Name ____ Role __________

Bouncy Ball

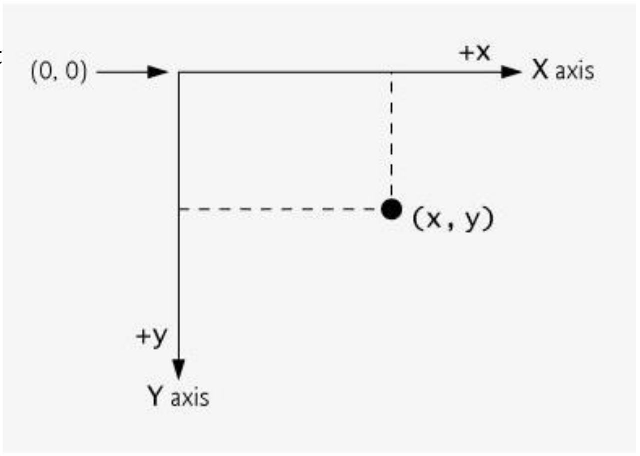
Your Tasks (Mark these off as you go)

- ☐ Understand the JavaScript coordinate system
- ☐ Write *if* statements to keep a ball in bounds
- ☐ Write *if* statements to increase and decrease the size of a ball
- ☐ Write code to create a random speed for the ball
- ☐ Write code to create a random starting point for the ball
- ☐ Write code to create a random custom color
- ☐ Receive credit for this lab guide

☐ Understand the JavaScript coordinate system

In this lab you will write code to manipulate an image in the browser. Before we get started it is important to get orientated with the JavaScript coordinate system.

- The upper-left corner of the browser has the coordinates (0, 0).
- The X-coordinate increases as you move left to right horizontally across the screen
- The Y-coordinate increases as you move down from the top of the screen.
- Coordinate units are measured in pixels.



The following function creates a ball and displays it on the screen at coordinates (xPos, yPos).

```
function makeBall(){
    var ball = document.createElement("div");
    ball.style.borderRadius = "50%";
    ball.style.width = diameter+"px";
    ball.style.height = diameter +"px";
    ball.style.backgroundColor = "RED";
    ball.style.position = "absolute";
    ball.style.left = xPos + "px";
    ball.style.top = yPos + "px";
    gameDiv.append(ball);
    ball.addEventListener("click", addPoint);
    ball.addEventListener("mouseover", changeMouse);
    return ball;
}
```

Declaring and initializing `xPos` and `yPos` to 0 displays the ball at coordinates (0, 0).

```
var xPos = 0, yPos = 0;
```

The number of pixels that the ball can move each time the screen is refreshed is defined in terms of the variables xDelta and yDelta,

```
int xDelta = 5, yDelta = 5;
```

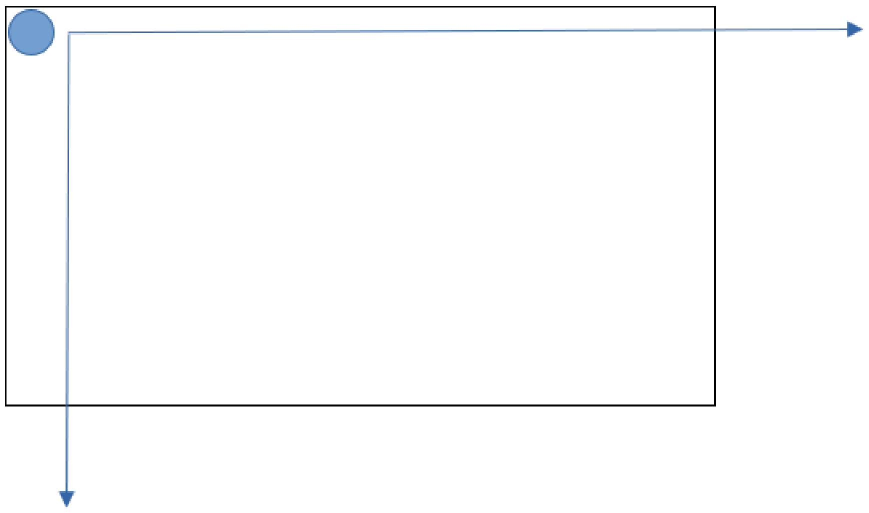
(a) Write code that could be used to move the ball horizontally across the screen. The object should move xDelta each time the screen is refreshed.

