

Разработка игры «Пары» с нуля: верстка, стили и логика*Екимова Яна Сергеевна**Приамурский государственный университет имени Шолом-Алейхема**Студент***Аннотация**

В статье представлен подробный алгоритм разработки логической браузерной игры «Пары: Соединяй по линии» с использованием базовых веб-технологий: HTML, CSS и JavaScript. Разработанное решение реализует полноценный игровой процесс с генерацией игрового поля, механизмом проверки корректности хода на основе правил соединения пар чисел, системой подсчета очков и интерактивным интерфейсом. Особое внимание уделено реализации ключевых алгоритмов: проверки нахождения элементов на одной прямой, поиска свободного пути между ними, валидации пар чисел и интеллектуальной системы подсказок. Материал предназначен для преподавателей и студентов, заинтересованных в применении современных технологий в обучении и стремящихся сделать процесс образовательного взаимодействия более эффективным и увлекательным.

Ключевые слова: Пары: Соединяй по линии, HTML5, CSS, JavaScript.

Developing the Pairs game from scratch: layout, styles, and logic*Ekimova Yana Sergeevna**Sholom Aleichem Priamurskiy State University**Student***Abstract**

The article presents a detailed algorithm for developing the logical browser game "Pairs: Connect along the Line" using basic web technologies: HTML, CSS and JavaScript. The developed solution implements a full-fledged gameplay with the generation of a playing field, a mechanism for checking the correctness of a move based on the rules for connecting pairs of numbers, a scoring system and an interactive interface. Special attention is paid to the implementation of key algorithms: checking the location of elements on the same straight line, searching for a free path between them, validating pairs of numbers and an intelligent hint system. The material is intended for teachers and students interested in using modern technologies in teaching and striving to make the process of educational interaction more effective and exciting.

Keywords: Pairs: Connect by line, HTML5, CSS, JavaScript.

1 Введение

1.1 Актуальность

В современном мире веб-технологий разработка интерактивных приложений и образовательных игр на основе HTML, CSS и JavaScript приобретает всё большую актуальность, поскольку позволяет создавать легковесные, кроссплатформенные и легко доступные решения, функционирующие непосредственно в браузере без необходимости установки дополнительного программного обеспечения. Актуальность реализации логической игры «Пары: Соединяй по линии» заключается в её высоком педагогическом потенциале — она выступает эффективным учебным проектом для освоения фундаментальных принципов веб-разработки, включая динамическое управление DOM-элементами, обработку событий, реализацию алгоритмов поиска путей и проверки условий выигрыша, а также создание адаптивного пользовательского интерфейса. Кроме того, игра способствует развитию логического мышления, внимания и стратегического планирования у игроков, что делает её ценным инструментом в образовательной среде. Адаптация подобных головоломок к современным веб-стандартам отвечает растущему спросу на интерактивный цифровой контент, сочетающий развлекательную и обучающую функции, и демонстрирует, как простые, но хорошо структурированные приложения могут быть использованы для популяризации программирования и цифровой грамотности.

1.2 Обзор исследований

В статье С.В. Рогатина, В.А. Антошкина рассматривается методика разработки компьютерных игр по технологии HTML5 с помощью CASE-средства GameMakerStudio компании YoYoGamesLtd. Проводится тестирование производительности работы компьютерных игр в браузере в зависимости от сложности использованных спрайтов игровых объектов [1]. Разработали игровое приложение «league of football heads» с использованием html5, css3, javascript И.А. Фёдоров, Е.Ю. Кузьменкова [2]. Я. А. Королук, Н.Ю.Куликова показали опыт разработки и использования интерактивных образовательных ресурсов на основе html5 для онлайн-обучения школьников информатике [3]. В работе рассмотрели возможности языка разметки гипертекста HTML5 для создания макета электронного учебника Д.М.Размологов, А.С. Панина, П.В. Захаров. Привели разработанный макет, а также часть таблицы стилей для дизайна макета [4]. И.С. Шегаев, А.Г.Полозов разработали новейший политологический глоссарий: основные категории, понятия, термины [5].

1.3 Цель исследования

Целью исследования является создание полнофункциональной веб-версии логической игры «Пары: Соединяй по линии» с использованием стандартных веб-технологий — HTML, CSS и JavaScript, обеспечивающих кроссплатформенность, адаптивность и интерактивность приложения без необходимости использования сторонних библиотек и фреймворков.

