

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
(МИНОБРНАУКИ РОССИИ)

федеральное государственное бюджетное образовательное учреждение
высшего образования
«Рыбинский государственный авиационный технический университет
имени П.А. Соловьёва»
(РГТУ имени П.А. Соловьёва)


Факультет радиоэлектроники и информатики
Кафедра математического и программного обеспечения электронных
вычислительных средств

Курсовая работа

по дисциплине
Тестирование и отладка ПО

Отчет

Студент группы ИПБ - 18 _____ Рыжов К. А.
Руководитель доцент _____ Овсянников Т. С.

06.05.21 

Рыбинск 2021

Содержание

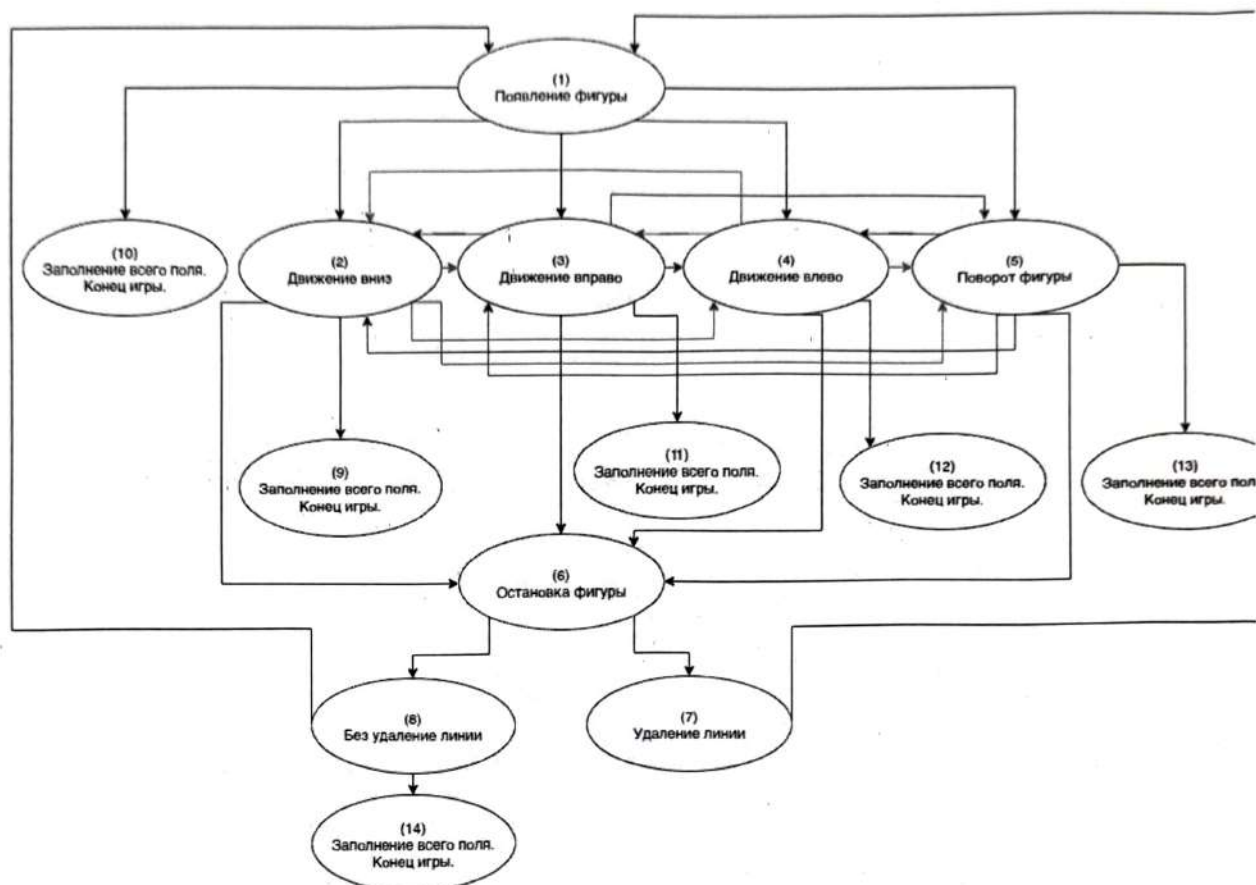
1 Описание тестируемой системы	3
2 Модульное тестирование	4
2.1. Управляющий граф	4
2.2. Тестирование по критериям C0, C1, C2	4
2.3. План модульного тестирования	5
2.4. Результаты модульного тестирования	5
2.5. Вывод по модульному тестированию	8
3 Интеграционное тестирование	9
3.1. План интеграционного тестирования	9
3.2. Результаты интеграционного тестирования	9
3.3. Вывод по интеграционному тестированию	11
4 Системное тестирование	12
4.1. План системного тестирования	12
4.2. Результаты системного тестирования	12
4.3. Вывод по системному тестированию	16
Вывод	17

