**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 №2

**По курсу: "Архитектура ЭВМ"**

Студент Сукочева Алис
Группа ИУ7-53Б
Название предприятия МГТУ им. Н. Э. Баумана, каф. ИУ7
Тема Работа с fs, readline-sync и express.

Студент: _____ Сукочева А.
подпись, дата Фамилия, И.О.
Преподаватель: _____ Попов А. Ю.
подпись, дата Фамилия, И. О.

Москва — 2020 г.

# TASK_1.

**Цель работы:**

— Изучить fs, readline-sync и express;
— Написать программы, для демонстрации изученного материала;
— Научиться взаимодействовать с пользователем через консоль;
— Изучить и реализовать работу с файлами;
— Изучить формат JSON и научиться работать с ним.

**Задание 1**

С клавиатуры считывается число N. Далее считывается N строк. Необходимо создать массив и сохранять в него строки только с четной длинной. Получившийся массив необходимо преобразовать в строку JSON и сохранить в файл.

**Задание 2**

Необходимо считать содержимое файла, в котором хранится массив строк в формате JSON. Нужно вывести только те строки на экран, в которых содержатся только гласные буквы.

**Задание 3**

С клавиатуры считывается строка - название расширения файлов. Далее считывается строка - адрес папки. Необходимо перебрать все файлы в папке и вывести содержимое файлов, у которых расширение совпадает с введенным расширением.

**Задание 4**

Дана вложенная структура файлов и папок. Все файлы имеют раширение "txt". Необходимо рекурсивно перебрать вложенную структуру и вывести имена файлов, у которых содержимое не превышает по длине 10 символов.

**Задание 5**

С клавиатуры считывается число N. Далее считывается N строк - имена текстовых файлов. Необходимо склеить всё содержимое введенных файлов в одну большую строку и сохранить в новый файл.

**Задание 6**

Написать код, который позволяет определить максимальный возможный уровень вложенности друг в друга полей в объекте, чтобы данный объект можно было преобразовать в строку формата JSON. Ответом является целое число.

**Задание 7**

Из файла считывается строка в формате JSON. В этой строке информация об объекте, в котором находится большое количество вложенных друг в друга полей. Объект представляет из себя дерево. Необходимо рекурсивно обработать дерево и найти максимальную вложенность в дереве. Необходимо вывести на экран ветку с максимальной вложенностью.

Листинг 1 — Код программы. TASK_1. Главнвая функция main

```
"use strict";

const readlineSync = require('readline-sync');

const tasks = require("./tasks");
const constants = require("./constants");

function main() {
    console.log(constants.TASKS_TEXT)
    const answer = readlineSync.question(constants.GREEN + "\n\nTask: ");

    if (answer > 7 || answer < 1) {
        console.log(constants.RED + "Error. Bad number.")
        return;
    }

    tasks['task' + answer]()
}

main();
```

