

**Московский государственный технический
университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по домашнему заданию

Выполнил:
студент группы ИУ5-32Б
Кудрявцев Андрей
Подпись и дата:

Проверила:
преподаватель каф. ИУ5
Гапанюк Ю. Е.
Подпись и дата:

Москва, 2023 г.

Постановка задачи

Разработать игру „змейка“ для двух игроков, используя языки «HTML», «CSS» и «JavaScript».

Для создания проекта и знакомства с новыми синтаксическими конструкциями за основу была взята данная статья:

<https://thecode.media/snake-js/>

Текст программы

```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5    <title>Змейка</title>
6    <style>
7      html,
8      body {
9        height: 100%;
10       margin: 0;
11     }
12
13     /*Задаём глобальные параметры*/
14     body {
15       background: black;
16       display: flex;
17       align-items: center;
18       justify-content: center;
19     }
20
21     /*Делаем границу вокруг игрового поля*/
22     canvas {
23       border: 1px solid white;
24     }
25   </style>
26 </head>
27
28 <body>
29   <!-- Рисуем игровое поле -->
30   <canvas width="400" height="400" id="game"></canvas>
31   <!-- Сам скрипт с игрой -->
32   <script>
33     // Поле, на котором всё будет происходить, — тоже как бы переменная
34     var canvas = document.getElementById('game');
35     // Классическая змейка — двумерная, сделаем такую же
36     var context = canvas.getContext('2d');
37     // Размер одной клеточки на поле — 16 пикселей
38     var grid = 16;
39     // Служебная переменная, которая отвечает за скорость змейки
40     var count = 0;
41     // А вот и сама змейка
42     var score1 = 0; // переменная для хранения количества набранных очков
43     var score2 = 0;
44     var snake1 = {
45       // Начальные координаты
46       x: 160,
47       y: 160,
48       // Скорость змейки — в каждом новом кадре змейка смещается по оси X или Y.
49       dx: grid,
50       dy: 0,
51       // Тащим за собой хвост, который пока пустой
52       cells: [],
53       // Стартовая длина змейки — 4 клеточки
54       maxCells: 4
55     };
56
57     var snake2 = {
58
59       x: 320,
60       y: 320,
61
62       dx: -grid,
63       dy: 0,
64
65       cells: [],
66
67       maxCells: 4
68     };
69
```

```

70 // А это — еда. Представим, что это красные яблоки.
71 var apple = {
72     // Начальные координаты яблока
73     x: 320,
74     y: 320
75 };
76 // Делаем генератор случайных чисел в заданном диапазоне
77 function getRandomInt(min, max) {
78     return Math.floor(Math.random() * (max - min)) + min;
79 }
80 // Игровой цикл — основной процесс, внутри которого будет всё происходить
81 function loop() {
82     // Хитрая функция, которая замедляет скорость игры с 60 кадров в секунду до 15 (60/15 = 4)
83     requestAnimationFrame(loop);
84     // Игровой код выполнится только один раз из четырёх, в этом и суть замедления кадров, а пока переменная
85     if (++count < 4) {
86         return;
87     }
88     // Обнуляем переменную скорости
89     count = 0;
90     // Очищаем игровое поле
91     context.clearRect(0, 0, canvas.width, canvas.height);
92     // Двигаем змейку с нужной скоростью
93     snake1.x += snake1.dx;
94     snake1.y += snake1.dy;
95     // Если змейка достигла края поля по горизонтали — продолжаем её движение с противоположной стороны
96     if (snake1.x < 0) {
97         snake1.x = canvas.width - grid;
98     }
99     else if (snake1.x >= canvas.width) {
100         snake1.x = 0;
101     }
102     // Делаем то же самое для движения по вертикали
103     if (snake1.y < 0) {
104         snake1.y = canvas.height - grid;
105     }
106     else if (snake1.y >= canvas.height) {
107         snake1.y = 0;
108     }
109     // Продолжаем двигаться в выбранном направлении. Голова всегда впереди, поэтому добавляем её координаты
110     snake1.cells.unshift({ x: snake1.x, y: snake1.y });
111     // Сразу после этого удаляем последний элемент из массива змейки, потому что она движется и постоянно ос
112     if (snake1.cells.length > snake1.maxCells) {
113         snake1.cells.pop();
114     }
115
116     snake2.x += snake2.dx;
117     snake2.y += snake2.dy;
118
119     if (snake2.x < 0) {
120         snake2.x = canvas.width - grid;
121     }
122     else if (snake2.x >= canvas.width) {
123         snake2.x = 0;
124     }
125
126     if (snake2.y < 0) {
127         snake2.y = canvas.height - grid;
128     }
129     else if (snake2.y >= canvas.height) {
130         snake2.y = 0;
131     }
132
133     snake2.cells.unshift({ x: snake2.x, y: snake2.y });
134
135     if (snake2.cells.length > snake2.maxCells) {
136         snake2.cells.pop();
137     }
138

