ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ГОРОДА МОСКВЫ ДОПОЛНИТЕЛЬНОГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ ЦЕНТР ПРОФЕССИОНАЛЬНЫХ КВАЛИФИКАЦИЙ И СОДЕЙСТВИЯ ТРУДОУСТРОЙСТВУ «ПРОФЕССИОНАЛ»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к итоговой аттестационной работе на тему

«Разработка web-ресурса с использованием технологий HTML, CSS, JavaScript, Bootstrap, jQuery»

(на примере web-ресурса: https://lusty.p-host.in/Diplom/d.html

репозиторий проекта: https://github.com/sergey7181/Diplom)

слушателя **Гурьева Сергея Михайловича** группы №: **066**

программы профессиональной переподготовки

«Frontend разработка»

Москва, 2024

ОГЛАВЛЕНИЕ.

ПОСТАНОВКА ЗАДАЧИ И ПЛАН РАБОТЫ.

Задачей является создание веб-приложения: карточная игра «Пьяница». В начале работы над проектом, необходимо определиться, какой будет внешний вид сайта. Функционал игры должен быть понятен пользователю, иметь дружественный интерфейс и управляться с помощью нажатия левой клавиши «мышки» на кнопки на экране монитора.

План работ:

1. Создание базы данных на хостинге.
2. Создание репозитория на GitHub.
3. Подключение фреймворка Bootstrap.
4. Подключение библиотеки jQuery.
5. Получение изображений игральных карт.
6. Определение внешнего вида сайта на языке разметки HTML и каскадных таблиц стилей CSS (далее «стилей»).
7. Написание функции раздачи карт.
8. Написание основного функционала игры.
9. Написание стартовой функции.
10. Проверка работоспособности веб-приложения.

ОСНОВНАЯ ЧАСТЬ.

Назначение веб-ресурса.

Созданное веб-приложение это карточная игра «Пьяница». Правила игры:

1. Каждый игрок получает 18 игральных карт случайным образом (Рис 2. (см. стр. 11)).
2. Игроки бросают на игральный стол по одной верхней карте из своей колоды.
3. У кого из игроков старше карта (без учета масти), тот и забирает карты со стола себе (Рис. 3. (см. стр. 12)).
4. Если карты одинаковые по-старшенству, то это спорная ситуация «Спор» и пункт 2. повторяется (Рис. 4. (см. стр. 12)).
5. Есть две исключительные ситуации: «6» берет «туза» (и только его) (Рис. 5. (см. стр. 13)), а «10» берет «короля» (или меньшую по-старшенству карту) (Рис. 6. (см. стр. 13)).
6. Игра заканчивается, когда у одного из игроков заканчиваются карты (Рис. 7. и Рис. 8. (см. стр. 14)).

Интерфейс веб-ресурса.

Весь диалог с пользователем выводится экран:

- как начать игру,
- что сейчас надо делать,
- чья взятка,
- кто выиграл или проиграл.

Также выводится информация о наличии в колоде игрока:

- «тузов»,
- «королей»,
- «шестерок»,

- количества карт.

Описание этапов разработки, описание функционала с приложением листингов исходного программного кода.

Этап разработки начинался с создания страницы, на которой подключаются:

- фреймворк «Bootstrap» (далее “фреймворк”),
- шрифты,
- иконка,
- файл со стилями веб-страницы «style.css»,
- библиотека jQuery,
- функционал игры на языке JS в файле «script.js».

Листинг 1.

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="Bootstrap/bootstrap.css">
  <link rel="preconnect" href="https://fonts.googleapis.com" />
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />
  <link href="https://fonts.googleapis.com/css2?family=Comfortaa:wght@400,700&display=swap"
  <link rel="stylesheet" href="style.css">
  <link rel="icon" href="Card/icon.jpg">
  <title>Игра в карты "Пьяница"</title>
</head>
```

Листинг 2.