Листинг 2 — Код программы. TASK_1. Реализация заданий

```
"use strict";

const readlineSync = require('readline-sync');
const fs = require("fs");
const { deflate } = require('zlib');

function task1() {
    const FILE_NAME = "data/task1.txt";

    const N = readlineSync.question("Input N: ");
    const arr = [];
    let line;

    for (let i = 0; i < N; i++) {
        line = readlineSync.question("Input str: ");
        if (!(line.length & 1))
            arr.push(line);
    }

    // Первый параметр(swedishFamilyObj) - значение, преобразуемое в ст
       року JSON
    // Второй параметр(null) - запрещает замены
```

```
22          // Третий параметр(4) - размер отступов
23          const jsonStr = JSON.stringify(arr, null, 4);
24
25          fs.writeFileSync(FILE_NAME, jsonStr);
26      }
27
28      function countVowels(str) {
29          const vowels = 'aeiou';
30          let count = 0;
31          let arr = str.toLowerCase().split('');
32
33          for (let i = 0; i < arr.length; i++) {
34              if (vowels.indexOf(arr[i]) !== -1) {
35                  count++;
36              }
37          }
38
39          return count;
40      }
41
42      function task2() {
43          const FILE_NAME = "data/task2.txt";
44
45          const contentFile = fs.readFileSync(FILE_NAME, "utf-8");
46          const obj = JSON.parse(contentFile);
47
48          console.log("File:" + contentFile);
49          console.log("Result:");
50          for (let i = 0; i < obj.length; i++) {
51              if (countVowels(obj[i]) === obj[i].length)
52                  console.log(obj[i]);
53          }
54      }
55
56      function task3() {
57          // Расширение файлов.
58          const extension = readlineSync.question("Input extension: ");
59          // Имя папки.
60          const folder = readlineSync.question("Input the folder's name: ");
61
62          let files;
63
64          if (!fs.existsSync(folder)) {
65              console.log("Error!\nThe folder does not exist!");
66              return;
67          }
68
```

```
69          files = fs.readdirSync(folder);
70
71          for (let i = 0; i < files.length; i++) {
72              let file = files[i].split('.');
73              if (file[file.length - 1] === extension) {
74                  let contentFile = fs.readFileSync(folder + "/" + files[i],
                        "utf-8");
75                  console.log(contentFile);
76              }
77          }
78      }
79
80      function recursionTask(folder) {
81          // По заданию сказано, что все файлы в формате txt
82          // Если будут файлы с другим форматом, то сломается программа.
83          // (потому что рекурсивная функция попытается открыть этот файл, та
                к
84          // как будет думать: 'всё что не txt - значит папка').
85          if (!fs.existsSync(folder)) {
86              console.log("Error!\nFolder does not exist!");
87              return;
88          }
89
90          let files = fs.readdirSync(folder);
91          let contentFile;
92
93          for (let i = 0; i < files.length; i++) {
94              let file = files[i].split('.');
95              if (file[file.length - 1] === "txt") {
96                  contentFile = fs.readFileSync(folder + "/" + files[i],
                        "utf-8");
97                  if (contentFile.length < 11) {
98                      console.log("Path: ", folder + "/" + files[i]);
99                  }
100                 // console.log(contentFile, "\n");
101             }
102             else {
103                 recursionTask(folder + "/" + files[i]);
104             }
105         }
106     }
107
108     function task4() {
109         const folder = readlineSync.question("Input the folder's name: ");
110         recursionTask(folder);
111     }
112
```

```
113     function task5() {
114         const FILE_NAME = "data/task5.txt"
115         const N = readlineSync.question("Input N: ");
116         let name;
117 
118         // Очищаем старое содержимое файла (если было)
119         fs.writeFileSync(FILE_NAME, "");
120         for (let i = 0; i < N; i++) {
121             name = readlineSync.question("Input str: ");
122             if (!fs.existsSync(name)) {
123                 console.log("Error!\nThe folder does not exist!");
124                 return;
125             }
126             let contentFile = fs.readFileSync(name, "utf-8");
127             fs.appendFileSync(FILE_NAME, contentFile);
128         }
129     }
130 
131     function task6() {
132         // result: 6978
133         let a = 1;
134         let cnt = 0;
135         try {
136             while (JSON.stringify(a)) {
137                 cnt++;
138                 a = { a };
139             }
140         } catch (err) {
141             console.log(cnt);
142         }
143     }
144 
145 
146     function find_max_branch(obj, data) {
147         if (typeof (obj) !== "object") {
148             return;
149         }
150 
151         if (data.curr_depth > data.max_depth)
152             data.max_depth = data.curr_depth;
153 
154         data.curr_depth++;
155 
156         for (let field in obj) {
157             console.log(field)
158             data.max_branch.push(field)
159             data.max_branch.pop();
```

```
160             find_max_branch(obj[field], data)
161         }
162
163         data.curr_depth--;
164     }
165
166     function task7() {
167         let data = {
168             "max_branch": [],
169             "curr_depth": 0,
170             "max_depth": 0
171         }
172         // data/task7.txt
173         // const file_name = readlineSync.question("Input file name: ");
174         const file_name = "data/task7.txt";
175         const jsonString = fs.readFileSync(file_name, "utf-8");
176
177         console.log("FILE:", jsonString)
178
179         const obj = JSON.parse(jsonString);
180         // console.log(obj);
181
182         find_max_branch(obj, data);
183         console.log(data)
184     }
185
186     module.exports = { task1, task2, task3, task4, task5, task6, task7 };
```

**Вывод:**

— Были изучены fs, readline-sync и express;
— Были написаны программы, для демонстрации изученного материала;
— Было изучено и реализовано взаимодействие с пользователем через консоль;
— Было изучено и реализована работа с файлами;
— Был изучен формат JSON, а также реализована работа с ним.