```

```

139 // Рисуем еду – красное яблоко
140 context.fillStyle = 'red';
141 context.fillRect(apple.x, apple.y, grid - 1, grid - 1);
142 // Одно движение змейки – один новый нарисованный квадратик
143 context.fillStyle = 'green';
144 // Обрабатываем каждый элемент змейки
145 snake1.cells.forEach(function (cell, index) {
146     // Чтобы создать эффект клеточек, делаем зелёные квадратики меньше на один пиксель, чтобы вокруг них
147     context.fillRect(cell.x, cell.y, grid - 1, grid - 1);
148     // Если змейка добралась до яблока...
149     if (cell.x === apple.x && cell.y === apple.y) {
150         // увеличиваем длину змейки
151         snake1.maxCells++;
152         // Рисуем новое яблочко
153         // Помним, что размер холста у нас 400x400, при этом он разбит на ячейки – 25 в каждую сторону
154         apple.x = getRandomInt(0, 25) * grid;
155         apple.y = getRandomInt(0, 25) * grid;
156
157         score1++; // увеличиваем количество набранных очков
158     }
159     // Проверяем, не столкнулась ли змея сама с собой
160     // Для этого перебираем весь массив и смотрим, есть ли у нас в массиве змейки две клетки с одинаковы
161     //-----
162     for (var i = index + 1; i < snake1.cells.length; i++) {
163         // Если такие клетки есть – начинаем игру заново
164         if (cell.x === snake1.cells[i].x && cell.y === snake1.cells[i].y) {
165             // Задаём стартовые параметры основным переменным
166             snake1.x = 160;
167             snake1.y = 160;
168             snake1.cells = [];
169             snake1.maxCells = 4;
170             snake1.dx = grid;
171             snake1.dy = 0;
172
173             score1 = 0; // сбрасываем количество набранных очков
174         }
175
176         if (snake2.x === snake1.cells[i].x && snake2.y === snake1.cells[i].y) {
177             // Задаём стартовые параметры основным переменным
178             snake2.x = 320;
179             snake2.y = 320;
180             snake2.cells = [];
181             snake2.maxCells = 4;
182             snake2.dx = -grid;
183             snake2.dy = 0;
184
185             score2 = 0; // сбрасываем количество набранных очков
186         }
187     }
188 }
189 });
190
191 context.fillStyle = 'blue';
192 snake2.cells.forEach(function (cell, index) {
193
194     context.fillRect(cell.x, cell.y, grid - 1, grid - 1);
195
196     if (cell.x === apple.x && cell.y === apple.y) {
197
198         snake2.maxCells++;
199
200         apple.x = getRandomInt(0, 25) * grid;
201         apple.y = getRandomInt(0, 25) * grid;
202
203         score2++; // увеличиваем количество набранных очков
204     }
205 }
206

