Лабораторная работа № 5. JavaScript. jQuery

1. Цель работы

Ознакомиться с основными возможностями JavaScript Framework jQuery, научиться применять его базовые структуры в html документах.

2. Основы JavaScript

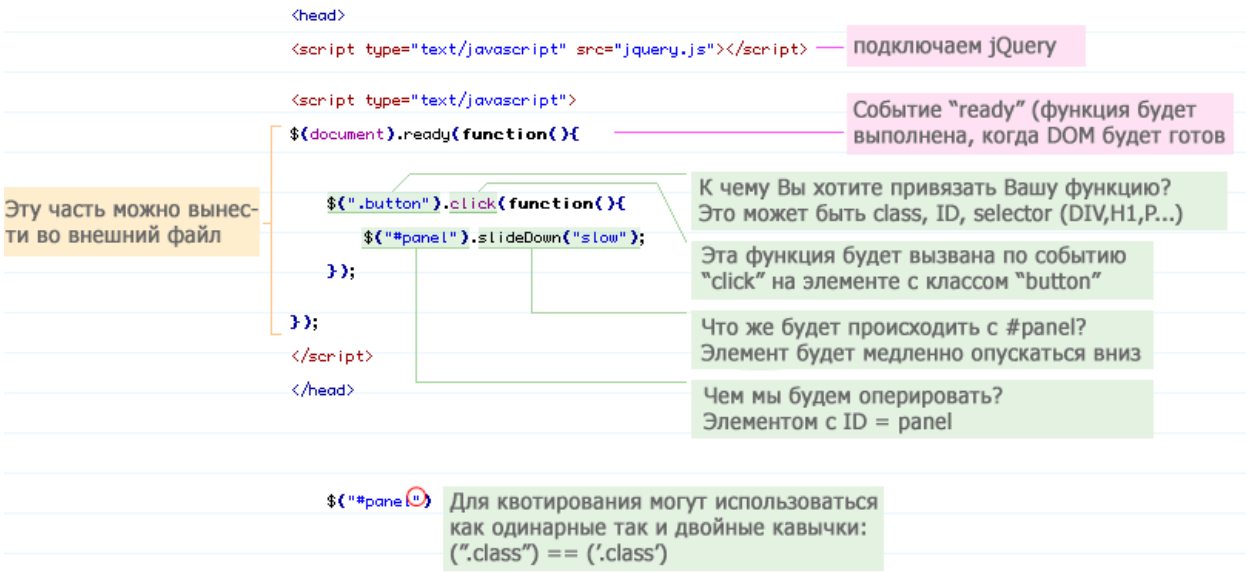
jQuery – библиотека Javascript, фокусирующаяся на взаимодействии Javascript и HTML. Библиотека jQuery помогает легко получать доступ к любому элементу DOM, обращаться к атрибутам и содержимому элементов DOM, манипулировать ими. Также библиотека jQuery предоставляет удобный API при работе с AJAX.

Как работает jQuery?

Для начала Вам понадобится сам фреймворк, его вы сможете скачать с домашней страницы проекта (http://jquery.com/), затем проинициализировать:

```
<script type="text/javascript" src="js/jquery.js"></script>
```

А основные моменты Вам поможет понять следующая диаграмма:



Как получить элемент с помощью jQuery?

Для того чтобы понимать как работает селектор Вам все же необходимы базовые знания CSS, т.к. именно от принципов CSS отталкивается селектор jQuery:

- $("#header") – получение элемента с id="header"
- $("h3″) – получить все <h3> элементы
- $("div#content .photo") – получить все элементы с классом ="photo" которые находятся в элементе div с id="content"
- $("ul li") – получить все <li> элементы из списка
- $("ul li:first") – получить только первый элемент <li> из списка

Выдвижная панель

Реализуем достаточно часто встречающийся на сайтах элемент, выдвижную панель. Она будет представлять блок, который появляется/исчезает по клику на кнопку.