**Пример работы:**

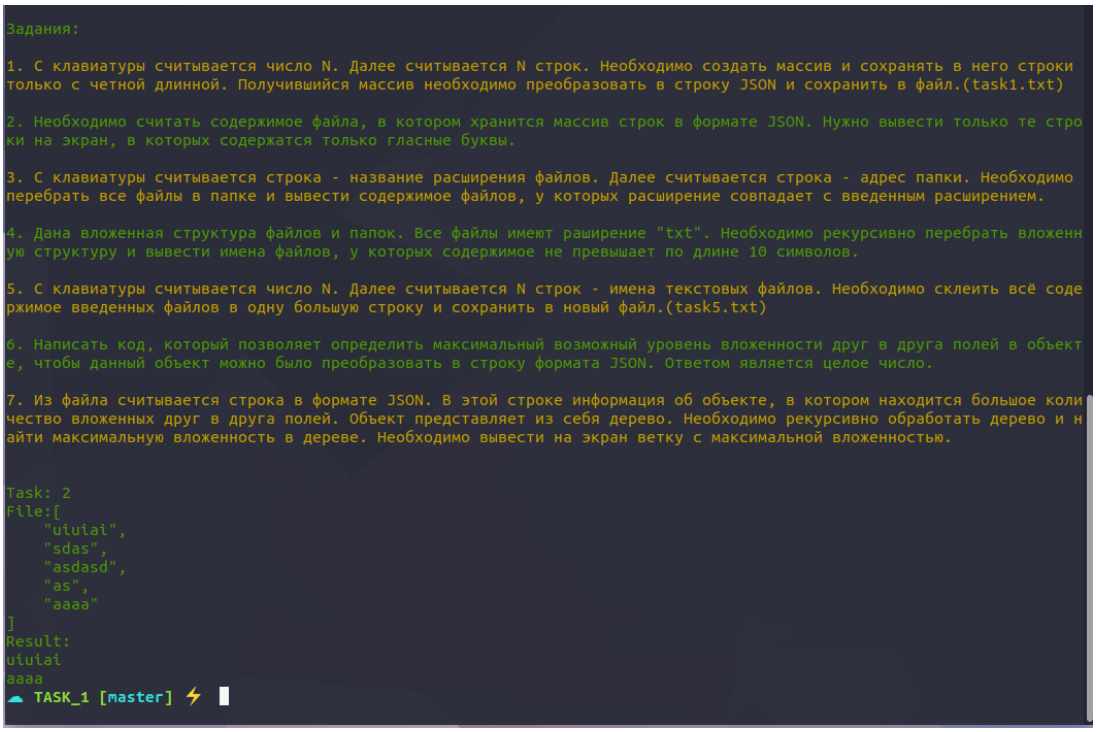


Рисунок 0.1 — Пример работы программы

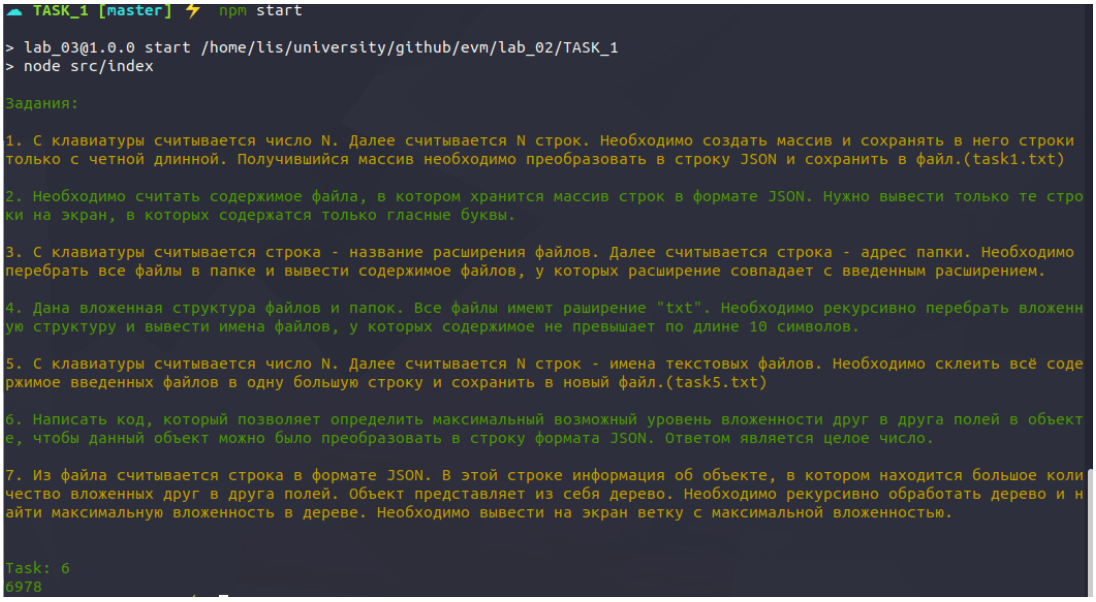


Рисунок 0.2 — Пример работы программы

# TASK_2.

**Цель работы:**

— Научиться запускать собственные сервера;
— Изучить и реализовать хранение данных на стороне сервера;
— Реализовать генерацию HTML страниц;
— Изучить и реализовать взаимодействие с пользователем.

**Задание 1**

Запустить сервер. Реализовать на сервере функцию для сравнения трёх чисел и выдачи наибольшего из них. Реализовать страницу с формой ввода для отправки запроса на сервер.

**Задание 2**

Запустить сервер. На стороне сервера должен храниться файл, внутри которого находится JSON строка. В этой JSON строке хранится информация о массиве объектов. Реализовать на сервере функцию, которая принимает индекс и выдает содержимое ячейки массива по данному индексу. Реализовать страницу с формой ввода для отправки запроса на сервер.

**Задание 3**

Написать программу, которая на вход получает массив названий полей и адрес запроса (куда отправлять). Программа должна генерировать HTML разметку страницы, в которую встроена форма для отправки запроса.

**Задание 4**

Запустить сервер. Реализовать на сервере функцию, которая принимает на вход числа A, B и C. Функция должна выдавать массив целых чисел на отрезке от A до B, которые делятся на C нацело.