```

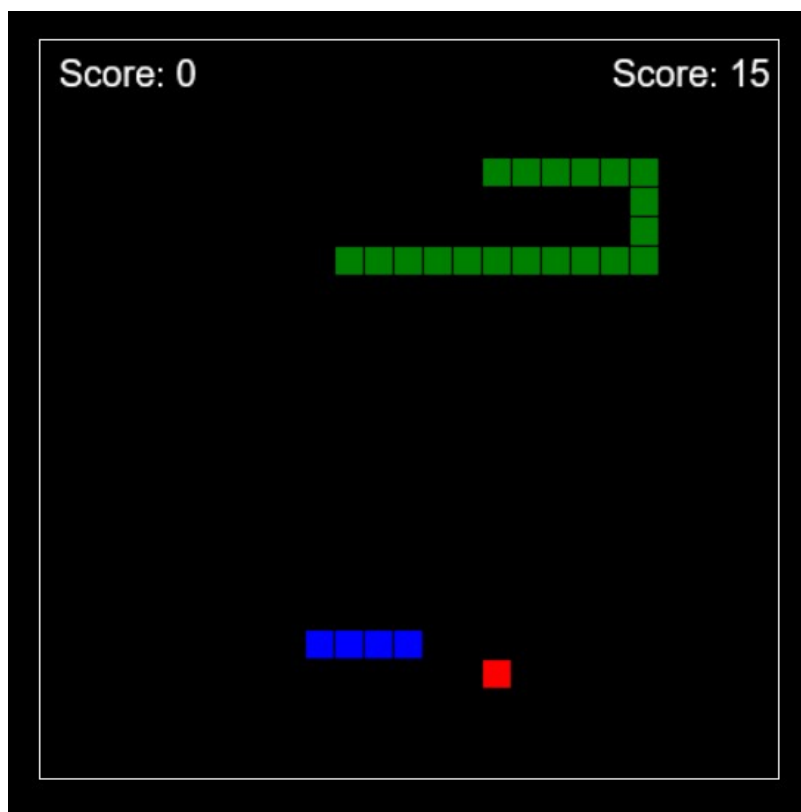
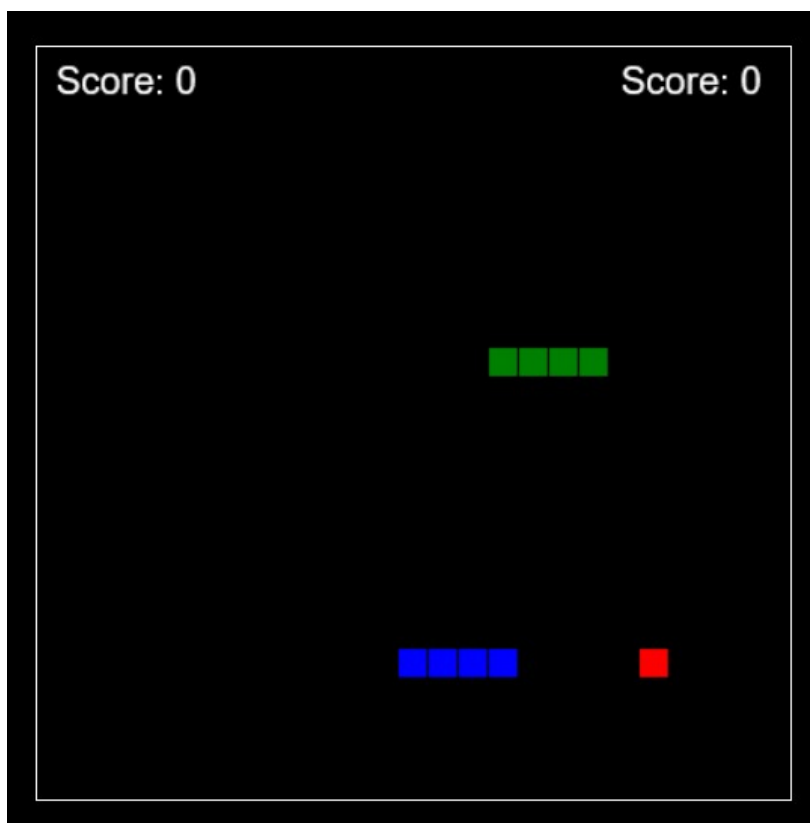
```

206 //-----
207 for (var i = index + 1; i < snake2.cells.length; i++) {
208
209     if (cell.x === snake2.cells[i].x && cell.y === snake2.cells[i].y) {
210
211         snake2.x = 320;
212         snake2.y = 320;
213         snake2.cells = [];
214         snake2.maxCells = 4;
215         snake2.dx = -grid;
216         snake2.dy = 0;
217
218
219
220         score2 = 0; // сбрасываем количество набранных очков
221     }
222
223     if (snake1.x === snake2.cells[i].x && snake1.y === snake2.cells[i].y) {
224         // Задаём стартовые параметры основным переменным
225         snake1.x = 160;
226         snake1.y = 160;
227         snake1.cells = [];
228         snake1.maxCells = 4;
229         snake1.dx = grid;
230         snake1.dy = 0;
231
232         score1 = 0; // сбрасываем количество набранных очков
233     }
234
235 }
236 });
237
238
239
240 // выводим количество набранных очков
241 context.fillStyle = 'white';
242 context.font = '20px Arial';
243 context.fillText('Score: ' + score2, 10, 25);
244
245 context.fillStyle = 'white';
246 context.font = '20px Arial';
247 context.fillText('Score: ' + score1, 310, 25);
248
249 }
250 // Смотрим, какие нажимаются клавиши, и реагируем на них нужным образом
251 document.addEventListener('keydown', function (e) {
252     // Дополнительно проверяем такой момент: если змейка двинется, например, влево, то ещё одно нажа
253     // Стрелка влево
254     // Если нажата стрелка влево, и при этом змейка никуда не двинется по горизонтали...
255     if (e.which === 37 && snake1.dx === 0) {
256         // то даём ей движение по горизонтали, влево, а вертикальное – останавливаем
257         // Та же самая логика будет и в остальных кнопках
258         snake1.dx = -grid;
259         snake1.dy = 0;
260     }
261     // Стрелка вверх
262     else if (e.which === 38 && snake1.dy === 0) {
263         snake1.dy = -grid;
264         snake1.dx = 0;
265     }

```

```
266 // Стрелка вправо
267 else if (e.which === 39 && snake1.dx === 0) {
268     snake1.dx = grid;
269     snake1.dy = 0;
270 }
271 // Стрелка вниз
272 else if (e.which === 40 && snake1.dy === 0) {
273     snake1.dy = grid;
274     snake1.dx = 0;
275 }
276
277 else if (e.which === 65 && snake2.dx === 0) {
278
279     snake2.dx = -grid;
280     snake2.dy = 0;
281 }
282
283 else if (e.which === 87 && snake2.dy === 0) {
284     snake2.dy = -grid;
285     snake2.dx = 0;
286 }
287
288 else if (e.which === 68 && snake2.dx === 0) {
289     snake2.dx = grid;
290     snake2.dy = 0;
291 }
292
293 else if (e.which === 83 && snake2.dy === 0) {
294     snake2.dy = grid;
295     snake2.dx = 0;
296 }
297
298 });
299 // Запускаем игру
300 requestAnimationFrame(loop);
301 </script>
302 </body>
303
304 </html>
```

Результат выполнения программы



Score: 3

Score: 14