2 Материалы и методы

Для создания приложения используется язык программирования HTML5. В качестве среды разработки используется текстовым редактор Notepad++, оптимизированным для языков программирования.

3 Результаты

Открываем Notepad++ и создаём файл index.html, для этого в меню выбираем файл → новый. Далее нажимаем сочетание клавиш "Ctrl+S" и в поле "Имя файла" вводим название файла "index" с расширением .html, сохраняем данный файл на рабочем столе.

Создадим основную структуру HTML-документа для игры "Пары: Соединяй по линии", которая начинается со стандартной декларации <!DOCTYPE html> и содержит базовую разметку с метатегами кодировки UTF-8 и адаптивного viewport, где в секции <head> указан заголовок страницы, а в теле документа размещен контейнер с заголовком игры, блоком инструкций, игровым полем 6×6, панелью управления с кнопками "Новая игра" и "Подсказка", а также областью для отображения счета и сообщений, что создает семантическую основу для последующего подключения CSS-стилей и JavaScript-логики игровой механики.

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
  <title>Пары: Соединяй по линии</title>
  <style>

  </style>
</head>
<body>
  <div class="container">
    <header>
      <h1>☹ Пары: Соединяй по линии</h1>
    </header>

    <div class="instructions">
      <p>Удали все числа! Выбери два числа, которые соответствуют следующим
правилам:</p>
      <div class="rules">
        <div class="rule"><strong>Одинаковые</strong> или в <strong>сумме дают
10</strong></div>
        <div class="rule">На одной <strong>прямой</strong> (ряд, столбец,
диагональ)</div>
        <div class="rule"><strong>Между ними нет других чисел</strong> (только
пустые ячейки)</div>
      </div>
    </div>
  </div>
```

```
<div class="game-container">
  <div class="score-board">
    <div class="score">Очки: <span id="score">0</span></div>
  </div>

  <div class="grid" id="grid"></div>

  <div class="controls">
    <button id="reset"><span>↺</span> Новая игра</button>
    <button id="hint"><span>💡</span> Подсказка</button>
  </div>

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

<script>

</script>
</body>
</html>
```

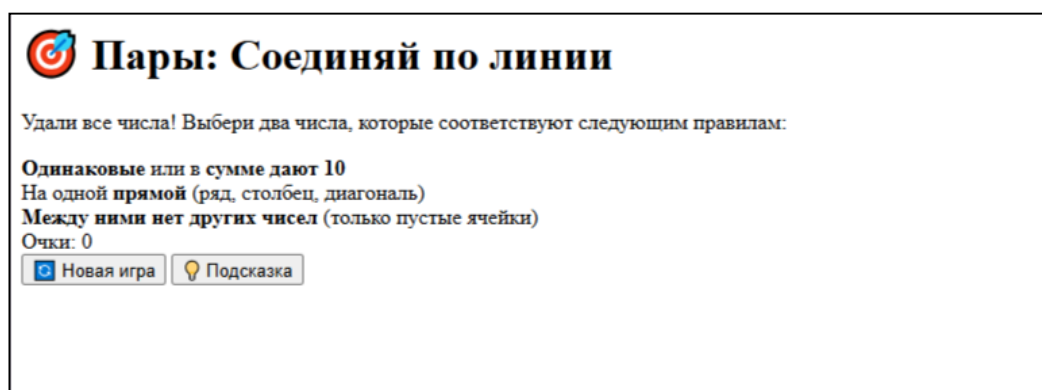


Рисунок 1. Основная структура HTML-документа

Далее добавим блок CSS-стилей, который используется для создания визуально привлекательного и функционального интерфейса игры "Пары: Соединяй по линии". Стили начинаются с определения цветовых переменных CSS для единообразия дизайна, затем следует базовый сброс отступов и применение модели box-sizing для всех элементов. Тело страницы оформлено градиентным фоном с мягкими сине-серыми оттенками и центрированием содержимого.

Основной контейнер игры выполнен в светлом стиле с закругленными углами и тенями для создания карточной эффектности. Игровое поле представляет собой сетку 4×4 ячейки с неоморфным дизайном, где каждая ячейка имеет собственные стили с плавными переходами, эффектами наведения и выделения. Ячейки украшены градиентным наложением через псевдоэлемент для придания глубины.

Панель управления включает стилизованные кнопки с градиентными фонами, иконками и анимациями при наведении, включая эффекты подъема и изменения теней. Для сообщений реализованы анимации пульсации подсказок и специальные стили для сообщений о победе. Все элементы интерфейса адаптивно меняют размеры и расположение на мобильных устройствах благодаря медиа-запросам, обеспечивая удобство использования на различных экранах.

Данный код пропишем на следующей строчке после тега <title Пары: Соединяй по линии</title> в секции <head> HTML-документа.