Листинг 3 — Код программы. TASK_2. Реализация заданий

```
1   const fs = require("fs");
2
3   const ENCODING = "utf-8"
4
5   const path = require("path");
6
7   function LoadPage(app, path, file_name) {
8       app.get(path, (request, response) => {
9           const fileContent = fs.readFileSync("public/" + file_name,
                ENCODING);
10          response.end(fileContent);
11      });
12  }
13
```

```
14      function task1(app) {
15          LoadPage(app, "/compare", "compare.html");
16
17          app.get("/compare/result", (request, response) => {
18              const a = request.query.a;
19              const b = request.query.b;
20              const c = request.query.c;
21
22              const aInt = parseInt(a);
23              const bInt = parseInt(b);
24              const cInt = parseInt(c);
25
26              if (!aInt || !bInt || !cInt) {
27                  response.end("Input error!");
28                  return;
29              }
30
31              const sInt = Math.max(aInt, bInt, cInt);
32
33              response.end("Maximum number: " + sInt);
34          });
35      }
36
37      function task2(app) {
38          LoadPage(app, "/array_object", "array_object.html");
39
40          app.get("/array_object/result", (request, response) => {
41              const index = request.query.index;
42              const indexInt = parseInt(index);
43              if (!indexInt) {
44                  response.end("Input error!");
45                  return;
46              }
47
48              const PATH = path.join(__dirname, "..", "data", "task2.json");
49
50              const array = JSON.parse(fs.readFileSync(PATH));
51
52              if (indexInt < 0 || indexInt > array.length) {
53                  response.end("Index input error!");
54                  return;
55              }
56
57              response.end("Index = " + indexInt + "\nelement = " +
                    array[indexInt - 1]);
58          });
59      }
```

```
60
61      function task3(app) {
62          LoadPage(app, "/generate_html", "generate_html.html");
63
64          app.get("/generate_html/result", (request, response) => {
65              const field_names = request.query.field_names;
66              const address = request.query.address;
67              const field_names_array = field_names.split(' ');
68
69              const pathBegin = path.join(__dirname, "..", "data",
                    "begin.txt");
70              const pathEnd = path.join(__dirname, "..", "data", "end.txt");
71
72              const fileBegin = fs.readFileSync(pathBegin, ENCODING)
73              const fileEnd = fs.readFileSync(pathEnd, ENCODING)
74
75              let fileContent = `<form method="GET" action="${address}">\n`
76              for (let i = 0; i < field_names_array.length; i++) {
77                  fileContent += `
78      <p>Введите ${field_names_array[i]}</p>\n\
79      <input name="${field_names_array[i]}" spellcheck="false"
            autocomplete="off"></input>`
80              }
81
82              fileContent += '\
83      <p><button type="submit">Отправить</button></p>\n\
84      </form>\n'
85
86              response.end(fileBegin + fileContent + fileEnd);
87          });
88
89      }
90
91      function task4(app) {
92          LoadPage(app, "/number_array", "number_array.html");
93
94          app.get("/number_array/result", (request, response) => {
95              const a = request.query.a;
96              const b = request.query.b;
97              const c = request.query.c;
98
99              const aInt = parseInt(a);
100             const bInt = parseInt(b);
101             const cInt = parseInt(c);
102
103             if (!aInt || !bInt || !cInt) {
104                 response.end("Input error!");
```

```
105                 return;
106             }
107 
108             if (cInt < 0) {
109                 response.end("C must be positive!");
110                 return;
111             }
112 
113             let result = "";
114             for (let num = aInt; num <= bInt; num++) {
115                 if (!(num % cInt)) {
116                     result = result + num + " ";
117                 }
118             }
119 
120             response.end(result);
121         });
122     }
123 
124 
125     module.exports = { task1, task2, task3, task4, LoadPage }; //, PATH };
```

Листинг 4 — Код программы. TASK_2. Главная страница

```
1   <!DOCTYPE html>
2   <html lang="ru">
3 
4   <head>
5       <meta charset="UTF-8">
6       <link rel="stylesheet" type="text/css" href="../task1.css">
7       <title>Выбор задания</title>
8   </head>
9 
10  <body>
11      <div id="header">
12          <!-- <h1 class="header-title">Выбор задания</h1> -->
13          <br>
14      </div>
15 
16      <div id="main">
17 
18          <h2>Найти наибольшее среди трех чисел</h2>
19          <form method="GET" action="/compare/">
20              <p><button type="submit">Ввести значения</button></p>
21              <!-- <input type="submit" value="Ввести значения"> -->
22          </form>
23 
```

```
24              <h2>Получить значение массива по индексу</h2>
25              <form method="GET" action="/array_object/">
26                  <p><button type="submit">Ввести индекс</button></p>
27                  <!-- <input type="submit" value="Ввести индекс"> -->
28              </form>
29
30              <h2>Получить разметку</h2>
31              <form method="GET" action="/generate_html/">
32                  <p><button type="submit">Ввести массив названий полей и адр
                        ес запроса</button></p>
33                  <!-- <input type="submit" value="Ввести массив названий пол
                        ей и адрес запроса "> -->
34              </form>
35
36              <h2>Получить массив чисел</h2>
37              <form method="GET" action="/number_array/">
38                  <p><button type="submit">Ввести числа A, B и C</button></p>
39                  <!-- <input type="submit" value="Ввести числа A, B и C">
                        -->
40              </form>
41
42          </div>
43      </body>
44
45      </html>
```

Листинг 5 — Код программы. TASK_2. Страница с заданием 1

```
1       <!DOCTYPE html>
2   <!-- <html lang="ru"> -->
3
4   <head>
5       <meta charset="UTF-8">
6       <link rel="stylesheet" type="text/css" href="../task1.css">
7       <title>Максимум</title>
8   </head>
9
10  <body>
11      <div id="header">
12          <!-- <h1 class="header-title">Максимальное число!</h1> -->
13      </div>
14
15      <div id="main">
16
17          <h2>Найти наибольшее среди трех чисел</h2>
18          <form method="GET" action="/compare/result">
19              <p>Введите A</p>
```

```
            <input name="a" spellcheck="false" autocomplete="off">
            <p>Введите B</p>
            <input name="b" spellcheck="false" autocomplete="off">
            <p>Введите C</p>
            <input name="c" spellcheck="false" autocomplete="off">
            <br>
            <br>
            <p><button type="submit">Найти наибольшее</button></p>
            <!-- <input type="submit" value="Найти наибольшее"> -->
        </form>

    </div>
</body>

</html>
```