1 Описание тестируемой системы

В качестве тестируемой системы была взята программа «Тетрис».

2 Модульное тестирование

2.1. Управляющий граф



На рисунке 1 представлен управляющий граф программы.

Рис. 1: Управляющий граф программы

2.2. Тестирование по критериям C0, C1, C2

2.2.1. Критерий C0

Прохождение каждой команды не менее одного раза.

«Появление фигуры» → «Движение вниз» → «Движение вправо» → «Движение влево» → «Поворот фигуры» → «Остановка фигуры» → «Удаление линии»;

1 → 2 → 3 → 4 → 6 → 7;

2.2.2. Критерий C1

Прохождение каждой команды не менее одного раза.

«Появление фигуры» → «Движение вниз» → «Движение вправо» → «Движение влево» → «Поворот фигуры» → «Остановка фигуры» → «Удаление линии»;

1 → 2 → 3 → 4 → 6 → 7;

6.2. Критерий С1

Прохождение каждой ветви не менее одного раза.

«Появление фигуры» → «Движение вниз» → «Движение вправо» → «Движение влево» → «Поворот фигуры» → «Остановка фигуры» → «Без удаления линии» → «Появление фигуры» → «Заполнение всего поля. Конец игры»;

1 → 2 → 3 → 4 → 5 → 6 → 8 → 1 → 10;

2.2.3. Критерий С2

Прохождение каждого пути не менее 1 раза.

«Появление фигуры» → «Движение вниз» → «Движение вправо» → «Движение влево» → «Поворот фигуры» → «Движение вправо» → «Остановка фигуры» → «Без удаления линии» → «Появление фигуры» → «Движение вниз» → «Остановка фигуры» → «Удаление линии» → «Появление фигуры» → «Движение вниз» → «Заполнение всего поля. Конец игры»;

1 → 2 → 3 → 4 → 5 → 3 → 6 → 8 → 1 → 2 → 6 → 7 → 1 → 2 → 9;

2.3. План модульного тестирования

2.3.1. Проверка правильности отрисовки кнопки, написанных на CSS.

2.3.2. Корректность выполнения функции добавления балла.

2.3.3. Проверка правильности выполнения функции проверки движения фигуры.

2.4. Результаты модульного тестирования

2.4.1. Функция отрисовки кнопки, написанных на CSS.

Тестирование функции `.button`. Цвета кнопки отображаются корректно. При наведении мышки кнопка подсвечивается более светлым оттенком. Размеры кнопки соответствуют заданным. Поэтому можно сделать вывод о том, что данная функция работает корректно.

```
122 .button {
123     position: relative;
124     width: 22rem;
125     height: 2.2rem;
126     text-align: center;
127     color: #fff;
128     letter-spacing: 1px;
129     text-decoration: none;
130     line-height: 23px;
131     font-size: 10px;
132     display: block;
133     margin: 30px;
134     text-shadow: -1px -1px 0 #a84155;
135     background: #d25068;
136     border: 1px solid #d25068;
137     width: 12rem;
138     background-image: linear-gradient(to bottom, #f66c7b, #d25068);
139     border-radius: 5px;
140     box-shadow: 0 1px 0 rgba(255, 255, 255, 0.5) inset,
141                0 -1px 0 rgba(255, 255, 255, 0.1) inset, 0 4px 0 #ad4257,
142                0 4px 2px rgba(0, 0, 0, 0.5);
143 }
144
```

2.4.2. Функция добавления балла.