```
:root {
  --primary: #4361ee;
  --secondary: #3a0ca3;
  --accent: #f72585;
  --success: #4cc9f0;
  --warning: #f9c74f;
  --light: #f8f9fa;
  --dark: #2b2d42;
  --card-bg: #ffffff;
  --cell-bg: #e9ecef;
  --cell-hover: #d8e2dc;
  --shadow: rgba(0, 0, 0, 0.1);
}

* {
  box-sizing: border-box;
  margin: 0;
  padding: 0;
}

body {
  font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
  text-align: center;
  background: linear-gradient(135deg, #f5f7fa 0%, #c3cfe2 100%);
  margin: 0;
  padding: 20px;
  min-height: 100vh;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  color: var(--dark);
}

.container {
  max-width: 800px;
  width: 100%;
  padding: 20px;
}

header {
```

```
    margin-bottom: 30px;
  }

  h1 {
    color: var(--primary);
    font-size: 2.5rem;
    margin-bottom: 15px;
    text-shadow: 2px 2px 4px rgba(0,0,0,0.1);
  }

  .instructions {
    background: var(--card-bg);
    padding: 20px;
    border-radius: 16px;
    box-shadow: 0 8px 20px var(--shadow);
    margin-bottom: 25px;
    text-align: left;
  }

  .instructions p {
    color: var(--dark);
    margin-bottom: 15px;
    line-height: 1.6;
  }

  .rules {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));
    gap: 15px;
    margin-top: 15px;
  }

  .rule {
    background: #f8f9fa;
    padding: 12px;
    border-radius: 10px;
    border-left: 4px solid var(--primary);
  }

  .game-container {
    background: var(--card-bg);
    padding: 25px;
    border-radius: 20px;
    box-shadow: 0 10px 25px var(--shadow);
    margin: 0 auto;
    max-width: 500px;
  }

  .score-board {
    display: flex;
    justify-content: center;
    align-items: center;
```

```
margin-bottom: 20px;
gap: 15px;
}

.score {
  font-size: 1.5rem;
  font-weight: bold;
  color: var(--primary);
  background: #f0f4ff;
  padding: 8px 20px;
  border-radius: 50px;
  box-shadow: inset 0 0 5px rgba(0,0,0,0.1);
}

.grid {
  display: grid;
  grid-template-columns: repeat(4, 70px);
  grid-template-rows: repeat(4, 70px);
  gap: 6px;
  margin: 20px auto;
  width: fit-content;
  background: linear-gradient(145deg, #e6e6e6, #ffffff);
  padding: 15px;
  border-radius: 15px;
  box-shadow: 5px 5px 15px #d9d9d9, -5px -5px 15px #ffffff;
}

.cell {
  width: 70px;
  height: 70px;
  background: var(--cell-bg);
  border-radius: 10px;
  display: flex;
  align-items: center;
  justify-content: center;
  font-size: 28px;
  font-weight: bold;
  cursor: pointer;
  user-select: none;
  transition: all 0.3s ease;
  box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
  color: var(--dark);
  position: relative;
  overflow: hidden;
}

.cell:before {
  content: "";
  position: absolute;
  top: 0;
  left: 0;
  right: 0;
```

```
bottom: 0;
background: linear-gradient(135deg, rgba(255,255,255,0.2) 0%, rgba(0,0,0,0.05)
100%);
border-radius: 10px;
}

.cell:hover {
background: var(--cell-hover);
transform: translateY(-3px);
box-shadow: 0 7px 10px rgba(0, 0, 0, 0.15);
}

.cell.selected {
background: var(--warning);
color: #000;
transform: scale(1.05);
box-shadow: 0 0 15px rgba(249, 199, 79, 0.6);
}

.cell.removed {
visibility: hidden;
background: transparent;
box-shadow: none;
transform: scale(0);
}