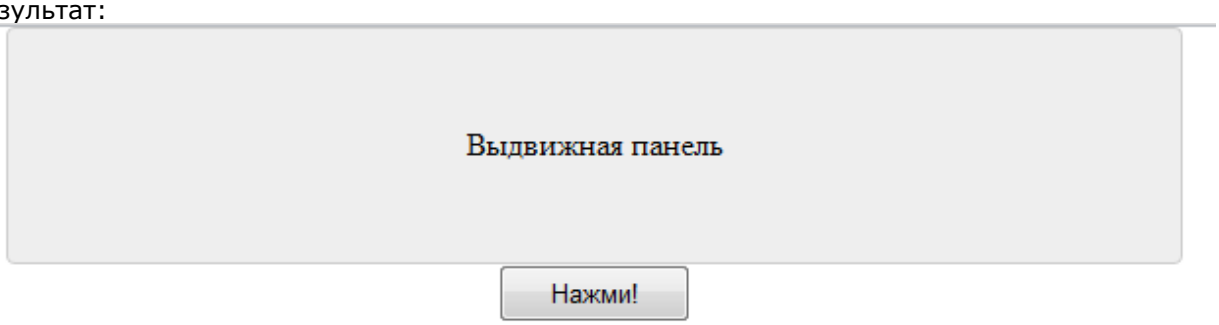
Код html:

```
<body>
    <div class="wrapper">
        <!-- Выдвижная панель -->
        <div id="panel">
            Выдвижная панель
        </div>
        <button class="btn-slide">Нажми!</button>
    </div>
</body>
```

Код Jquery:

```
<script type="text/javascript">
    $(document).ready(function(){
        $(".btn-slide").click(function(){
            $("#panel").slideToggle("slow");
            $(this).toggleClass("active");
        });
    });
</script>
```

Результат:



/* Необходимо создать файл с именем lab_1_1.html, где будут реализованы эти элементы */

Следующий пример покажет как можно красиво и легко убирать (растворять) элементы на странице.

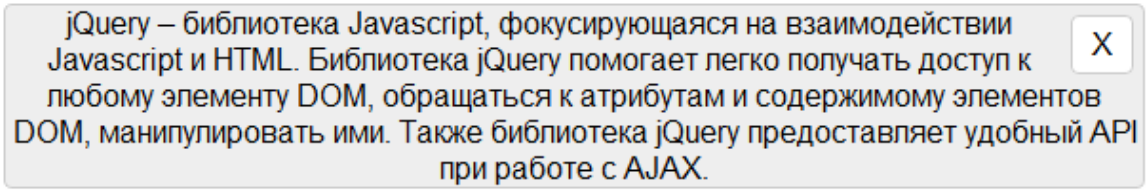
Код html:

```
<div class="pane">
    <span class="delete">X</span>
    <p>jQuery – библиотека Javascript, фокусирующаяся на взаимодействии
    Javascript и HTML. Библиотека jQuery помогает легко получать доступ
    к любому элементу DOM, обращаться к атрибутам и содержимому элементов
    DOM, манипулировать ими. Также библиотека jQuery предоставляет
    удобный API при работе с AJAX.</p>
</div>
```

Код Jquery:

```
//Удаление элемента со страницы
$(".pane .delete").click(function(){
    $(this).parents(".pane").animate({ opacity: "hide" }, "slow");
});
```

Результат:



/* Необходимо дополнить файл lab_1_1.html, этими элементами */

Связанная анимация

Рассмотрим более сложный пример, который лучше раскрывает функциональность Jquery. Нам необходимо управлять положением элемента на странице, его размером и прозрачностью.

