**Lab 3 Questions**

1. **Comment out/Remove the stddraw.clear function call:**

* The ball just creates a line behind it as the previous frames are not clearing. Since the ball is stuck in a strict back and forth motion, The ball cannot leave this singular line that it creates. This is not what is supposed to happen, you are supposed to use the stddraw.clear function so that the previous frame gets cleared. That way each frame creates the illusion that a ball is bouncing back and forth.

1. **Modify the velocity. Try at least 3 different values, both increasing and decreasing:**

* Obviously, when you increase the velocity the ball speeds up, and when you decrease the velocity the ball slows down. I used the values .005, .09, and .08, These were all significant speed changes to the default velocity value.

1. **Replace the point with a line, which has its first point fixed at (0, 0) and which uses the bouncing point as its end point.**

* This creates a line with a fixed point at (0, 0), but the rest of the line moves in the pattern that the ball normally would. This creates a flexible line that bounces back and forth on the screen, while one end is still fixed at the origin.
* The following is the code that I edited from the packet to get this result:

import stddraw

x\_coord = 0

y\_coord = .5

velocity = .01

while True:

# Clear the canvas (so that only one dot appears)

stddraw.clear()

# Draw the point

stddraw.line(0, 0, x\_coord, y\_coord)

# Display the background canvas

stddraw.show(10)

# Update the coordinates of the point

# in the next frame.

# If we have hit either boundary, reverse the direction

# of motion (to keep the dot on the canvas).

if (x\_coord + velocity) >= 1 or (x\_coord + velocity) <= 0:

velocity = -velocity

# Get new coordinate

x\_coord = x\_coord + velocity