Функция `addScore()` работает корректно. При выполнении условия, когда все клетки заняты на одном уровне, функция добавляет в счетчик `score` (баллы текущей игры) 10 очков и в счетчик `lines` (количество заполненных линий) 1 очко. Отображение и добавление очков отображается верно.

```

239 function addScore() {
240   for (currentIndex = 0; currentIndex < GRID_SIZE; currentIndex += GRID_WIDTH) {
241     const row = [currentIndex, currentIndex + 1, currentIndex + 2, currentIndex + 3, currentIndex + 4, currentIndex + 5, currentIndex + 6,
242       if (row.every(index => squares[index].classList.contains('block2')) {
243         score += 10;
244         lines += 1;
245         scoreDisplay.innerHTML = score;
246         linesDisplay.innerHTML = lines;
247         row.forEach(index => {
248           squares[index].style.backgroundColor = 'none';
249           squares[index].classList.remove('token: block2') || squares[index].classList.remove('token: block')
250         });
251         //splice array
252         const squaresRemoved = squares.splice(currentIndex, width);
253         squares = squaresRemoved.concat(squares);
254         squares.forEach(cell => grid.appendChild(cell));
255       }
256     }
257   }
258 }
259
260 //Styling eventListeners
261 hamburgerBtn.addEventListener('click', (e) => {
262   menu.style.display = 'flex';
263 });
264 span.addEventListener('click', () => {
265   menu.style.display = 'none';
266 });
267
268 })

```

2.4.3. Функция проверки движения фигуры.

Тестирование функций `moveright()` и `moveleft()`. Сами по себе они не объемные, они имеют две проверки. В данной программе и выполнении этих функций возможны только два варианта движения фигуры, либо при движении вправо/лево происходит передвижение, либо игровое поле закончилось и при этом движение вправо/лево невозможно. При задании неверной позиции фигуры функция работает верно, так как условия не срабатывают и фигура продолжает движение.

```

154 //move left and prevent collisions with shapes moving left
155 function moveright() {
156     undraw();
157     const isAtRightEdge = current.some(index => (currentPosition + index) % width === width - 1);
158     if (!isAtRightEdge) currentPosition += 1;
159     if (current.some(index => squares[currentPosition + index].classList.contains('block2'))) {
160         currentPosition -= 1;
161     }
162     draw();
163 }
164
165 //move right and prevent collisions with shapes moving right
166 function moveleft() {
167     undraw();
168     const isAtLeftEdge = current.some(index => (currentPosition + index) % width === 0);
169     if (!isAtLeftEdge) currentPosition -= 1;
170     if (current.some(index => squares[currentPosition + index].classList.contains('block2'))) {
171         currentPosition += 1;
172     }
173     draw();
174 }
175

```

2.5. Вывод по модульному тестированию

По результатам выполнения модульного тестирования были протестированы все сценарии работы программы и оценено его поведение по критериям C0, C1, C2, построен управляющий граф программы. Тестирование по критериям C0-C2 позволяет охватывать все возможные сценарии.

Модульные тесты позволяют достаточно быстро проверить, не привело ли очередное изменение кода к появлению ошибок в уже оттестированных местах программы. После были предоставлены результаты тестирования.

3 Интеграционное тестирование

3.1. План интеграционного тестирования

3.1.1. Проверка функций стилей файла style.css.

3.1.2. Проверка выполнения функций файла app.js.

3.2. Результаты интеграционного тестирования

3.2.1. Функции файла style.css.

В данном файле style.css задаются стили для отображения всех кнопок, прорисовки игрового поля и анимации. Функции верно отображают цвета кнопок, фигур и всей страницы в целом. Тестирование всех функций визуального отображения и стилей было выполнено успешно без нареканий.