Листинг 6 — Код программы. TASK_2. Страница с заданием 2

```
    <!DOCTYPE html>
<!-- <html lang="ru"> -->

<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" type="text/css" href="../task1.css">
    <title>Элемент массива</title>
</head>

<body>
    <div id="header">
        <!-- <h1 class="header-title">Элемент массива!</h1> -->
    </div>

    <div id="main">
        <h2>Получить значение массива по индексу</h2>
        <form method="GET" action="/array_object/result">
            <p>Индекс</p>
            <input name="index" spellcheck="false" autocomplete="off">
            <br>
            <br>
            <p><button type="submit">Получить</button></p>
            <!-- <input type="submit" value="Получить"> -->
        </form>

    </div>
</body>

</html>
```

Листинг 7 — Код программы. TASK_2. Страница с заданием 3

```
    <!DOCTYPE html>
<!-- <html lang="ru"> -->

<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" type="text/css" href="../task1.css">
    <title>Генерация страницы</title>
</head>

<body>
    <div id="header">
        <!-- <h1 class="header-title">Генерация разметки!</h1> -->
    </div>

    <div id="main">
        <h2>Получить разметку страницы</h2>
        <form method="GET" action="/generate_html/result">
            <p>Название полей</p>
            <input name="field_names" spellcheck="false"
                autocomplete="off">
            <p>адрес</p>
            <input name="address" spellcheck="false" autocomplete="off">
            <br>
            <br>
            <p><button type="submit">Получить</button></p>
            <!-- <input type="submit" value="Получить"> -->
        </form>

    </div>
</body>

</html>
```

Листинг 8 — Код программы. TASK_2. Страница с заданием 4

```
    <!DOCTYPE html>

<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" type="text/css" href="../task1.css">
    <title>Массив</title>
</head>

<body>
    <div id="header">
        <!-- <h1 class="header-title">Массив!</h1> -->
```

```
    </div>

    <div id="main">
        <h2>Получить массив целых чисел на отрезке от A до B, которые делятся на C нацело.</h2>
        <form method="GET" action="/number_array/result">
            <p>Введите A</p>
            <input name="a" spellcheck="false" autocomplete="off">
            <p>Введите B</p>
            <input name="b" spellcheck="false" autocomplete="off">
            <p>Введите C</p>
            <input name="c" spellcheck="false" autocomplete="off">
            <br>
            <br>
            <p><button type="submit">Получить массив</button></p>
            <!-- <input type="submit" value="Получить массив"> -->
        </form>

    </div>
</body>

</html>
```

Листинг 9 — Код программы. TASK_2. Стили страницы

```
    * {
        box-sizing: border-box;
        color: #FFC0CB;
        text-align: center;
    }
    h2 {
        color: #FFC0CB;

    }

    p {
        font-size: 30px;

    }

    :root {
        --background: #E0FFFF;
        --background-accent: #FFC0CB;
        --background-accent-2: #FFFFFF;
        --light: #FFFFFF;
        --dark: #E0FFFF;
        --text: #C8A2C8
```

```
    }

    body {
          background-color: var(--background);
        background-image: linear-gradient(
            var(--background-accent-2) 50%,
            var(--background-accent) 50%
        ), linear-gradient(
            var(--background-accent) 50%,
            var(--background-accent-2) 50%
        );
        background-repeat: no-repeat;
        background-size: 100% 30px;
        background-position: top left, bottom left;
        /* min-height: 100vh; */
    }

    div {
        display: block;
        width: 400px;
        margin: 0 auto 0 auto;
        position: absolute;
        left: 0;
        right: 0;
        top: 15vh;
     }

    button {
        display: block;
        cursor: pointer;
        outline: none;
        border: none;
        background-color: var(--light);
        width: 400px;
        height: 70px;
        border-radius: 30px;
        font-size: 1.2rem;
        font-weight: 600;
        color: var(--text);
        background-size: 100% 100%;
        box-shadow: 0 0 0 7px var(--light) inset;
        margin-bottom: 15px;

        /* text-shadow: #cad5e2 1px 1px 0, #cad5e2 2px 2px 0,
                #cad5e2 3px 3px 0, #cad5e2 4px 4px 0,
                #cad5e2 5px 5px 0; */
    }
```

```
70
71      button:hover {
72          background-image: linear-gradient(
73              145deg,
74              transparent 10%,
75              var(--dark) 10% 20%,
76              transparent 20% 30%,
77              var(--dark) 30% 40%,
78              transparent 40% 50%,
79              var(--dark) 50% 60%,
80              transparent 60% 70%,
81              var(--dark) 70% 80%,
82              transparent 80% 90%,
83              var(--dark) 90% 100%
84          );
85          animation: background 3s linear infinite;
86      }
87
88      input {
89          /* border: none; */
90          font-size:22px;
91          font-family:Tahoma, Geneva, sans-serif;
92          /* color:#000000;   */
93          margin:0 0 6px 0;
94      }
95
96      @keyframes background {
97        0% {
98          background-position: 0 0;
99        }
100       100% {
101         background-position: 400px 0;
102       }
103     }
```

**Вывод:**

— Мы запустили собственные сервера;

— Мы изучили и реализовали хранение данных на стороне сервера;

— Реализовали генерацию HTML страниц;

— Изучили и реализовали взаимодействие с пользователем.