.controls {
display: flex;
justify-content: center;
gap: 15px;
margin-top: 25px;
flex-wrap: wrap;
}

button {
padding: 12px 25px;
font-size: 16px;
font-weight: 600;
border: none;
border-radius: 50px;
cursor: pointer;
transition: all 0.3s ease;
box-shadow: 0 4px 8px rgba(0, 0, 0, 0.15);
display: flex;
align-items: center;
gap: 8px;
}

button:hover {
transform: translateY(-3px);
box-shadow: 0 6px 12px rgba(0, 0, 0, 0.2);
}
```

```
button:active {
  transform: translateY(0);
}

#reset {
  background: var(--primary);
  color: white;
}

#reset:hover {
  background: var(--secondary);
}

#hint {
  background: var(--accent);
  color: white;
}

#hint:hover {
  background: #c1121f;
}

.message {
  margin-top: 20px;
  min-height: 30px;
  font-weight: 600;
  padding: 10px;
  border-radius: 8px;
  transition: all 0.3s ease;
}

.win {
  color: var(--success);
  font-size: 1.2rem;
  background: rgba(76, 201, 240, 0.1);
  padding: 15px;
}

.hint {
  border: 2px solid var(--warning);
  animation: pulse 1s infinite;
}

@keyframes pulse {
  0% { box-shadow: 0 0 0 0 rgba(249, 199, 79, 0.4); }
  70% { box-shadow: 0 0 0 10px rgba(249, 199, 79, 0); }
  100% { box-shadow: 0 0 0 0 rgba(249, 199, 79, 0); }
}

@media (max-width: 600px) {
  .grid {
```

```
grid-template-columns: repeat(4, 60px);
grid-template-rows: repeat(4, 60px);
}

.cell {
  width: 60px;
  height: 60px;
  font-size: 24px;
}

h1 {
  font-size: 2rem;
}

.rules {
  grid-template-columns: 1fr;
}
}
```

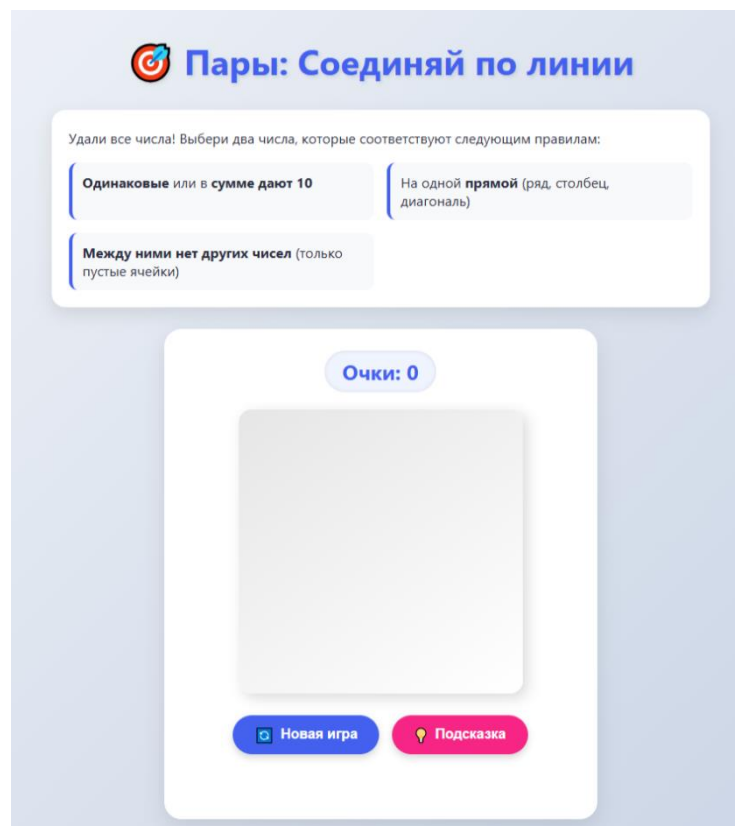


Рисунок 2. Добавление CSS-стилей

Затем переходим к реализации JavaScript-логики игры, которая начинается с получения ссылок на DOM-элементы: игровое поле (grid), элемент отображения счета (scoreElement), область сообщений (messageElement) и кнопки управления (resetButton, hintButton).

Инициализируем основные переменные состояния игры: массив cells для хранения ячеек, selectedCell для отслеживания выбранной ячейки и score для хранения текущего счета игрока.

Определяем массив `validPairs`, содержащий все допустимые комбинации пар чисел, которые можно соединять. Включаем в него как пары одинаковых чисел, так и пары, дающие в сумме 10, что составляет основное правило игры.

Вставим данный код после закрывающего тега `</div>` и после открывающего тега `<script>`:

```
const grid = document.getElementById('grid');
const scoreElement = document.getElementById('score');
const messageElement = document.getElementById('message');
const resetButton = document.getElementById('reset');
const hintButton = document.getElementById('hint');

let cells = [];
let selectedCell = null;
let score = 0;

// Все допустимые пары
const validPairs = [
  [1,1], [2,2], [3,3], [4,4], [5,5], [6,6], [7,7], [8,8], [9,9],
  [1,9], [2,8], [3,7], [4,6], [6,4], [7,3], [8,2], [9,1]
];
```

Далее напишем функцию `createGrid()`, которая очищает игровое поле и инициализирует новую игру, создавая пустую сетку 4×4 и заполняя её 16 числами — 8 случайно выбранными парами чисел, сумма которых равна 10 или одинаковыми значениями, при этом порядок чисел перемешивается алгоритмом Фишера-Йейтса; затем каждая ячейка отображается на экране как элемент `div` с соответствующим значением, к которому привязывается обработчик клика для выбора и сопоставления пар, при этом реализованы проверки: на прямую линию между ячейками (`areAligned`), наличие свободного пути между ними без препятствий (`hasClearPath`) и корректность пары (`isValidPair`), а при успешном совпадении ячейки удаляются, счёт увеличивается, и проверяется условие победы — полное очищение поля, а также будет добавлена функция подсказки (`findHint`), которая ищет первую доступную пару для удаления и временно выделяет её, помогая игроку.

Вставим данный код в продолжение тега `<script>`:

```
function createGrid() {
  grid.innerHTML = '';
  cells = Array(4).fill().map(() => Array(4).fill(null));
  const numbers = [];

  // Генерируем 8 пар → 16 чисел
  for (let i = 0; i < 8; i++) {
    const pair = validPairs[Math.floor(Math.random() * validPairs.length)];
    numbers.push(pair[0], pair[1]);
  }
}
```

```
// Перемешиваем
for (let i = numbers.length - 1; i > 0; i--) {
  const j = Math.floor(Math.random() * (i + 1));
  [numbers[i], numbers[j]] = [numbers[j], numbers[i]];
}

// Заполняем
let idx = 0;
for (let row = 0; row < 4; row++) {
  for (let col = 0; col < 4; col++) {
    cells[row][col] = numbers[idx++];
    const cell = document.createElement('div');
    cell.className = 'cell';
    cell.dataset.row = row;
    cell.dataset.col = col;
    cell.textContent = cells[row][col];
    cell.addEventListener('click', handleCellClick);
    grid.appendChild(cell);
  }
}

selectedCell = null;
score = 0;
scoreElement.textContent = score;
messageElement.textContent = "";
messageElement.classList.remove('win');
}

// Проверка: есть ли чистый путь между двумя ячейками?
function hasClearPath(r1, c1, r2, c2) {
  if (r1 === r2 && c1 === c2) return false;

  const dr = r2 > r1 ? 1 : r2 < r1 ? -1 : 0;
  const dc = c2 > c1 ? 1 : c2 < c1 ? -1 : 0;

  let r = r1 + dr;
  let c = c1 + dc;

  while (r !== r2 || c !== c2) {
    if (cells[r][c] !== null) {
      return false; // Есть препятствие
    }
    r += dr;
    c += dc;
  }
  return true;
}

// Проверка, подходят ли числа
function isValidPair(n1, n2) {
  return n1 === n2 || n1 + n2 === 10;
}
```

```

// Проверка, на одной ли прямой
function areAligned(r1, c1, r2, c2) {
  return r1 === r2 || c1 === c2 || Math.abs(r1 - r2) === Math.abs(c1 - c2);
}

function handleCellClick(e) {
  const cell = e.target;
  const row = parseInt(cell.dataset.row);
  const col = parseInt(cell.dataset.col);

  if (cells[row][col] === null) return; // Уже удалено

  if (selectedCell) {
    const r1 = selectedCell.row, c1 = selectedCell.col;
    const r2 = row, c2 = col;

    const prevCell = document.querySelector(`.cell[data-row="${r1}"][data-col="${c1}"]`);

    if (areAligned(r1, c1, r2, c2) && hasClearPath(r1, c1, r2, c2)) {
      const n1 = cells[r1][c1];
      const n2 = cells[r2][c2];

      if (isValidPair(n1, n2)) {
        cell.classList.add('removed');
        prevCell.classList.add('removed');
        cells[r1][c1] = null;
        cells[r2][c2] = null;

        score += 10;
        scoreElement.textContent = score;
        messageElement.textContent = `+10! Удалены ${n1} и ${n2}`;
        setTimeout(() => { messageElement.textContent = ""; }, 1500);