При тестировании отдельно запускался файл `style.css`, чтобы проверить корректность отображение всех стилей. На выходе получилось корректно отобразить все заданные цвета, размеры текстов их стилей, а также корректное отображение анимации нажатия кнопки «start/pause».

3.2.2. Функции файла `app.js`.

Тестирование файла `app.js` было выполнено корректно. В данном файле прописаны основные функции игры «Тетрис».

При тестировании отдельно запускался файл `app.js`. Данный модуль программы корректно добавляет баллы в два раздела (баллы текущей игры, количество заполненных линий) при заполнении целой строки фигурами. Также было протестированы действия проверки правильного движения фигуры по игровому полю.

```
1  :root {
2    /* default font size in browsers is 16px = 1em, we make
3       things easier for us and make 10px our base size.
4       We have 10/16 = 0.625 = 1rem as it is set on root element.
5       So 1rem is now 10px throughout our stylesheet.*/
6     font-size: 0.625em;
7  }
8
9  * {
10     box-sizing: border-box;
11  }
12
13  body {
14     font-family: "Montserrat", sans-serif;
15     font-size: 1.6rem;
16     margin: auto;
17     max-width: 60rem;
18     color: #d8edea;
19     background: radial-gradient(
20       circle,
21       rgba(175, 196, 174, 1) 0%,
22       rgba(104, 204, 191, 1) 89%,
23       rgba(94, 191, 178, 1) 100%
24     );
25  }
```


3.3. Вывод по интеграционному тестированию

По результатам выполнения интеграционного тестирования были составлены наборы тестов, которые используют компоненты, уже проверенные с помощью модульного тестирования. Целью интеграционного тестирования является выявление багов при взаимодействии между этими программными модулями и в первую очередь направлено на проверку обмена данными между этими самими модулями.

```
1 document.addEventListener( type: 'DOMContentLoaded', listener: () => {
2   // TODO: we can also get the grid size from user
3   const GRID_WIDTH = 10;
4   const GRID_HEIGHT = 20;
5   const GRID_SIZE = GRID_WIDTH * GRID_HEIGHT;
6
7   // no need to type 200 divs :)
8   const grid = createGrid();
9   let squares = Array.from(grid.querySelectorAll( selectors: 'div' ));
10  const startBtn = document.querySelector( selectors: '.button' );
11  const hamburgerBtn = document.querySelector( selectors: '.toggler' );
12  const menu = document.querySelector( selectors: '.menu' );
13  const span = document.getElementsByClassName( className: 'close' )[0];
14  const scoreDisplay = document.querySelector( selectors: '.score-display' );
15  const linesDisplay = document.querySelector( selectors: '.lines-score' );
16  let currentIndex = 0;
17  let currentRotation = 0;
18  const width = 10;
19  let score = 0;
20  let lines = 0;
21  let timerId;
22  let nextRandom = 0;
23  const colors = [
24    'url(images/blue_block.png)',
25    'url(images/pink_block.png)',
26    'url(images/purple_block.png)',
27    'url(images/peach_block.png)',
28    'url(images/yellow_block.png)'
29  ];
30 }
```

4 Системное тестирование

4.1. План системного тестирования

4.1.1. Выполнение функционального тестирования

В ходе выполнения функционального тестирования будет проверяться удобство и правильность интерфейса, а также выполняться тестирование пользователем

4.1.2. Выполнение нефункционального тестирования

В ходе выполнения нефункционального тестирования будет проверяться работоспособность системы под разными нагрузками, быстрота работы программы, а также удобство ее работы.

