Name ____

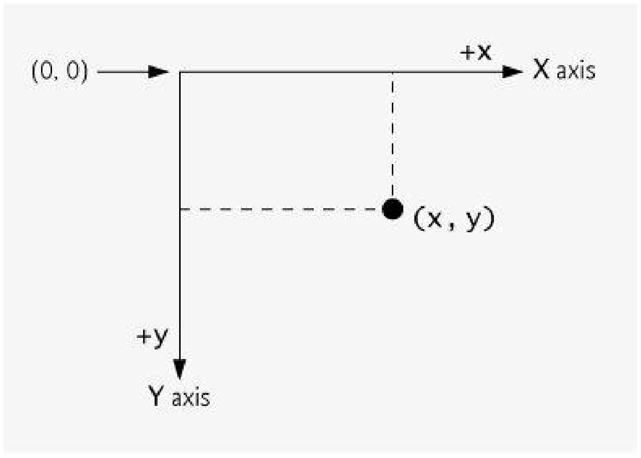
Bouncy Ball

Your Tasks (Mark these off as you go)

- ❑ Understand the Java 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 Java coordinate system

In this lab you will write code to manipulate an image in a Graphical User Interface (GUI). Before we get started it is important to get orientated with the Java coordinate system.



The upper-left corner of the GUI component 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.

Consider an object located at the following coordinates

```
int x = 0, y = 0;
```

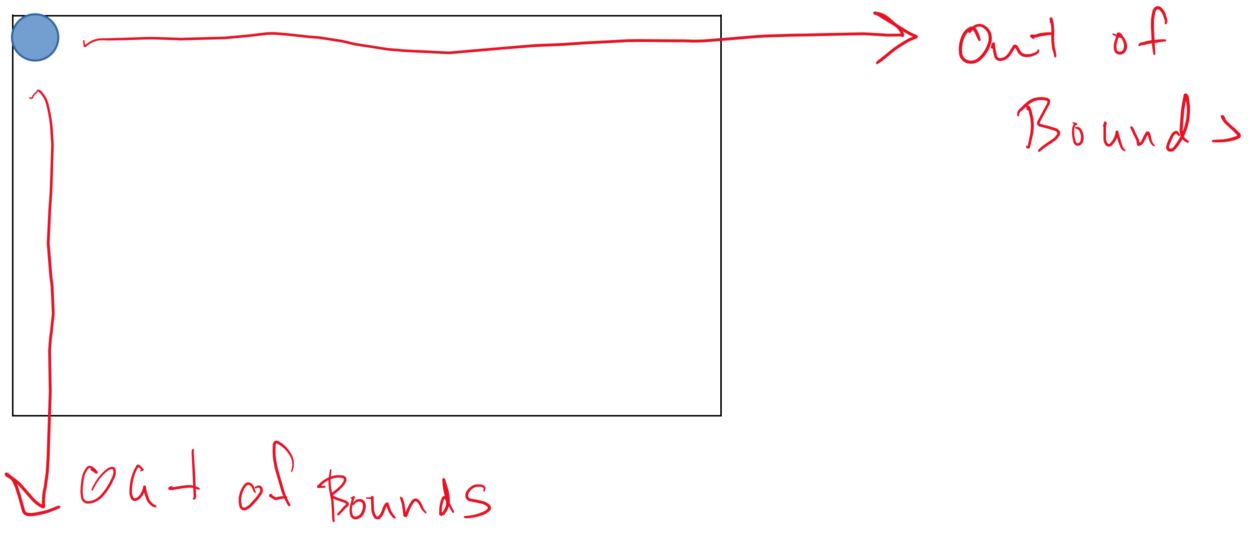
The number of pixels that the object can move each time the screen is refreshed is defined in terms of the variables xDelta and yDelta,

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

(a) Write code that could be used to move an object 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. 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

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

Consider a ball confined to the screen below. In the previous example, you wrote code that could be used to increment an object by an int 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 the following screen where,

```
int width = 600, height = 400;
```

(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 x boundaries (`width` 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.

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

int radius = 10, int rDelta = 1, MIN_RADIUS = 1, and MAX_RADIUS = 20;

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 larger xDelta and yDelta, the larger the distance, and the faster the movement.

Consider the following variables beow

```
final int DELTA_RANGE = 20;
int xDelta = 5, yDelta = 5;
```

Write code below 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 `−DELTA_RANGE/2` to `+DELTA_RANGE/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 x = width or y = 0 to y = width.

The code below would initialize a ball in the middle of the screen.

```
      final int INIT_WIDTH = 600; // width of the screen
      final int INIT_HEIGHT = 400;// height of screen
      x = INIT_WIDTH / 2;
      y = INIT_HEIGHT / 2;
```

Write code that could replace the x and y values above such that the ball would be initialized at some random point on the screen.

❑ Write code to create a random custom color

The Color class in Java allows us to create colors. For example, the code below creates the color "red". Notice in the code below, color is a Color type variable. Color.RED is the value we assigned to the variable color.

```
Color color = Color.RED;
```

Once a color is assigned to our color variable it can be used to color objects on our screen. In addition to the built in colors (red, blue, black, yellow, etc), Java also allows us to create custom colors. These custom colors have three components: red, green, and blue. Each of the components has value between 0 and 256 (256 is not inclusive). The following code could be used to create a custom color,

```
int R = 100, G = 141, B = 4;

Color customColor = new Color(R, G, B);
```

Consider the color declared below, `Color color = Color.BLUE;` Write code that could be used to generate a random custom color. To do this, first create a random value for each of the int variables R, G, and B that ranges from 0 up to 256. Next, create a new color using the syntax illustrated 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.