        // Проверка победы
        if (isWin()) {
          messageElement.textContent = 🎉 Поздравляем! Поле очищено!;
          messageElement.className = 'message win';
        }
        else {
          messageElement.textContent = `❌ ${n1} и ${n2} не подходят.`;
        }
        else {
          messageElement.textContent = '⊘ Нет прямой или путь заблокирован.';
        }

        prevCell.classList.remove('selected');
        selectedCell = null;
      } else {
        cell.classList.add('selected');
        selectedCell = { row, col };
      }
    }
  }
}

```

```

    }
  }

  function isWin() {
    for (let r = 0; r < 4; r++) {
      for (let c = 0; c < 4; c++) {
        if (cells[r][c] !== null) return false;
      }
    }
    return true;
  }

  function findHint() {
    const positions = [];
    for (let r = 0; r < 4; r++) {
      for (let c = 0; c < 4; c++) {
        if (cells[r][c] !== null) {
          positions.push({ r, c });
        }
      }
    }
  }

  for (let i = 0; i < positions.length; i++) {
    for (let j = i + 1; j < positions.length; j++) {
      const a = positions[i], b = positions[j];
      if (areAligned(a.r, a.c, b.r, b.c) && hasClearPath(a.r, a.c, b.r, b.c)) {
        const n1 = cells[a.r][a.c];
        const n2 = cells[b.r][b.c];
        if (isValidPair(n1, n2)) {
          const cell1 = document.querySelector(`.cell[data-row="${a.r}"][data-col="${a.c}"]`);
          const cell2 = document.querySelector(`.cell[data-row="${b.r}"][data-col="${b.c}"]`);

          cell1.classList.add('hint');
          cell2.classList.add('hint');

          messageElement.textContent = `💡 Подсказка: ${n1} ↔ ${n2}`;

          setTimeout(() => {
            cell1.classList.remove('hint');
            cell2.classList.remove('hint');
            messageElement.textContent = "";
          }, 2500);
          return;
        }
      }
    }
  }

  messageElement.textContent = `💡 Нет ходов? Нажми "Новая игра";
  setTimeout(() => { messageElement.textContent = ""; }, 2000);
}

```

Осталось реализовать обработчики событий для интерактивных элементов управления и инициализацию начального состояния сетки. В частности, к кнопкам сброса и подсказки добавляются слушатели событий, обеспечивающие повторное создание игрового поля и поиск возможных решений для текущего состояния. При загрузке приложения вызывается функция `createGrid()`, которая генерирует новую случайную сетку sudoku, соответствующую правилам игры. Это позволяет сразу начать работу с визуально корректным и валидным игровым полем.

Вставим данный код в продолжение тега `<script>`:

```
// Кнопки
resetButton.addEventListener('click', createGrid);
hintButton.addEventListener('click', findHint);