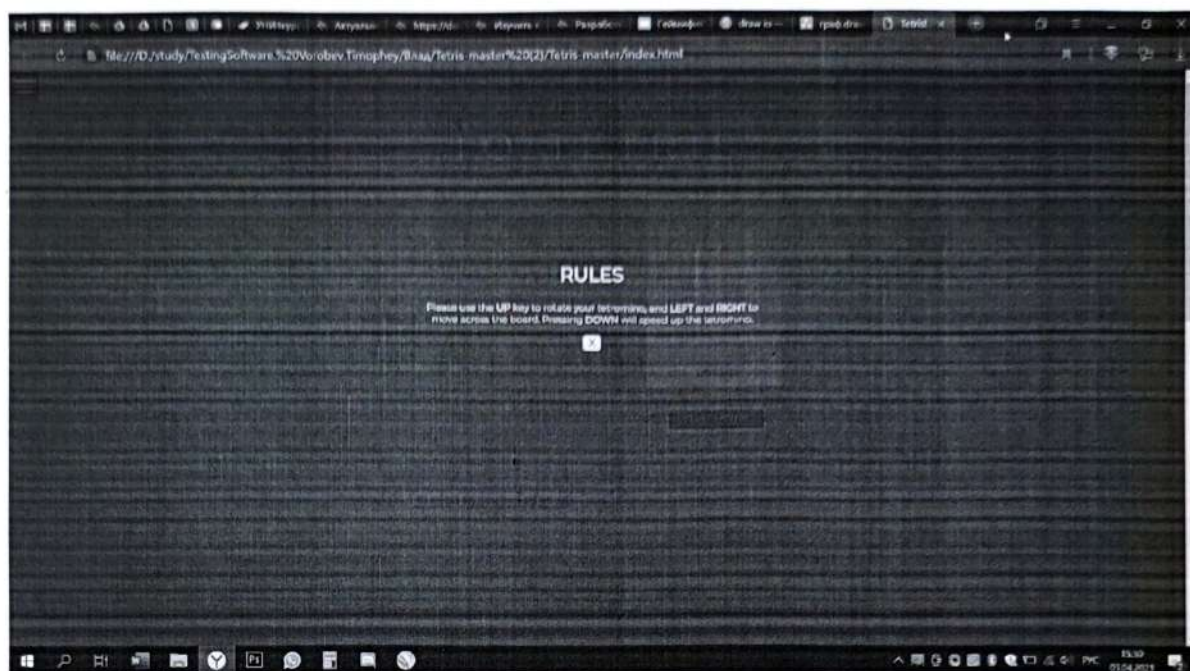
4.2. Результаты системного тестирования

4.2.1. Функциональное тестирование

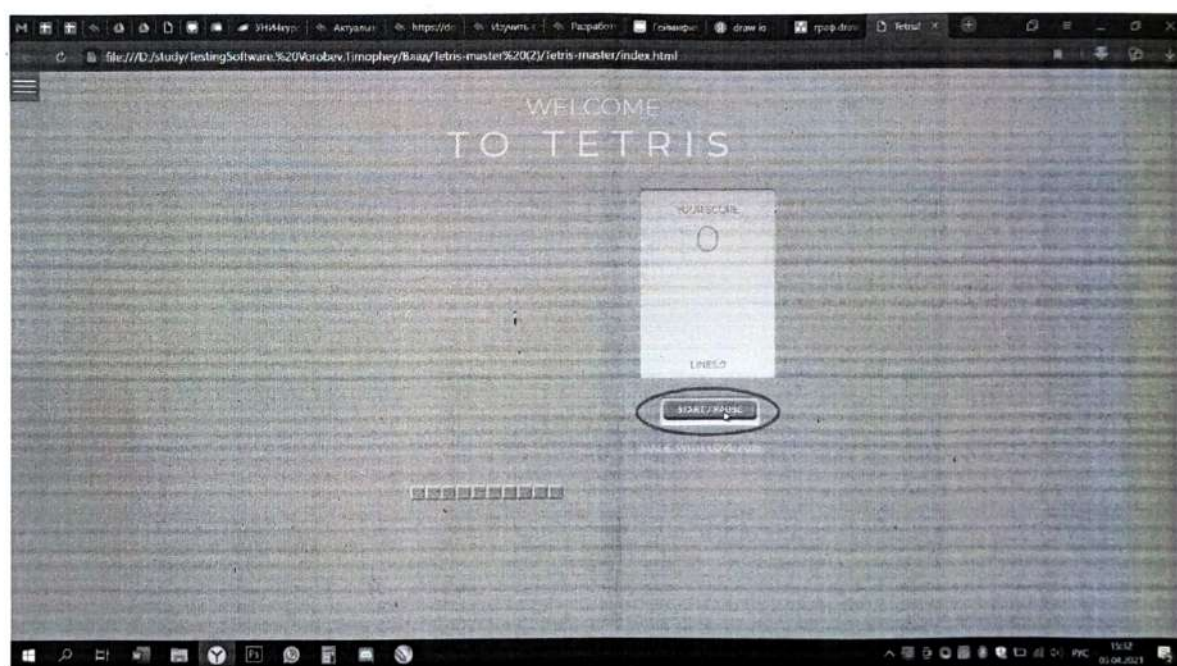
Функциональное тестирование выполняется с использованием функциональной спецификации, предоставленной клиентом, и проверяет систему на соответствие функциональным требованиям. Функциональное тестирование описывает, что делает продукт.