```
  <!-- <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.4/jquery.min.js"></scr
  <script src="jQuery/jquery.js"></script>
  <script src="script.js"></script>
</body>
</html>
```

Проект состоит из трех файлов: «d.html», «style.css», «script.js» и трех папок: одна – с изображениями игральных карт и иконки «Card», вторая – со стилями фреймворка «Bootstrap», третья – с библиотекой jQuery.

Адаптивность веб-страницы реализована с помощью стилей фреймворка и функции clamp() в стилях файла «style.css». Функция clamp() обеспечила адаптивность по вертикали кнопок и информационных экранов.

Для контроля корректности алгоритма наличие карт в колодах игроков выводится в консоль в виде массива (в этом виде они хранятся и обрабатываются в коде). Первой выводится колода карт компьютера, второй – пользователя. Цифра в элементе массива определяет «старшинство» карты, буква – масть карты: t – «треф (крести)», c – «черви», b – «бубны», p – «пики (вини)».

При загрузке веб-страницы определяются глобальные переменные.

Листинг 3.

```
let avt =  [];                                  // Колода карт автора
let user = [];                                  // Колода карт игрока
let col2 = document.getElementById("col_2");    // Место на столе, куда бросает карту автор
let col3 = document.getElementById("col_3");    // Место на столе, куда бросает карту игрок
let info = document.getElementById("info");     // Верхний экран
let info_t = document.getElementById("info_t"); // Кол-во тузов у игрока
let info_k = document.getElementById("info_k"); // Кол-во королей у игрока
let info_6 = document.getElementById("info_6"); // Кол-во "6" у игрока
let info_V = document.getElementById("info_V"); // Всего карт у игрока
let vziatka = [];                               // Взятка на столе
```

Пользователь видит Рис.1 (см. стр. 11) и ему предлагается начать игру, нажав на кнопку «Старт». Запускается функция start() (Листинг 4):

- обнуляются переменные,
- активируется кнопка "Бросайте карту",
- определяется начальное положение бокового экрана (функция mesto_ekran2()),
- включается прослушивание события изменения размера дисплея гаджета пользователя для изменения местоположения бокового экрана,
- раздаются карты игрокам – функция razdacha_card() (Листинг 5.),

- включается боковой экран с информацией о наличии карт у пользователя – функция infor() (Листинг 6.).

Листинг 4.

```
function start() {
  avt =  []; user = []; vziatka = [];
  document.getElementById("col_1").hidden = false;     // Колода карт автора на столе
  document.getElementById("col_1").style.opacity = 1;
  col2.hidden = true;
  col3.hidden = true;
  let col4 = document.getElementById("col_4");         // Колода карт игрока на столе
  col4.hidden = false;
  col4.style.opacity = 1;

  document.querySelector(".brosoc").setAttribute('onclick', "brosoc()");  // Активация кнопки

  document.querySelector(".ekran2").hidden = false;    // Включаем боковой экран
  mesto_ekran2();                                      // Начальное местоположение бокового э
  window.addEventListener('resize', mesto_ekran2);     // Изменяем местоположение бокового эк

  razdacha_card();                                     // Раздаем карты автору и игроку
  infor();                                             // Вывод инф-ии на боковой экран
  inf_console();                                       // Вывод колоды карт автора и игрока в
  info.innerText = `Карты розданы\nБросайте карту`;

  function mesto_ekran2() {                            // Назначаем местоположение бокового э
    document.querySelector(".ekran2").style.top = col4.getBoundingClientRect().top - 13 + "px
  }
}
```

Листинг 5.

```
function razdacha_card() {
  let masts = ['t', 'c', 'b', 'p'];    // Масть карт t-треф, c-черви, b-бубны, p-пики
  masts.forEach( (item) => {           // Формирование колоды автора из 36 карт
    for (let i = 0; i < 9; i++) {
      avt.push(`${i}` + item);
    }
  });
  for (let i = 0; i < 18; i++) {            // Формирование колоды карт игрока: берем из коло
    let sl = Math.round(Random(0, 35 - i));
    user[i] = avt[sl];
    avt.splice(sl, 1);
  }

  function Random(min, max) {                   // Генерация случайных чисел в диапазоне (mi
    return Math.random() * (max - min) + min;
  }
} // Конец функции razdacha_card()
```

Листинг 6.

