



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»

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №4**  
**По курсу: "Архитектура ЭВМ"**

Студент \_\_\_\_\_ Сукочева Алис

Группа \_\_\_\_\_ ИУ7-53Б

Название предприятия \_\_\_\_\_ МГТУ им. Н. Э. Баумана, каф. ИУ7

Тема Взаимодействие серверов. Дочерние процессы. Аргументы командной строки

Студент: \_\_\_\_\_ Сукочева А.

подпись, дата \_\_\_\_\_ Фамилия, И.О.

Преподаватель: \_\_\_\_\_ Попов А. Ю.

подпись, дата \_\_\_\_\_ Фамилия, И. О.

## TASK\_1.

**Цель работы:**

—

### **Задание 1**

Листинг 1 — Код программы. TASK\_1. Главнвая функция main

---

Листинг 2 — Код программы. TASK\_1. Реализация заданий

---

**Вывод:**

— Были изучены

**Пример работы:**

## TASK\_2.

### Цель работы:

- Изучать и реализовать взаимодействие между серверами.
- Изучать и реализовать дочерние процессы.
- Изучать и реализовать `process.argv`.

### Задание 1

Создать сервер А. На стороне сервера хранится файл с содержимым в формате JSON. При получении запроса на `/insert/record` идёт добавление записи в файл. При получении запроса на `/select/record` идёт получение записи из файла. Каждая запись хранит информацию о машине (название и стоимость).

Создать сервер Б. На стороне сервера хранится файл с содержимым в формате JSON. Каждая запись в файле хранит информацию о складе и массиве машин, находящихся на данном складе. То есть каждая запись хранит в себе название склада (строку) и массив названий машин (массив строк). При получении запроса на `/insert/record` идёт добавление записи в файл. При получении запроса на `/select/record` идёт получение записи из файла.

Создать сервер С. Сервер выдаёт пользователю страницы с формами для ввода информации. При этом сервер взаимодействует с серверами А и Б. Реализовать для пользователя функции:

создание нового типа машины получение информации о стоимости машины по её типу создание нового склада с находящимися в нём машинами получение информации о машинах на складе по названию склада Реализовать удобный для пользователя интерфейс взаимодействия с системой (использовать поля ввода и кнопки).

### Задание 2

Написать скрипт, который принимает на вход число и считает его факториал. Скрипт должен получать параметр через `process.argv`.

Написать скрипт, который принимает на вход массив чисел и выводит на экран факториал каждого числа из массива. Скрипт принимает параметры через `process.argv`.

При решении задачи вызывать скрипт вычисления факториала через `execSync`.

Листинг 3 — Код программы. TASK\_2. Главный процесс

```
1  "use strict";
2
3  // импортируем библиотеку
4  const execSync = require('child_process').execSync;
5
6  const MY_ARG = 2
```

```

7   const OPTIONS = { encoding: 'utf8' };
8
9   // Функция, для считывания аргументов
10  // Переданных в командной строке.
11  function readArgv(array) {
12      let i = MY_ARG;
13
14      while (process.argv[i])
15          array.push(parseInt(process.argv[i++]));
16
17      return array;
18  }
19
20  // Функция, вызывающая дочерний процесс
21  // Для каждого элемента из массива array.
22  // Дочерний процесс в свою очередь
23  // Считает факториал числа.
24  function arrayFactorial(array) {
25      let cmd;
26
27      for (let i in array) {
28          cmd = 'node factorial ${array[i]}';
29          console.log(execSync(cmd, OPTIONS))
30      }
31  }
32
33
34  function main() {
35      let array = [];
36      readArgv(array);
37      arrayFactorial(array);
38  }
39
40  main();

```

Листинг 4 — Код программы. TASK\_2. Дочерний процесс.

```

1  "use strict";
2
3  // Функция, которая вычисляет факториал
4  // Числа, переданного аргументом командной строки.
5  function factorial() {
6      let num = parseInt(process.argv[2]);
7      let result = 1;
8
9      for (let i = 1; i < num; i++)
10         result *= i;

```

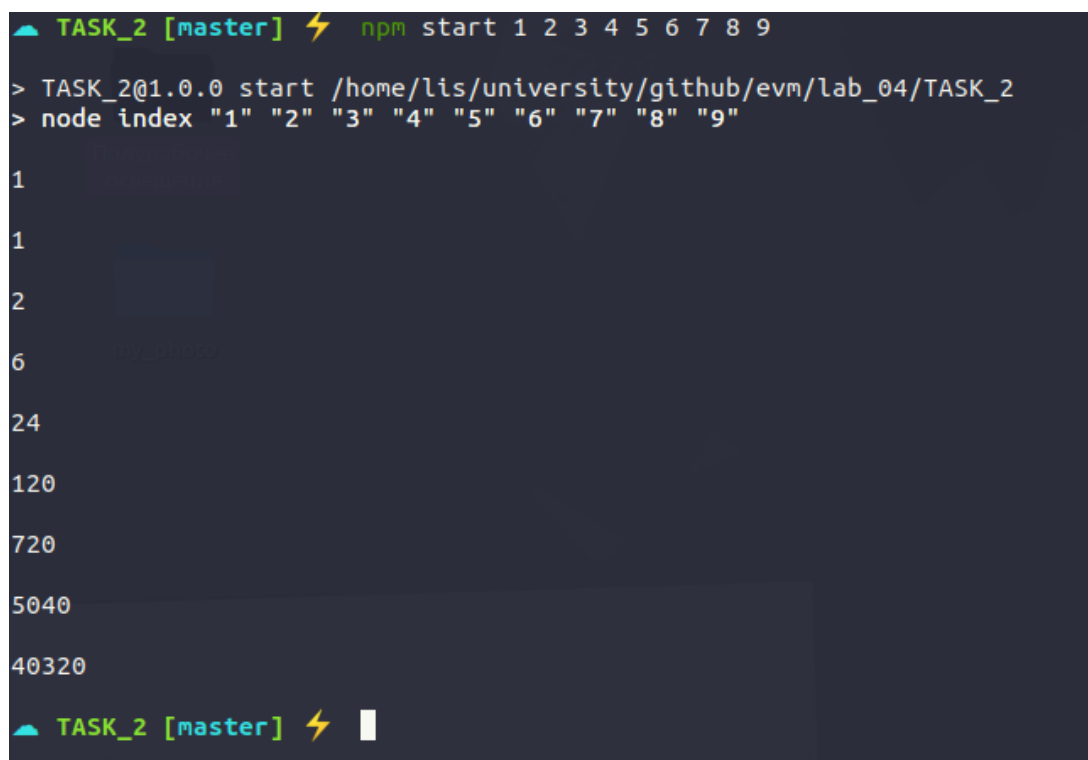
```
11
12     console.log(result);
13 }
14
15 factorial();
```

Листинг 5 — Код программы. TASK\_2. Главный файл

### Вывод:

- Мы изучили и реализовали взаимодействие между серверами.
- Изучили и реализовали дочерние процессы.
- Изучили и реализовали `process.argv`.

### Пример работы:



```
TASK_2 [master] ⚡ npm start 1 2 3 4 5 6 7 8 9
> TASK_2@1.0.0 start /home/lis/university/github/evm/lab_04/TASK_2
> node index "1" "2" "3" "4" "5" "6" "7" "8" "9"
1
1
2
6
24
120
720
5040
40320
TASK_2 [master] ⚡
```

Рисунок 0.1 — Пример работы программы