Вывод скрипта

*Задание 2.*

Цель задания: написать Node.js скрипт, который читает содержимое файла data.csv, парсит его и выводит в консоль список пользователей, отсортированный по возрасту (по возрастанию).

1. Создаем файл data.csv с содержимым. (Рисунок 4)

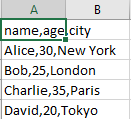


Рисунок 4 – Создание файла data.csv

1. Был создан файл task2.js, в котором реализован следующий код (Рисунок 5). В данном скрипте используется модуль fs для чтения файла, содержимое файла разбивается на строки с помощью метода split('\n'), каждая строка (кроме заголовка) разбивается на значения с помощью split(','), после чего данные преобразуются в массив объектов, где каждый объект содержит поля name, age и city. Массив объектов сортируется по возрасту с помощью метода sort.

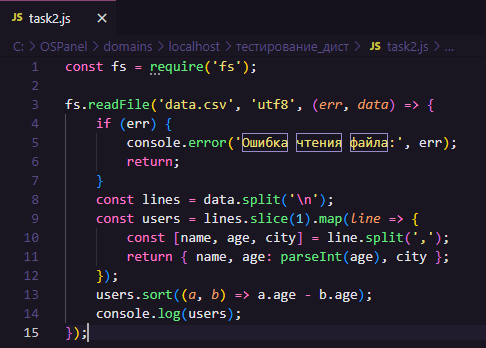


Рисунок 5 – Скрипт

1. Запуск скрипта. В процессе выполнения в консоль был выведен следующий результат. (Рисунок 6)

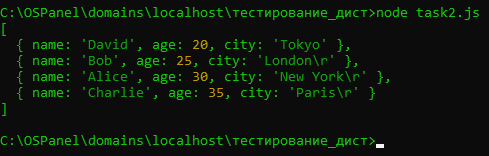


Рисунок 6 – Запуск скрипта

*Задание 3.*

Цель задания: написать Node.js скрипт, который читает файл log.txt, парсит его, подсчитывает количество логов каждого типа (INFO, WARNING, ERROR), находит все записи логов типа ERROR и выводит их в формате: [Дата и время] Сообщение.

1. Был создан файл log.txt с неким содержимым. (Рисунок 7)

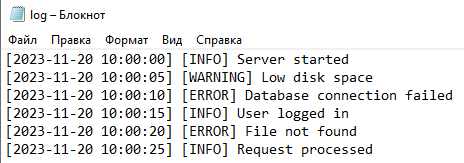


Рисунок 7 – Создание текстового файла

1. Был создан файл task3.js в котором реализован следующий код (Рисунок 8). В данном коде используется модуль fs для чтения файла, содержимое файла разбивается на строки с помощью метода split('\n'), каждая строка парсится с помощью регулярного выражения для извлечения даты, типа и сообщения, подсчитывается кол-во логов каждого типа. Все ошибки (тип ERROR) сохраняются в отдельный массив и выводится в консоль.

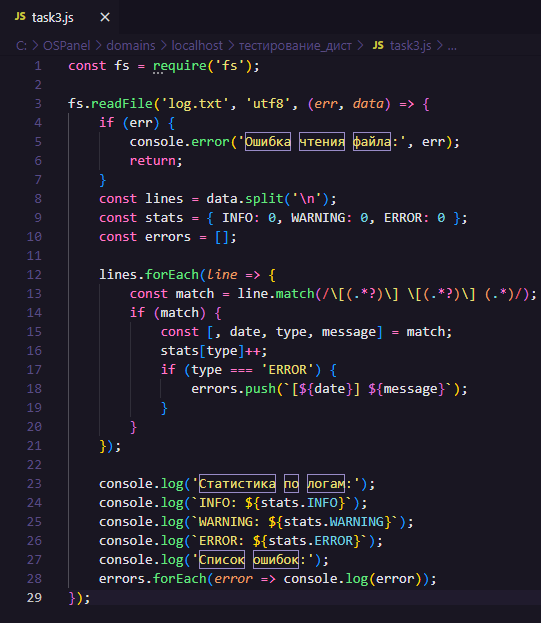


Рисунок 8 – Скрипт

1. Запускаем скрипт также через консоль и «node task3.js». В результате выполнения скрипта в консоль был выведен следующий результат. (Рисунок 9)

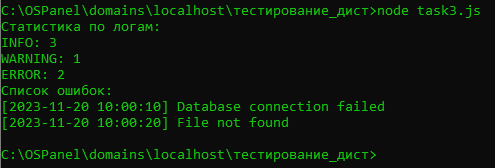


Рисунок 9 – Запуск скрипта

Вывод: в ходе выполнения данной работы были успешно реализованы три задания, связанные с обработкой текстовых файлов и данных с использованием Node.js. Каждое задание было направлено на развитие навыков работы с файловой системой, обработкой строк, парсингом данных и анализом информации.