**Пример работы:**

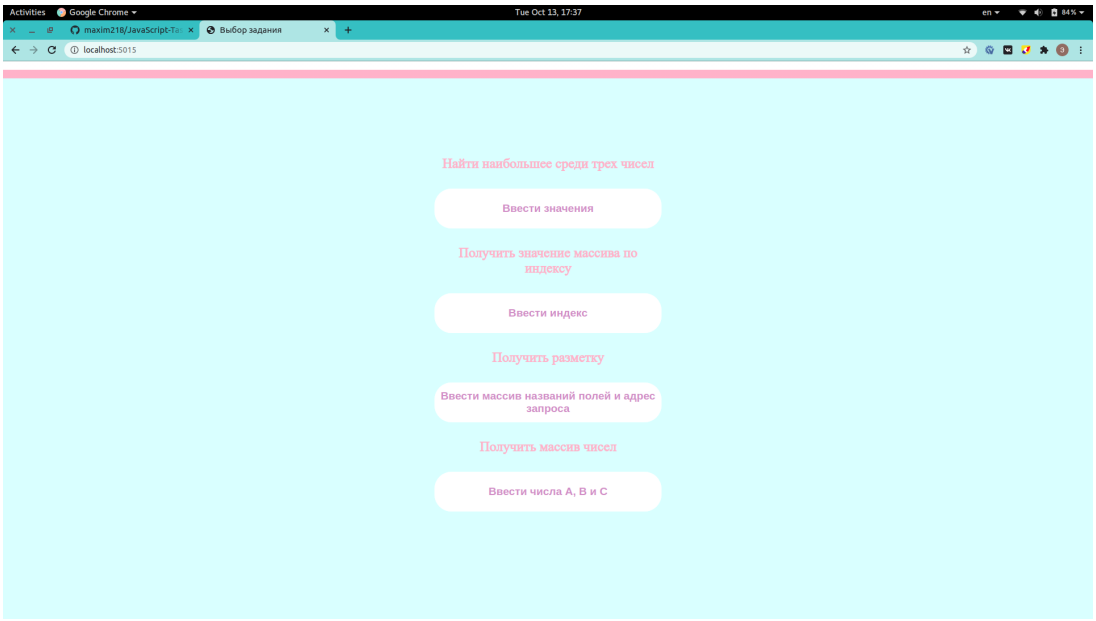


Рисунок 0.1 — Пример работы программы

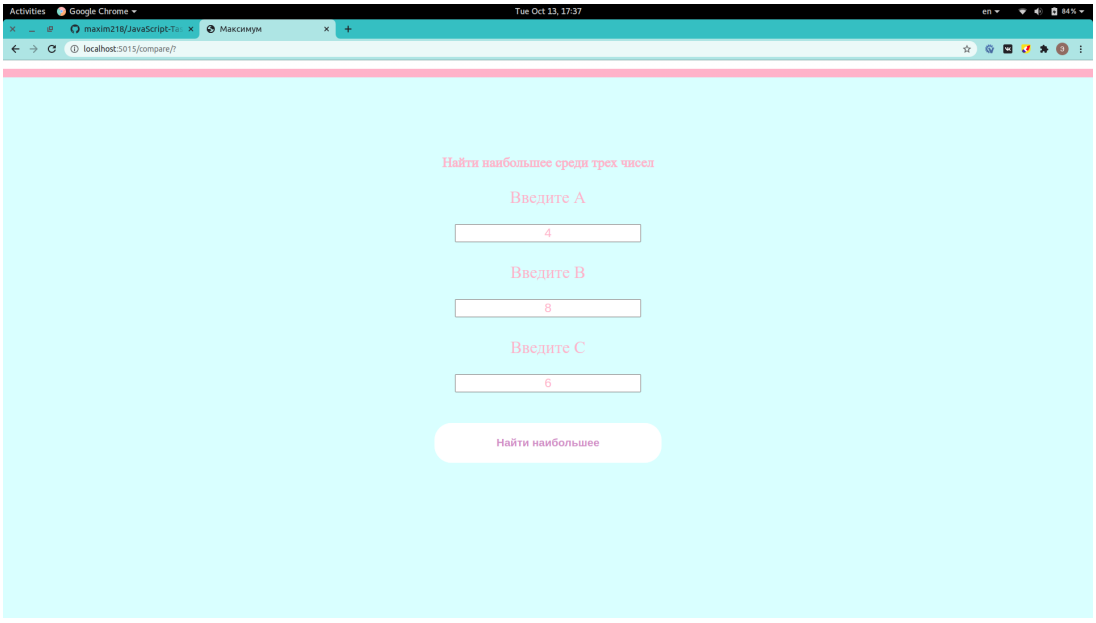


Рисунок 0.2 — Пример работы программы

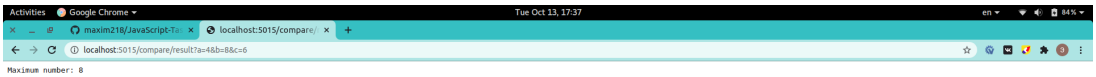


Рисунок 0.3 — Пример работы программы