4.2.1.1. Тестирование интерфейса

Игра запускается с помощью открытия файла index.html. После открытия файла загружается первоначальное окно с описанием механизма передвижения и вращения фигур.

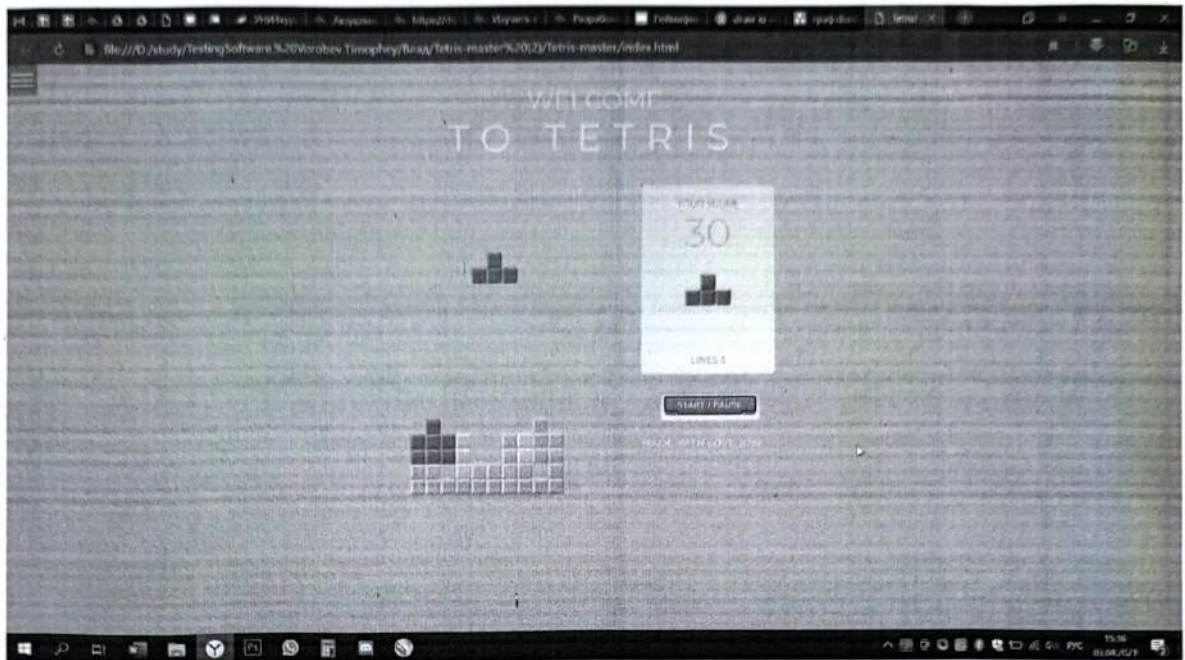


Для начала игры нужно нажать на кнопку «start/pause».