```
function infor() {                  // Вывод инф-ии на боковой экран
  info_t.innerText = schit("8");
  info_k.innerText = schit("7");
  info_6.innerText = schit("0");
  info_V.innerText = user.length;;

  function schit(a) {               // Вычисляет кол-во карт старшинства 'a'
    let st = user.filter( (item) => { return item.includes(a); });
    return st.length;
  }
} // Конец функции infor()
```

В результате пользователь видит Рис.2 (см. стр. 11) и ему предлагается бросить карту, нажав на кнопку «Бросить карту». Запускается функция brosoc() (Листинг 7.), в которой реализован основной алгоритм игры:

- медленно исчезает взятка и чтобы функция fadeOut() успела завершиться, оставшаяся часть кода функции brosoc() помещается в функцию setTimeuot() для асинхронности (задерживаеся выполнение остального кода),
- запускается функция infor() (Листинг 6.)., обновляющая информацию на боковом дисплее,
- проверяется условие завершения игры (Листинг 7.),
- появляются новые карты на столе » (Листинг 9.),
- проверяется условие одновременного появления на столе пары карт: «6» и «туз», «10» и «король» (Листинг 9.),
- определяется, кому из игроков принадлежит взятка (карты на столе) (Листинг 8.),
- определяется функция dob_col(), добавляющая взятку в колоду карт игрока (Листинг 8.).

Листинг 7.

```
function brosoc() {
  infor();                          // Вывод инф-ии на боковой экран
  if (!spor) {                      // Если не "Спор"
    $(col2).fadeOut(900);           // "Растворение" взятки
    $(col3).fadeOut(900);
  }
  setTimeout(()=> {                 // Асинхронное выполнение, чтобы успела выполниться fadeOut
    while(true) {
      spor = false;
      let avt0 = avt.shift();       // Берем верние карты из колоды
      let user0 = user.shift();
      if(user0 == undefined) {                              // Если карты закончились - коне
        info.innerText = `Вы проиграли!\nСыграем еще ?`;
        document.querySelector(".brosoc").disabled = true;  // Дезактивация кнопки "Бросаем
        document.getElementById("col_4").hidden = true;
        col2.hidden = true;
        col3.hidden = true;
        break;
      }
      if(avt0 == undefined) {
        info.innerText = `Вы выиграли!!!\nСыграем еще ?`;
        document.querySelector(".brosoc").disabled = true;
        document.getElementById("col_1").hidden = true;
        col2.hidden = true;
        col3.hidden = true;
        break;
      }
```

Листинг 8.

```
      if(avt0_int > user0_int) {
        avt = dob_col(avt, "Моя взяла");      // Добавляем взятку в колоду и выводим: "Моя взя
      } else if( avt0_int < user0_int ) {
        user = dob_col(user, "Ваша взяла");
      } else {
        spor = true;
        info.innerText = `Это спор\nБросайте карту`;
        break;
      }
      inf_console();
      break;

      function dob_col(col_x, soob) {
        info.innerText = soob + `\n` + "Бросайте карту" ;  // Выводим: "Моя взяла" или "Ваша
        col_x = col_x.concat(vziatka);                     // Добавляем взятку в колоду
        vziatka = [];                                      // Обнуляем взятку
        return col_x;
      }
    }       // Конец цикла while(true)
  }, 1000); // Конец функции setTimeout()
}           // Конец функции brosoc()
```

Листинг 9.

```
vziatka.push(`${avt0}`);                // Записываем карты в массив взятки на столе
vziatka.push(`${user0}`);

col3_img.setAttribute('src', `Card/${user0}.gif`);                              // Бросаем ка
col2.innerHTML = `<img src="Card/${avt0}.gif" class="img-fluid" alt="Карта">`;
col3.innerHTML = `<img src="Card/${user0}.gif" class="img-fluid" alt="Карта">`;

col2.style.removeProperty('display');  // Включаем видимость взятки на столе
col3.style.display = 'block';          // Отображается на экране белый прямоугольник перед за
col2.hidden = false;
col3.hidden = false;
$(col2).fadeIn(900);                   // Постепенное появление взятки
$(col3).fadeIn(900);