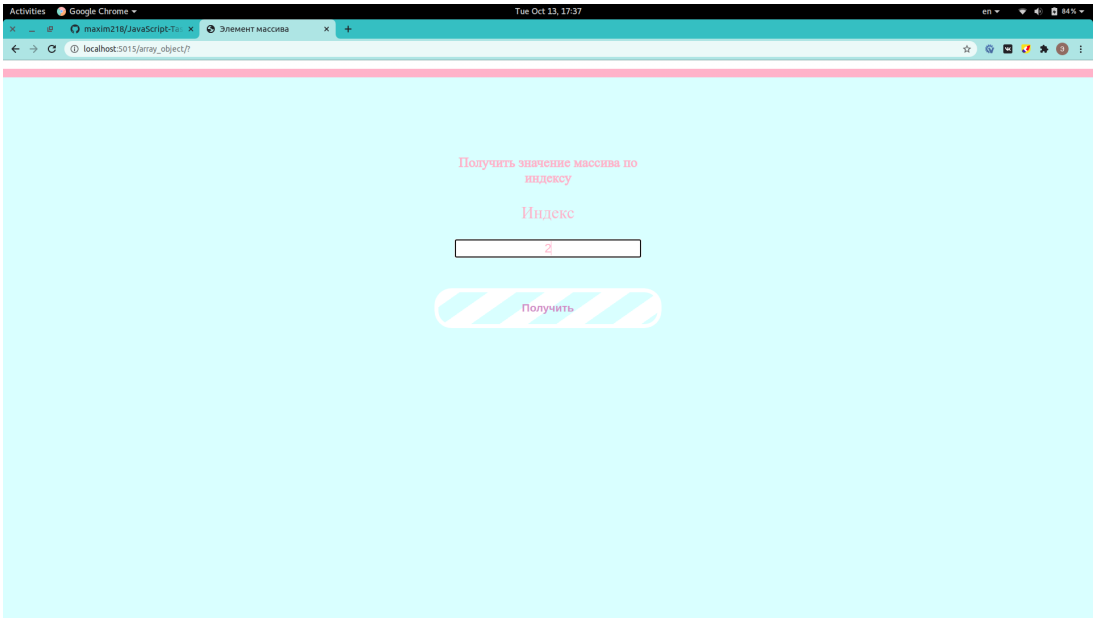


Рисунок 0.4 — Пример работы программы

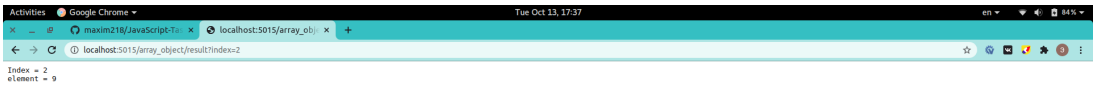


Рисунок 0.5 — Пример работы программы

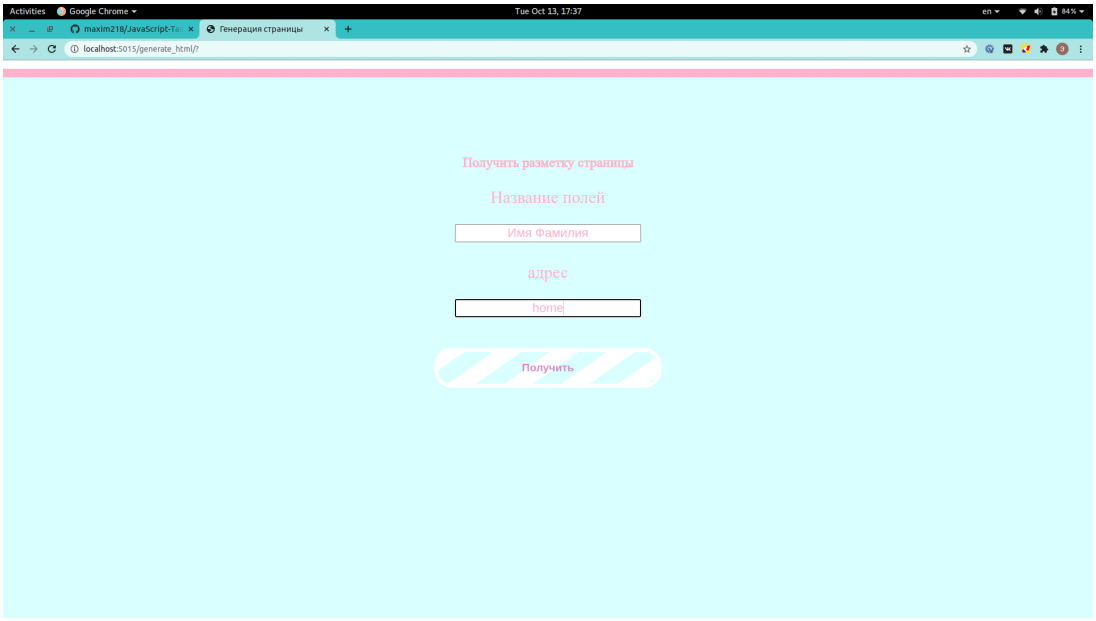


Рисунок 0.6 — Пример работы программы