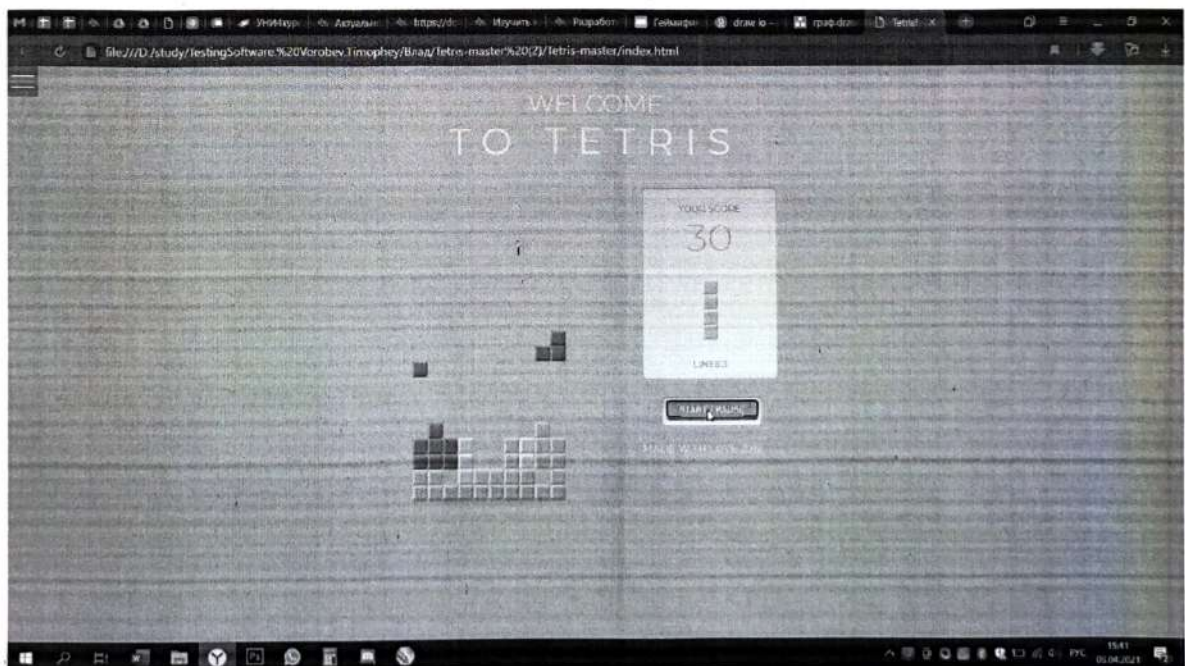
Прорисовка всех фигур выполнена корректна и соответствует цветам, написанным в коде. Добавление очков (количество уничтоженных линий и общее количество баллов) происходит корректно сразу после выполнения поставленных условий.

Единственный минусом в интерфейсе является отсутствие рамок поля.



