Лабораторная работа №1

Задача 1.

Создадим скрипт в файл name.js Запустим через консоль командой
> node name.js
Убедимся что при вызове сервера через браузер localhost:8080 в консоль выводится
"HTTP works!"

Скрипт:

```
const http = require('http'); // подключение модуля

const server = http.createServer((request, response) => { // вызов метода создания http сервера
	console.log("HTTP works!");
});

server.listen(8080);
```

Задача 2.

Добавим в скрипт возврат ответа

```
const http = require('http'); // подключение модуля

http.createServer((request, response) => { // вызов метода создания http сервера
	console.log("HTTP works!");
	response.writeHead(200, {'Content-Type':'text/html'});
	response.write('<h1>Hello</h1>');
	response.end();
}).listen(8080);
```

response.writeHead(statusCode[, statusMessage][, headers]) — отсылает заголовки ответа на запрос

- **status code** — HTTP три цифры кода к примеру 404.
- **statusMessage** — опциональный параметр задается программистом
- **headers** - возвращаемые заголовки.

response.end() - Это метод сообщает серверу что все заголовки и тело были высланы. Этот метод должен вызываться на каждый ответ. В нем также можно возвращать данные.
response.write() - пересылка порции данных

1. Запустите сервер. Перешлите клиенту свою простую страницу.
2. Отдайте пользователю ошибку 404.

Задача 3.

Отдадим станицу через чтение файла.

Станица в файле index.html

```
<html>
	<head>
		<title>Node-page</title>
```

```
	</head>
	<body>
		<h1>Просто страница</h1>
	</body>
</html>
```

Скрипт сервера app.js

```
const http = require('http'); // подключение модуля http
const fs = require('fs'); // подключение модуля для работы с файлом
const filename = "index.html";

http.createServer((request, response) => {// вызов метода создания http сервера
	fs.readFile(filename, 'utf8', (err, data) => {
		if (err) {
			console.log('Could not find or open file for reading\n');
			response.statusCode = 404;
			response.end();
		} else {
			console.log(`The file ${filename} is read and sent to the client\n`);
			response.writeHead(200, {'Content-Type':'text/html'});
			response.end(data);
		}
	});

	console.log("Request accepted!");
}).listen(8080, ()=>{
	console.log("HTTP server works in 8080 port!\n");
});
```

fs.readFile(path[, options], callback) – асинхронное чтение файла, как только файл будет прочитал вызывается функция callback.

- **path** – строка имя файла (может включать в себя путь к файлу);
- **options** – необязательный параметр, может задаваться в виде строки, тогда он задаёт кодировку файла и прочитанный из файла информация будет представлять собой строку.
- **callback(err, data)** – функция обратного вызова, ей передаются два аргумента:
  - **err** – ошибка чтения файла;
  - **data** – прочитанное содержимое файла, если задана кодировка, то это строка.

Тут мы сначала подключаем еще один важный Node-модуль, отвечающий за работу с файловой системой. Затем при обработке запроса читаем файл index.html в нужной кодировке и записываем его в ответ сервера. Теперь при открытии браузером адреса http://127.0.0.1:8080 можем любоваться вашей веб-страницей.

1. Сделайте три файла header.html , body.html, footer.html с простой html верстkой и отдайте контент за один вызов сервера.