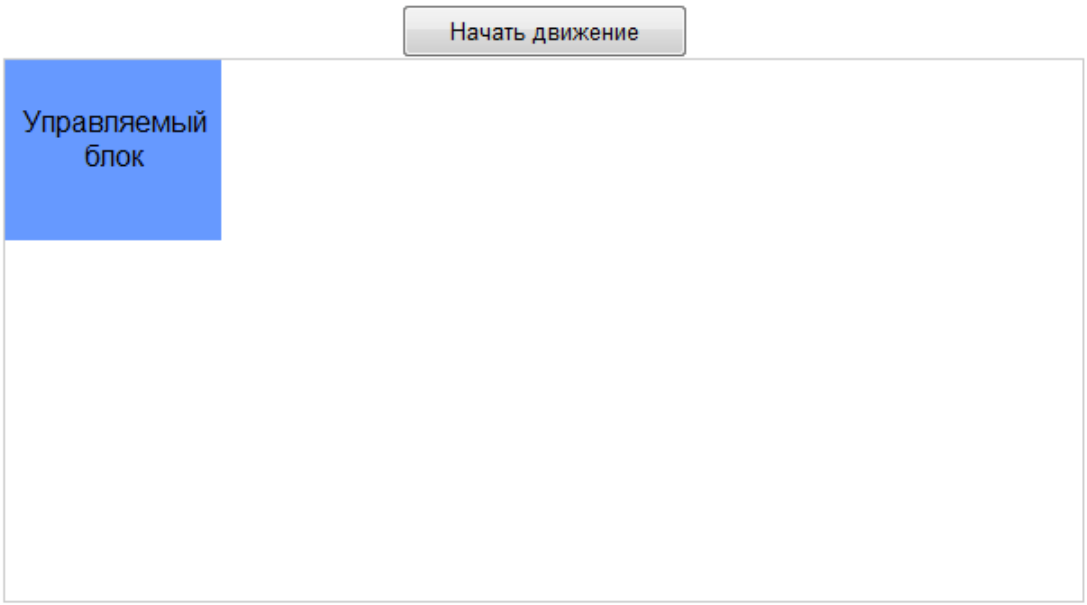
Код html:

```
<!-- Управление элементом на странице -->
<button class="run">Начать движение</button>
<div class="control_block_container">
    <div id="box">Управляемый блок</div>
</div>
```

Код Jquery:

```
//Управление элементом на странице
$(".run").click(function(){
    $("#box").animate({opacity: "0.1", left: "+=400"}, 1200)
    .animate({opacity: "0.4", top: "+=160", height: "40", width: "40"}, "slow")
    .animate({opacity: "1", left: "0", height: "50", width: "100"}, "slow")
    .animate({top: "0"}, "fast")
    .slideUp()
    .slideDown("slow")
    return false;
});
```

Результат:



Краткий разбор кода:
Line 1: привязываемся к событию click для элемента <a class=”run”>
Line 2: манипулируем элементом – уменьшаем его прозрачность до 0.1, наращиваем позицию left еще на 400px, со скоростью 1200 (milliseconds)
Line 3: затем медленно изменяем следующие параметры: opacity=0.4, top=160px, height=20, width=20; скорость анимации указывается вторым параметром : “slow”, “normal”, “fast” или в миллисекундах
Line 4: затем opacity=1, left=0, height=100, width=100, скорость – “slow”
Line 5: затем top=0, скорость – “fast”
Line 6: затем slideUp (с дефолтной скоростью анимации – “normal”)
Line 7: затем slideDown, скорость – “slow”
Line 8: возвращаем false для того чтобы браузер не перешел по ссылке
/* Необходимо дополнить файл lab_1_1.html, этими элементами */

Реализация блоков «Аккордеон»