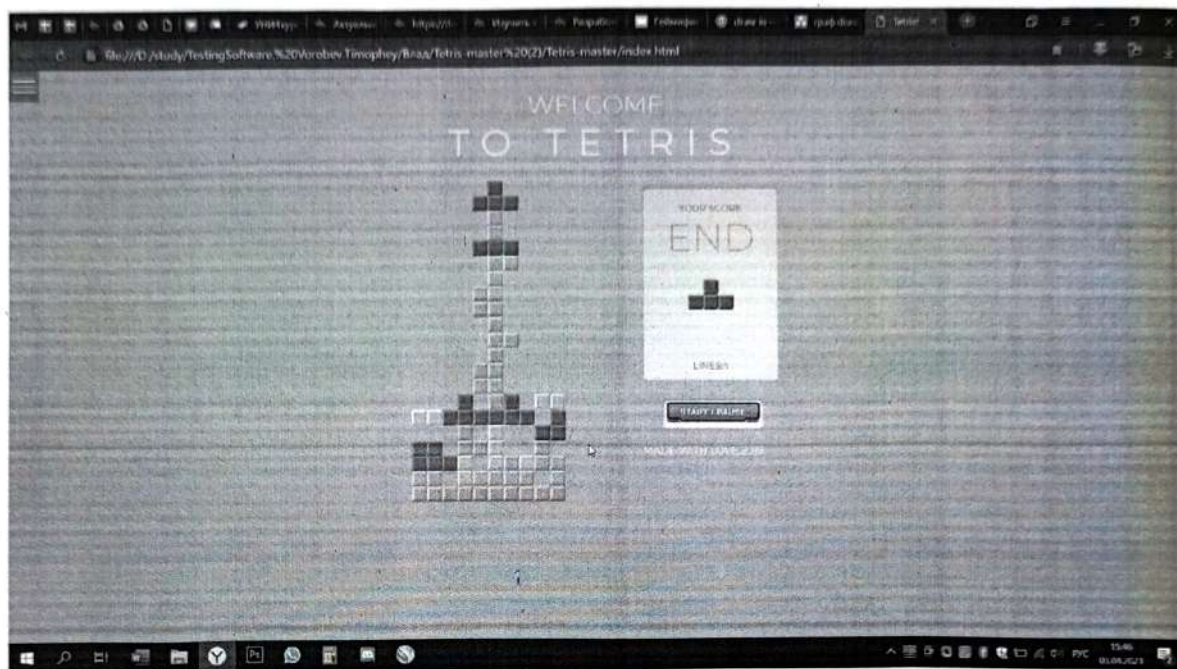
4.2.1.2. Тестирование пользователем

Вращение и передвижение фигуры работает корректно за исключением тех моментов, когда пользователь начинает вращать фигуру на границе игрового поля. Происходит разделение фигуры, отрезанная часть переходит на другую половину поля и продолжает движение дальше по той стороне.



Нажатие на кнопку «start/pause» работает корректно, без задержек.

При заполнении всего поля на табло появляется надпись «END». Дальнейшее появление фигур не происходит. Поэтому можно сделать вывод, что окончание игры работает корректно. Но начать новую игру не получится. Кнопка «start/pause» не работает. Чтобы начать новую игру, нужно обновить страницу.



4.2.2. Нефункциональное тестирование

4.2.2.1. Производительность

Программа при разных нагрузках в целом работает корректно. Движение и вращение фигур происходит быстро. Добавление баллов происходит без нареканий. Кнопка «start/pause» работает корректно.

4.2.2.2. Скорость

Реакция программы на все нажатия, движение и вращение фигур происходит быстро.

4.2.2.3. Удобство

С точки зрения интерфейса в данной программе нет четких границ игрового поля.

С точки зрения пользователя в данной программе есть несколько недочетов. При вращении фигуры, её часть переходит за границы поля, а также после окончания игры, чтобы начать новую игру необходимо обновить страницу.

4.3. Вывод по системному тестированию

По результатам выполнения системного тестирования были составлены наборы тестов, выполнено функциональное и нефункциональное тестирование, описано действие программы в разных ситуациях. На основе тестирования сделан вывод о том, что необходимо добавить границы игрового поля, пересмотреть движение и вращения фигуры, а также добавить корректное начало новой игры после проигрыша.

Вывод

В результате выполнения курсовой работы были составлены наборы модульных, интеграционных и системных тестов, выполнено функциональное и нефункциональное тестирование, а также тестирование по критериям C0, C1, C2, описано действие программы в разных ситуациях, составлен управляющий граф программы, который дал наглядно понять, какие пути развития событий существуют при работе программы.

Модульное тестирование было выполнено в полном объеме.

В интеграционном тестировании сначала был рассмотрен модуль задания стилей файла `style.css`. Тестирование всех функций визуального отображения и стилей было выполнено успешно. Далее был протестирован файл `app.js`, в котором прописаны функции движения фигуры, добавление очков и т.д.. Всё работало верно.

В системном тестировании были проверены следующие категории: тестирование пользователем, производительность, скорость, удобство программы. На основе тестирований сделан вывод о том, что необходимо добавить границы игрового поля, пересмотреть движение и вращения фигуры, а также добавить корректное начало новой игры после проигрыша.