#### Задание **10**

#### **Использование**

Задание выполняется в малой группе 2 (при нечетном количестве 3) человека.

1 человек отвечает за клиентскую часть кода, 1 (2) человек — за серверную. После реализации должна быть выполнена проверка корректности работы в многопользовательской среде.

В качестве отчета высылать страницу HTML и скрипт для браузера.

Вариант 1

Создать игру, в которой люди голосуют против друг друга и за кого проголосовало больше людей, отстраняется из игры. Если у двух человек одинаковое количество голосов, выбывающий выбирается случайно. Предусмотреть голосование против себя самого.

Вариант 2

Игра "угадай слово". Участники должны первыми угадать слово. Открытые в данный момент буквы слова видно всем. Участники отвечают независимо друг от друга. Кто нажал на букву и не угадал, получает штраф – он не может отвечать 30 секунд, кто угадал – плюс очки и так пока кто-то не угадает слово.

Вариант 3

Играют два человека в крестики-нолики. Играют два человека. Игрок, который ходит первым, выбирается случайно. Игровое поле три на три.

Вариант 4

Игра ассоциации. Люди встают в очередь в том порядке, в котором подключились к игре. Программа генерирует случайное слово. Первый человек видит слово и должен написать не само слово, а любую ассоциацию, нажать на кнопку завершить ответ, после чего этот ответ передается на сервер. Далее сервер передает эту ассоциацию второму в очереди, после чего второй должен угадать это слово и написать новую ассоциацию к нему и т.д. История ассоциаций каждого человека должна быть сохранена сервере и показана в конце игры вместе с исходным словом.

Критерии оценки задания

Написан код HTML для клиента – 2 балла, оценка снижается на 1 балл за каждую ошибку

Написан код JavaScript для сервера - 3 балла, оценка снижается на 1 балл за каждую ошибку

Написан код JavaScript для клиента - 3 балла, оценка снижается на 1 балл за каждую ошибку

Выполнено тестирование — 2 балла, иначе 0 баллов.

Итого 10 баллов

Пример: Чат

Каждую секунду получаем последние введенные сообщений.

При отправке сообщения получаем общее количество сообщений.

## Код для **клиента**: public/index.html

<html>

<body>

<pre id="chat"></pre>

Сообщение: <input id="newmsg" type="text" size="100"><br>

<button id="submit">Отправить</button>

<div id="message"></div>

</body>

<script>

submit.onclick=async function(){

let response = await fetch("/addcomment?newmsg="+encodeURIComponent(newmsg.value));

if (response.ok) {

let data = await response.json();

message.innerText=data.message;

}

};

// каждую секунду получаем сообщения и выводим в chat

setInterval(async function(){

let response = await fetch("/lastcomments");

if (response.ok) {

let data = await response.json();

chat.innerText=data.join('\n');

}

},1000);

</script>

</html>

## Код для сервера: server.js

let express = require('express');

let app = express();

app.use(express.static(\_\_dirname + '/public'));

app.use(express.json());

app.use(express.urlencoded({ extended: false }));

let msg=[]; // храним сообщения в массиве

// обработка запросов

app.get('/addcomment', function(req, res) { // добавить сообщение

msg.push(req.query.newmsg);

res.send({message:msg.length});

});

app.get('/lastcomments', function(req, res) { // последние 10 сообщений

res.send(msg.slice(-10));

});

app.listen(3000, function () {

console.log('Listening on port 3000!');

});