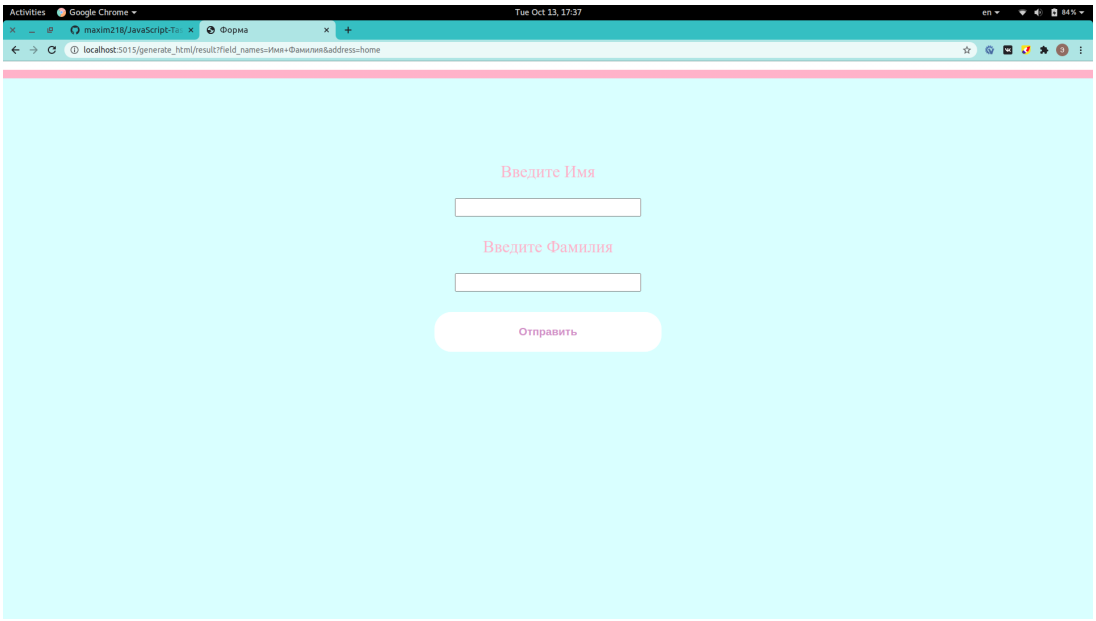


Рисунок 0.7 — Пример работы программы

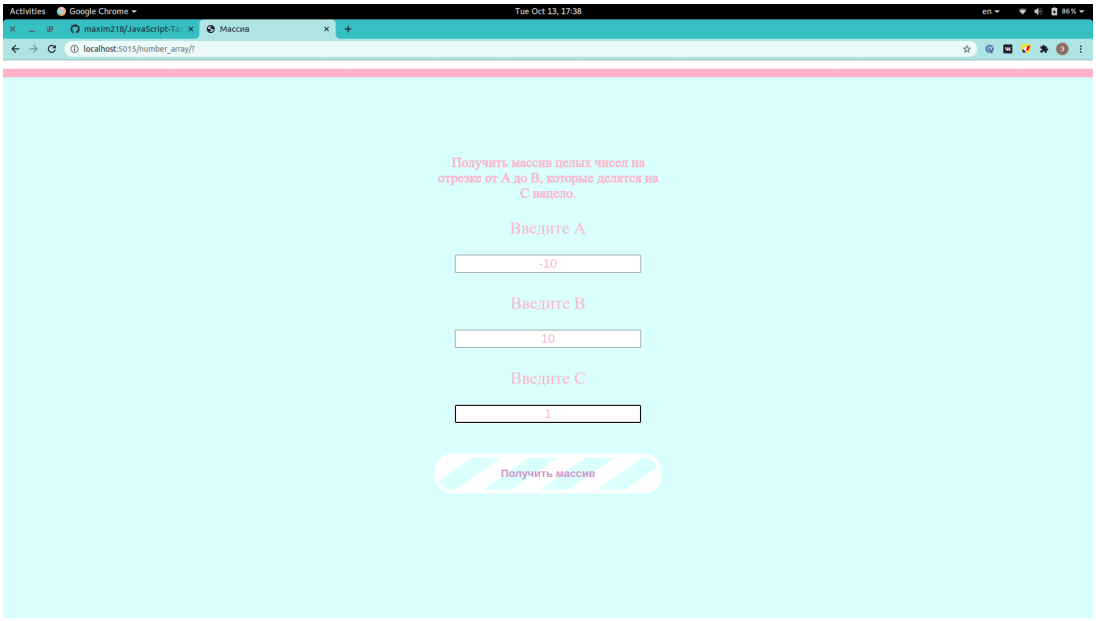


Рисунок 0.8 — Пример работы программы

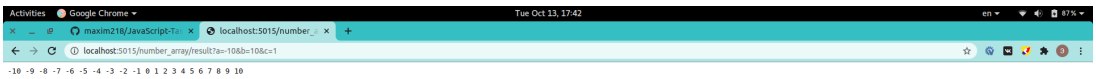


Рисунок 0.9 — Пример работы программы