Animation: Bouncing Ball

Objective(s):

- You will learn to animate a bouncing ball
- You will incorporate conditional if-statements to control the animation

Reference(s):

- In case you wish to look at other functions, use the Processing Java Reference
- Processing's Interactivity Tutorial: https://processing.org/tutorials/interactivity/

Directions:

1. Create a Processing sketch in Java mode (and save it as JavaBouncingBall)
2. Paste the following code into your sketch:

```
//variables that all methods can use
int x = 150;
int y = 150;
int r = 25;
int dy = 1; //delta y as in small change in y

//void is the return type of the setup() method
//setup() does not return anything, to its return type is void
void setup()      //setup() runs just once
{
  size(300, 300);  //size() must be first line in setup()
}

//the draw() method does not return anything, its return type is void
void draw()           //draw() runs over-and-over-and-over again
{
  background(126);
  circle(x, y, 2*r);
  y += dy;
}
```

3. What do you see when you run your animation?
4. Add some color to the background and ball

5. Let's make the ball bounce at the bottom of the screen. Add the following if-statement to the draw() function:

```
if (y > 300)
{
      dy *= -1;
}
```

6. Run it and you'll see that the bounce is not perfect. Fix the code so that it bounces when the edge of the ball hits the bottom of the screen.
7. The ball still flies out the top of the screen, so add an or-statement to the if that will detect the ball at the top of the screen
8. Add code to make the ball also move horizontally, so that the ball is bouncing all over the screen. (i.e. add x-direction movement)
9. Complete the submission requirements (i.e. narrated program demonstration and code walk-through)

For fun...see what happens when you comment out background() in the draw() function

Additional Challenges:

1. Add variables for color (r, g, b) and manipulate the color as animation runs
2. Speed the ball up when the ball bounces off a wall
3. Manipulate the size of the ball as the animation runs
4. Randomize the starting direction of the ball (you will need to lookup documentation for the random() function)
5. Add another bouncing ball to the screen
6. Add collision detection and bouncing between the two balls