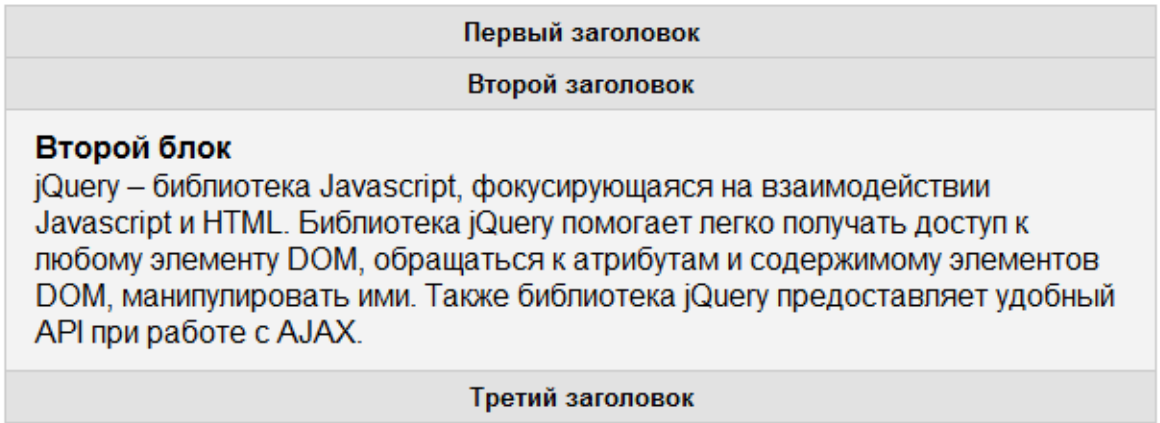
Рассмотрим очень распространенный в современном вебе элемент, блоки, организованные «гармошкой» или «аккордеоном».
Код Jquery:

```
//Реализация блоков "аккордеон"
$(".accordion h3:first").addClass("active");
$(".accordion p:not(:first)").hide();
$(".accordion h3").click(function(){
    $(this).next("p").slideToggle("slow")
    .siblings("p:visible").slideUp("slow");
    $(this).toggleClass("active");
    $(this).siblings("h3").removeClass("active");
});
```

Код html:

```
<div class="accordion">
    <h3>Первый заголовок</h3>
    <p><b>Первый блок</b><br />jQuery – библиотека Javascript,
    фокусирующаяся на взаимодействии Javascript и HTML.
    Библиотека jQuery помогает легко получать доступ
    к любому элементу DOM, обращаться к атрибутам и
    содержимому элементов DOM, манипулировать ими.
    Также библиотека jQuery предоставляет удобный API
    при работе с AJAX.</p>
    <h3>Второй заголовок</h3>
    <p><b>Первый блок</b><br />jQuery – библиотека Javascript,
    фокусирующаяся на взаимодействии Javascript и HTML.
    Библиотека jQuery помогает легко получать доступ
    к любому элементу DOM, обращаться к атрибутам и
    содержимому элементов DOM, манипулировать ими.
    Также библиотека jQuery предоставляет удобный API
    при работе с AJAX.</p>
    <h3>Третий заголовок</h3>
    <p><b>Первый блок</b><br />jQuery – библиотека Javascript,
    фокусирующаяся на взаимодействии Javascript и HTML.
    Библиотека jQuery помогает легко получать доступ
    к любому элементу DOM, обращаться к атрибутам и
    содержимому элементов DOM, манипулировать ими.
    Также библиотека jQuery предоставляет удобный API
    при работе с AJAX.</p>
</div>
```

Результат:



/* Необходимо дополнить файл lab_1_1.html, этими элементами */

Разберем записанный код:
Первой строчкой мы добавляем класс “active” первому элементу <h3> внутри (класс”active” отвечает за смену цвета блока).
Во второй строчке мы прячем все не первые <p> элементы внутри .
Когда происходит клик по заголовку <h3>, для следующего в нём элемента <p> будет применен эффект slideToggle, затем для всех остальных элементов <p> будет применен эффект slideUp.
Следующие действие изменяет класс заголовка на “active”, затем ищем все остальные заголовки <h3> и убираем у них класс “active”
Мы рассмотрели несколько примеров применения популярного фреймворка jquery.
В заключении рассмотрим инструментарий при работе с jquery
Выбор элементов по Id либо ClassName аналогично используемому в CSS

```
1  $('#sidebar');     // выбор элемента с id = sidebar
2  $('.post');        // выбор элементов с class = post
3  $('div#sidebar'); // выбор элемента div с id = sidebar
4  $('div.post');     // выбор элементов div с class = post
```