let avt0_int = parseInt(avt0);         // Считываем цифру карты отвечающую за старшинство
let user0_int = parseInt(user0);
if( (avt0_int == 0 && user0_int == 8) || (avt0_int == 4 && user0_int == 7) ) { // Если встреч
  avt = dob_col(avt, "Моя взяла");                    // Добавляем взятку в колоду и выводим
  inf_console();
  break;
}
if( (user0_int == 0 && avt0_int == 8) || (user0_int == 4 && avt0_int == 7) ) {
  user = dob_col(user, "Ваша взяла");
  inf_console();
  break;
}
```

В программе использовалось шесть методов обработки массивов.

Работоспособность программы можно проверить по адресу размещения на хостинге:

https://lusty.p-host.in/Diplom/d.html

Исходные файлы проекта можно просмотреть на GitHub по ссылке: https://github.com/sergey7181/Diplom

СПИСОК ЛИТЕРАТУРЫ.

1. Подробное руководство по HTML и CSS [электронный ресурс]:
   офиц. сайт, URL https://itchief.ru/html-and-css/.
2. Современный учебник JavaScript [электронный ресурс]:
   офиц. сайт, URL https://learn.javascript.ru/.
3. "Bootstrap 4. Документация на русском языке" [электронный ресурс]:
   офиц. сайт, URL https://bootstrap-4.ru/docs/4.6/getting-started/introduction/.
4. Русская документация по API jQuery [электронный ресурс]:
   офиц. сайт, URL https://jquery-docs.ru/.

СКРИНШОТЫ веб-ресурса.

Рис 1. После загрузки веб-ресурса.

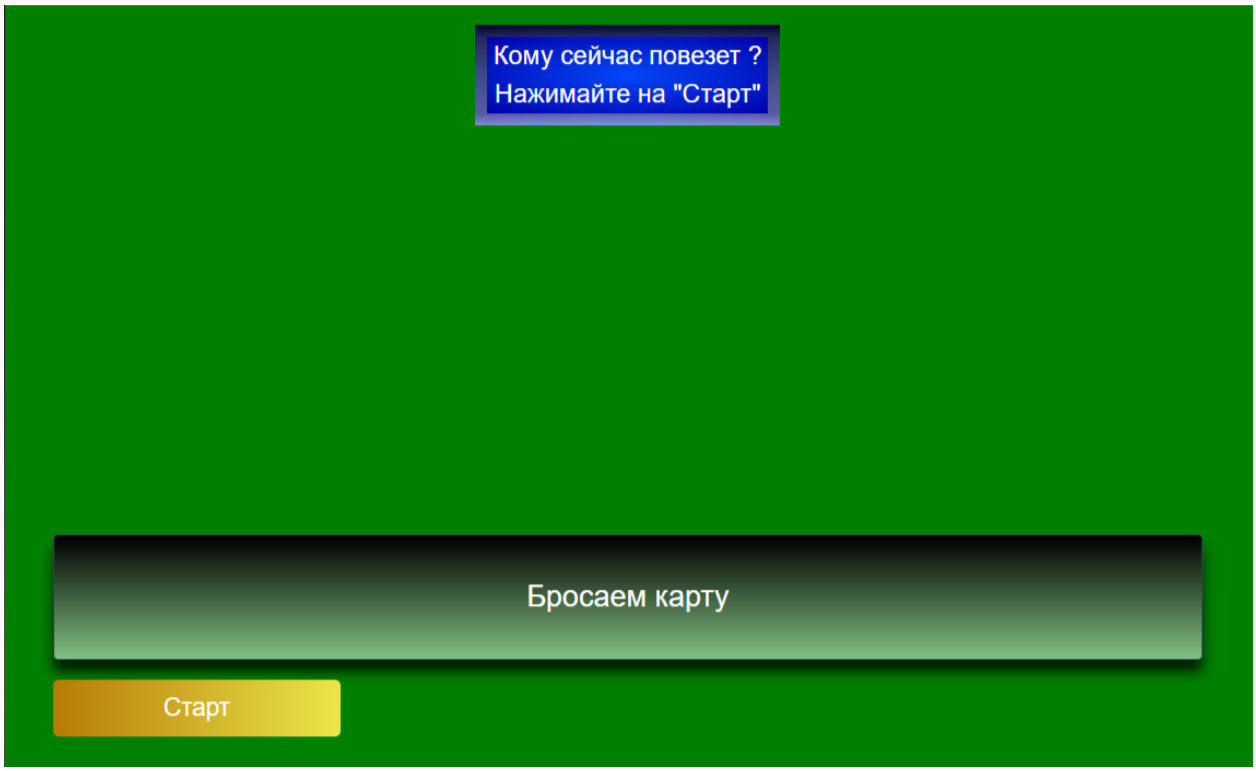

Рис 2. После нажатия на кнопку «Старт»

Рис 3. Взятка пользователя.

Рис 4. Спорная ситуация «Спор».

Рис 5. «6» берет «туза» (и только его).

Рис 6. «10» берет «короля» (или меньшую по-старшенству карту).

Рис 7. Пользователь выиграл.

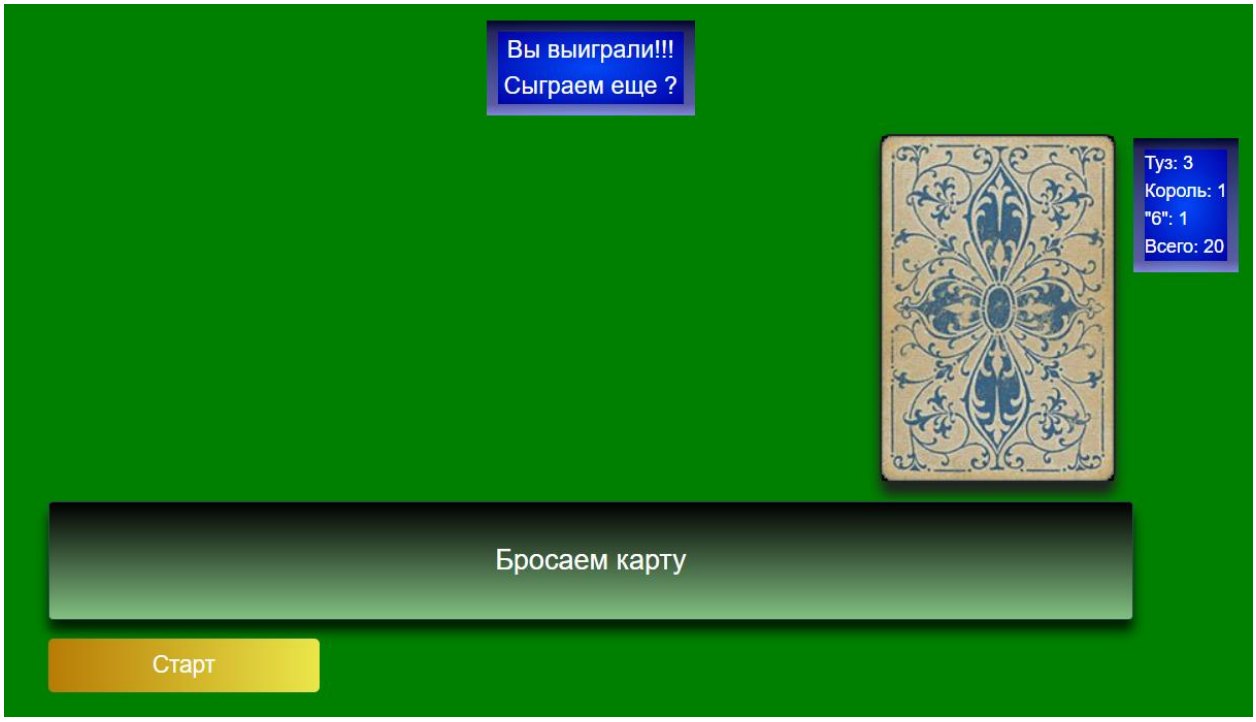

Рис 8. Пользователь проиграл.