// Старт
createGrid();
```

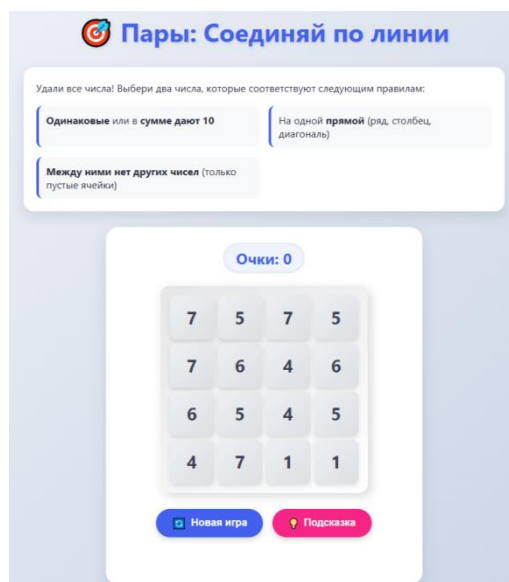


Рисунок 3. Главная страница

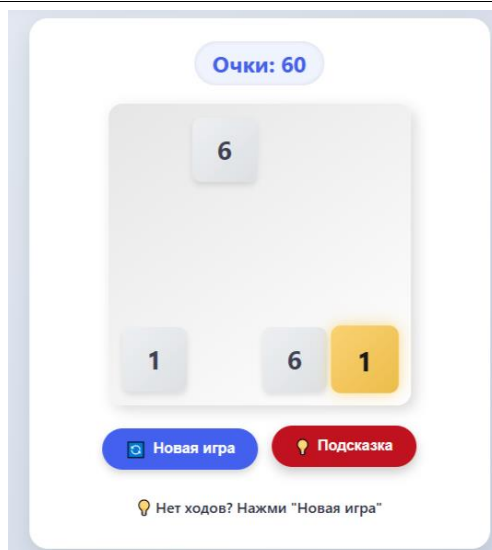


Рисунок 4. Процесс игры

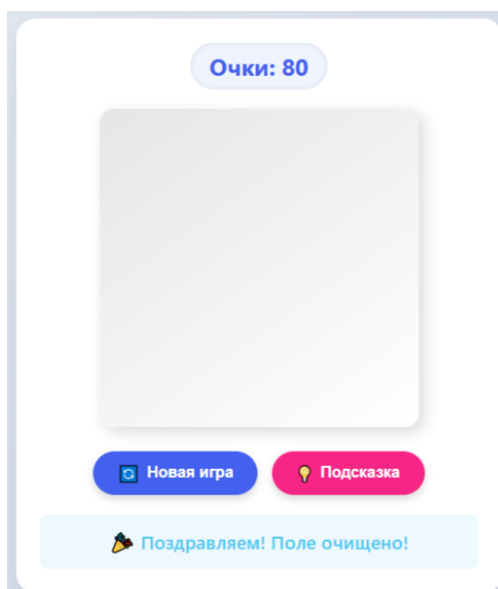


Рисунок 5. Конец игры

4 Выводы

В данной работе подробно рассмотрено создание логической веб-игры «Пары: Соединяй по линии» с использованием технологий HTML, CSS и JavaScript, реализованной без привлечения сторонних библиотек. Разработанная игра включает полную игровую механику, основанную на правилах соединения пар чисел, одинаковых или дающих в сумме 10, расположенных на одной прямой с чистым путём между ними. Реализованы ключевые компоненты: динамическая генерация игрового поля 4×4 с помощью алгоритма Фишера–Йейтса для перемешивания чисел, проверка выравнивания и наличия свободного пути между ячейками, обработка пользовательских действий, система подсчёта очков и визуальное выделение выбранных элементов. Добавлена функция подсказки, автоматически находящая доступную пару для хода, что повышает игровую доступность. Интерфейс игры адаптирован под различные устройства за счёт гибкой

вёрстки и медиа-запросов, а визуальное оформление выполнено в современном неоморфном стиле с плавными анимациями, тенями и эффектами наведения, что обеспечивает высокий уровень пользовательского опыта. Созданная игра демонстрирует эффективность использования базовых веб-технологий для разработки интерактивных образовательных приложений, сочетающих развлекательную и развивающую функции. Предложенное решение ориентировано на преподавателей и студентов, заинтересованных в применении современных веб-технологий для повышения эффективности образовательного процесса и превращения обучения в более увлекательный и интерактивный опыт.

Библиографический список

1. Рогатин С. В., Антошкин В. А. Разработка компьютерных игр по технологии html5 //Информатика и прикладная математика. 2015. №. 21. С. 12-16.
2. Фёдоров И. А., Кузьменкова Е. Ю. Разработка игрового приложения «league of football heads» с использованием html5, css3, javascript //Новые математические методы и компьютерные технологии в проектировании, производстве и научных исследованиях. 2018. С. 311.
3. Королёк Я. А., Куликова Н. Ю. Опыт разработки и использования интерактивных образовательных ресурсов на основе html5 для онлайнобучения школьников информатике //XXV Всероссийская студенческая научно-практическая конференция Нижневартковского государственного университета. 2023. С. 48-55.
4. Размологов Д. М., Панина А. С., Захаров П. В. Использование html5 для разработки электронного учебника //Современные тенденции развития науки и технологий. 2016. №. 4-4. С. 92-94.
5. Шегаев И. С., Полозов А. Г. Новейший политологический глоссарий: основные категории, понятия, термины. М.: Перо, 2015. 242 с.