Выбор элементов из иерархии объектов DOM:

```
$('div span');          // выбор всех span элементов в элементах div
$('div').find('span');  // выбор всех span элементов в элементах div
$('div > span');        // выбор всех span элементов в элементах div,
                        //где span является прямым потомком div'a
$('div, span');         // выбор всех div и span элементов
$('span + img');        // выбор всех img элементов перед которыми
                        //идут span элементы
$('span ~ img');        // выбор всех img элементов после первого элемента span
$('#banner').prev();    // выбор предыдущего элемента от найденого
$('#banner').next();    // выбор следующего элемента от найденого
$('*');                 // выбор всех элементов
$('p > *');             // выбор всех потомков элементов p
$('p').children();      // --
$('p').parent();        // выбор всех прямых предков элементов p
$('* > p');             // выбор всех предков элементов p
$('p').parents();       // --
$('p').parents('div');  // выбор всех предков элемента p которые есть div
```

Фильтры:

```
$('div:first');      // выбираем первый div в доме
$('div:last');       // выбираем последний div в доме
$('div:not(.red)'); // выбираем div'ы у которых нету класса red
$('div:even');       // выбираем четные div'ы
$('div:odd');        // выбираем нечетные div'ы
$('div:eq(N)');      // выбираем div идущим под номером N в DOMe
$('div:gt(N)');      // выбираем div'ы, индекс которых больше чем N в DOMe
$('div:lt(N)');      // выбираем div'ы, индекс которых меньше чем N в DOMe
$(':header');        // выбо заголовоков h1, h2, h3 и т.д.
$('div:animated');  // выбор элементов с активной анимацией
$('div:contains(text)'); // выбираем div'ы содержащие текст
$('div:empty');          // выбираем пустые div'ы
$('div:has(p)');         // выбираем div'ы которые содержат p
$('div.red').filter('.bold') // выбираем div'ы которые содержат класс red и класс bold
$('div:hidden');         // выбираем скрытые div'ы
$('div:visible');        // выбираем видимые div'ы
```

3. Групповое задание

Создать кнопку, по клику на которой заданный элемент на странице будет вести себя следующим образом:

1	Зеленый прямоугольник 40*200 пикс	1. Отображается в левом верхнем углу окна браузера 2. Движется к правому верхнему углу браузера. 3. Меняет прозрачность до 0,5 4. Меняет цвет на красный 5. Движется на исходную позицию в левый верхний угол 6. Увеличивает размеры в 2 раза.
2	Серый квадрат 80*80 пикс посередине слово КВАДРАТ	1. Отображается в центре окна браузера. 2. Исчезает слово Квадрат 3. Появляется черная рамка толщиной 2 пикселя 4. Движется в правый верхний угол 5. Увеличивается в 3 раза 6. Движется в левый нижний угол по пути изменяя прозрачность до 0
3	Сама кнопка, по которой будет произведен клик	1. Кнопка уменьшается в размерах на 50% 2. Движется в правый верхний угол 3. Исчезает 4. Появляется в противоположном углу 5. Изменяет прозрачность до 0,5 6. Движется к верхней границе окна браузера