(b) Write code that could be used to move an object vertically down the screen one pixel at a time. The object should move yDelta each time the screen is refreshed.

(c) Write code that could be used to move an object diagonally across the screen one pixel at a time.

☐ Write *if* statements to keep a ball in bounds

Consider a ball confined to the box below. In the previous example, you wrote code that could be used to increment an object by an integer amount defined as either xDelta or yDelta. However, if you continuously increment the x and y coordinates by Xdelta or yDelta, eventually the ball is going to disappear off the screen.



Consider a ball confined to a box with the following dimensions `var boxWidth = 500, boxHeight = 500;` (a) Write code that could be used to keep the ball in bounds in the horizontal direction. You will need to write if statements to switch the direction the ball is moving once it reaches the horizontal boundaries (500 or 0) (b) Do the same for the vertical direction.

☐ Write *if* statements to increase and decrease the size of a ball

Now that we have our ball staying in bounds, let’s make our ball grow and shrink as it traverses across the screen.

The following function creates a ball with a diameter. Note, the width and height are used to specify the radius.

```
function makeBall(){
    var ball = document.createElement("div");
    ball.style.borderRadius = "50%";
    ball.style.width = diameter+"px";
    ball.style.height = diameter +"px";
    ball.style.backgroundColor = "RED";
    ball.style.position = "absolute";
    ball.style.left = xPos + "px";
    ball.style.top = yPos + "px";
    gameDiv.append(ball);
    ball.addEventListener("click", addPoint);
    ball.addEventListener("mouseover", changeMouse);
    return ball;
}
```

Consider the following variables which define the amount the ball grows each time the screen is refreshed, along with the balls minimum and maximum diameter.

```
var diameter = 10, dDelta = 1, minDiameter = 5, maxDiameter = 50;
```

Write code that could be used to increase the size of the ball until it reaches a maximum, then decrease the size of the ball until it reaches a minimum.

☐ Write code to create a random speed for the ball

Each time the screen is refreshed the ball will move in the x and y direction the distance specified by xDelta and yDelta. The large xDelta and yDelta, the larger the distance, and the faster the movement.

Consider the following variables below,

```
var deltaRange = 20;
var xDelta = 5, yDelta = 5;
```

Write code that could be used to create a random speed for the ball. The xDelta and yDelta values should be replaced with random deltas which range from -deltaRange/2
to +deltaRange/2

☐ Write code to create a random starting point for the ball

Rather than having our ball start moving from the same location each time the program is run, we can create a random location. The random location can be in any range from x = 0 to xPos = boxWidth or y = 0 to y = boxHeight.

The code below would initialize a ball in the middle of the box,

```
      var boxWdith= 500; // width of the box
      var boxHeight = 500; //height of the box
      xPos = boxWidth / 2;
      yPos = boxHeight / 2;
```

Write code that could replace the xPos and yPos values above such that the ball would be initialized at some random point in the box.

☐ Write code to create a random custom color

We have already used JavaScript's built-in colors to style elements on the screen. For example, the code below styles the background of the ball to red,

```
ball.style.backgroundColor = "RED";
```

In addition to the built in colors (red, blue, black, yellow, etc), JavaScript also allows us to create custom colors. These custom colors have three components: red, green, and blue. Each of the components has a value between 0 and 256 (256 is not inclusive). The following code, for example, could be used to color our ball with a custom color,

```
var R = 50, G = 100, B = 10;
var customColor = "RGB("+R+","+G+","+B+")";
ball.style.backgroundColor = customColor;
```

Write code that could be used to generate a random custom color. To do this, first create a random value for each of the variables R, G, and B that ranges from 0 up to 256. Next, use your custom color to style ball as shown above.

☐ Receive Credit for this lab guide

Submit this portion of the lab to Pluska to receive credit for the lab guide. Once received, your completed code challenges will also be graded and will count towards your final lab grade.