4. Варианты для индивидуальных заданий

1. По экрану хаотично перемещается фигура. Внутри фигуры отображаются ее текущие координаты. При клике на объекте левой кнопкой мышки он начинает увеличиваться до максимально допустимого размера. При повторном клике начинает уменьшаться до минимального. Предусмотреть ввод пользователем минимального и максимального допустимого размера фигуры.
2. По экрану хаотично перемещаются несколько фигур разного цвета и формы (количество фигур и виды их форм задает пользователь). При столкновении двух фигур между собой их цвет и форма должны изменяется на новый (случайно выбранный), отличный от их первоначальных.
3. На экране через заданный пользователем интервал в секундах появляются случайные фигуры (случайный цвет, форма, размер). Формы, цвета и диапазон размеров для создания фигур определяются настройками пользователя. При клике левой кнопкой мыши на фигуре, она должна исчезать с экрана.
4. Пользователь вводит слово. При нажатии на кнопку СТАРТ, слово разбивается на буквы, которые расплываются в стороны и начинают хаотично перемещаются по экрану, а при нажатии кнопки СТОП, буквы плавно собираются обратно в слово на исходной позиции.
5. По экрану хаотично перемещаются несколько фигур разного цвета и формы. При клике левой кнопкой мыши на фигуре, она распадается на случайно определяемое количество (от 2 до 4) таких же дочерних фигур, но размером на 50% меньше, которые продолжают дальнейшее движение самостоятельно.
6. По экрану хаотично перемещаются несколько фигур. Стартовое количество объектов задает пользователь. В центре экрана располагается черный прямоугольник. Если фигура попадает внутрь чёрного прямоугольника, она уничтожается. Наверху страницы вести учет существующих и уничтоженных объектов.
7. На экране через фиксированный интервал времени появляются красные и черные фигуры одинаковой формы и размера. В левом нижнем углу экрана располагается черный прямоугольный контейнер. Реализовать функцию перетаскивания фигур. При перемещении фигуры в контейнер она уничтожается. Наверху страницы вести учет уничтоженных в контейнере красных объектов.
8. По экрану хаотично перемещаются фигуры. При наведении курсора на фигуру она, продолжая движение, мгновенно увеличивается до максимального размера и остается в таком состоянии, пока на нее наведен курсор. Стартовое количество объектов, и максимальный размер в процентах от стартового задает пользователь.
9. По экрану хаотично перемещаются несколько фигур разного цвета и формы. При клике левой кнопкой мыши на фигуре, она скрывается, а наверху страницы появляется информация о скрытой фигуре (форма и цвет) и кнопка, позволяющая снова ее визуализировать. После нажатия на кнопку соответствующая фигура появляется, а кнопка с информацией исчезает.

10. Пользователь вводит в одном текстовом поле сообщение, а во втором запрещенные слова. При нажатии на кнопке «Цензура» все запрещенные слова в сообщении должны быть скрыты за красными прямоугольниками, которые появляются последовательно, по ходу текста, меняя свою прозрачность от 100% до 0%.

11. По экрану хаотично перемещается фигура. Внутри фигуры отображаются таймер отсчитывающий время, указывающее, сколько секунд фигура находится в движении. При клике на объекте левой кнопкой мышки он останавливается. При повторном клике снова начинает движение. Предусмотреть выбор пользователем фигуры и ее исходного размера.

12. По экрану хаотично перемещаются несколько фигур разного цвета и формы (количество фигур задает пользователь). При столкновении двух фигур между собой они сливаются в новую фигуру случайной и формы и цвета, отличных от их первоначальных. Наверху страницы вести учет существующих объектов.

13. По экрану хаотично перемещается фигура. При каждом клике левой кнопкой мыши на экране появляется прямоугольное препятствие. Реализовать для фигуры алгоритм уклонения, при котором она при своем движении будет стараться по возможности обходить все препятствия на своем пути.

14. По экрану хаотично перемещаются слова. При клике на словах левой кнопкой мышки они опускается вниз страницы, и выстраиваются последовательно в линию, формируя предложение. При нажатии на кнопку СБРОСИТЬ, предложение распадается на слова, которые снова начинают хаотично перемещаться по экрану.

15. По экрану хаотично перемещаются несколько фигур разного цвета и формы. При столкновении двух фигур между собой, каждая из них распадается на три небольших квадрата, которые разлетаясь в стороны, меняют свою прозрачность до 100% и исчезают.

16. По экрану хаотично перемещаются несколько фигур. Стартовое количество объектов задает пользователь. В левом нижнем углу экрана располагается черный прямоугольник, а правом верхнем красный. Если фигура попадает внутрь чёрного прямоугольника, она исчезает и появляется из красного.

17. На экране через фиксированный интервал времени появляются красные и черные фигуры одинаковой формы и размера. Слева по центру экрана располагается черный прямоугольный контейнер для уничтожения черных фигур, а справа красный контейнер для уничтожения красных. Реализовать функцию перетаскивания фигур. При перемещении фигуры в соответствующий по цвету контейнер она уничтожается.

18. По экрану хаотично перемещаются фигуры. Фигуры то увеличиваются, то уменьшаются в размере. При клике левой кнопкой мышки на объекте, который находился в процессе увеличения, он удаляется с экрана. При клике левой кнопкой мышки на объекте, который находился в процессе уменьшения, ничего не происходит. Наверху страницы вести учет оставшимся фигурам.

19. На экране сверху вниз непрерывно падают пронумерованные фигуры разного цвета. Как только фигура визуализируется на экране, наверху страницы появляется о ней информация (номер и цвет) и кнопка, позволяющая изменить на случайный ее цвет. Как только объект опускается за пределы нижней границы информация, и кнопка соответствующая данной фигуре исчезает.

20. Пользователь вводит в текстовом поле сообщение указывая начало и конец сворачиваемого/разворачиваемого абзаца тегами <region> и </region>. По нажатию на кнопку «СОЗДАТЬ» ниже по странице формируется текст с маркерами, обеспечивающими возможность свернуть или развернуть соответствующий абзац текста.

21. По экрану хаотично перемещается фигура. Внутри фигуры, с определенным интервалом времени, отображаются случайные буквы английского алфавита. При клике на объекте левой кнопкой мышки буквы меняют свой цвет и размер. Предусмотреть выбор пользователем интервала смены букв в секундах, выбор допустимых цветов и размеров для букв.

22. По экрану хаотично перемещаются несколько фигур разного цвета (количество фигур задает пользователь). При столкновении двух фигур одна из них уничтожается (какая именно останется, определяется случайным образом). Внутри каждой фигуры отображаются счетчик уничтоженных ей объектов.

23. Пользователь управляет перемещением фигуры по экрану (w – вперед, s – назад, a – влево, d – вправо). Кроме фигуры пользователя по экрану хаотично перемещаются еще несколько фигур «противников». Реализовать алгоритм преследования, при котором если фигура пользователя попадает в зону видимости противника, то он начинает ее преследование.

24. По экрану хаотично перемещаются буквы. При столкновении они объединяются, порождая слоги или слова. Стартовое количество букв, и их размер задает пользователь. При нажатии на кнопку СБРОСИТЬ, все слоги и слова распадаются на буквы, которые снова начинают хаотично перемещаться по экрану.

25. На экране через фиксированный интервал времени появляются фигуры, которые начинают хаотично перемещаться по экрану, с каждой секундой прозрачность фигур увеличивается. При достижении прозрачности 100% фигура самоуничтожается. При клике левой кнопкой мыши на фигуре, она должна мгновенно исчезать с экрана. Наверху страницы вести учет уничтоженных пользователем объектов и самоуничтожившихся объектов.

26. По экрану хаотично перемещаются красные и черные фигуры. Стартовое количество объектов каждого цвета задает пользователь. Слева по центру экрана располагается черный прямоугольник, а справа красный. Если фигура попадает внутрь чёрного прямоугольника, она меняет цвет на черный. Если фигура попадает в красный прямоугольник, становится красной. Наверху страницы вести учет красным и черным фигурам.

27. На экране через фиксированный интервал времени появляются красные и черные фигуры. Также на экране располагается черный прямоугольный контейнер со счетчиком для хранения черных фигур и красный контейнер со счетчиком для хранения красных. Реализовать функцию

перетаскивания фигур и организовать корректную работу счетчиков подсчитывающих количество соответствующих фигур в контейнерах.

28. По экрану хаотично перемещаются фигуры. В центре экрана располагается окружность. Внутри каждой фигуры расположена цифра 1 или 0. Если объект попадает внутрь окружности, цифра становится равна 1, а размер фигуры увеличивается. При выходе за пределы окружности цифра становится равна 0, а размеры фигуры возвращается к первоначальным.

29. По экрану хаотично перемещаются пронумерованные фигуры. Через определенный интервал времени каждая фигура меняет свой цвет, форму и размер. Наверху страницы размещена таблица с актуальной информацией о фигурах (номер, размер, форма и цвет) и кнопка, позволяющая заблокировать/разблокировать все изменения соответствующей фигуры.

30. Пользователь выбирает тип фигуры и вводит ее название. По нажатию на кнопку «ДОБАВИТЬ» данная фигура с соответствующей подписью появляется на случайной позиции рабочего поля страницы. Реализовать функцию выбора с прямоугольной подсветкой нескольких фигур на рабочем поле и возможность их группового удаления по нажатию на кнопку «УДАЛИТЬ».