



Animation

Chris Piech + Mehran Sahami
CS106A, Stanford University

Turing Award Winner



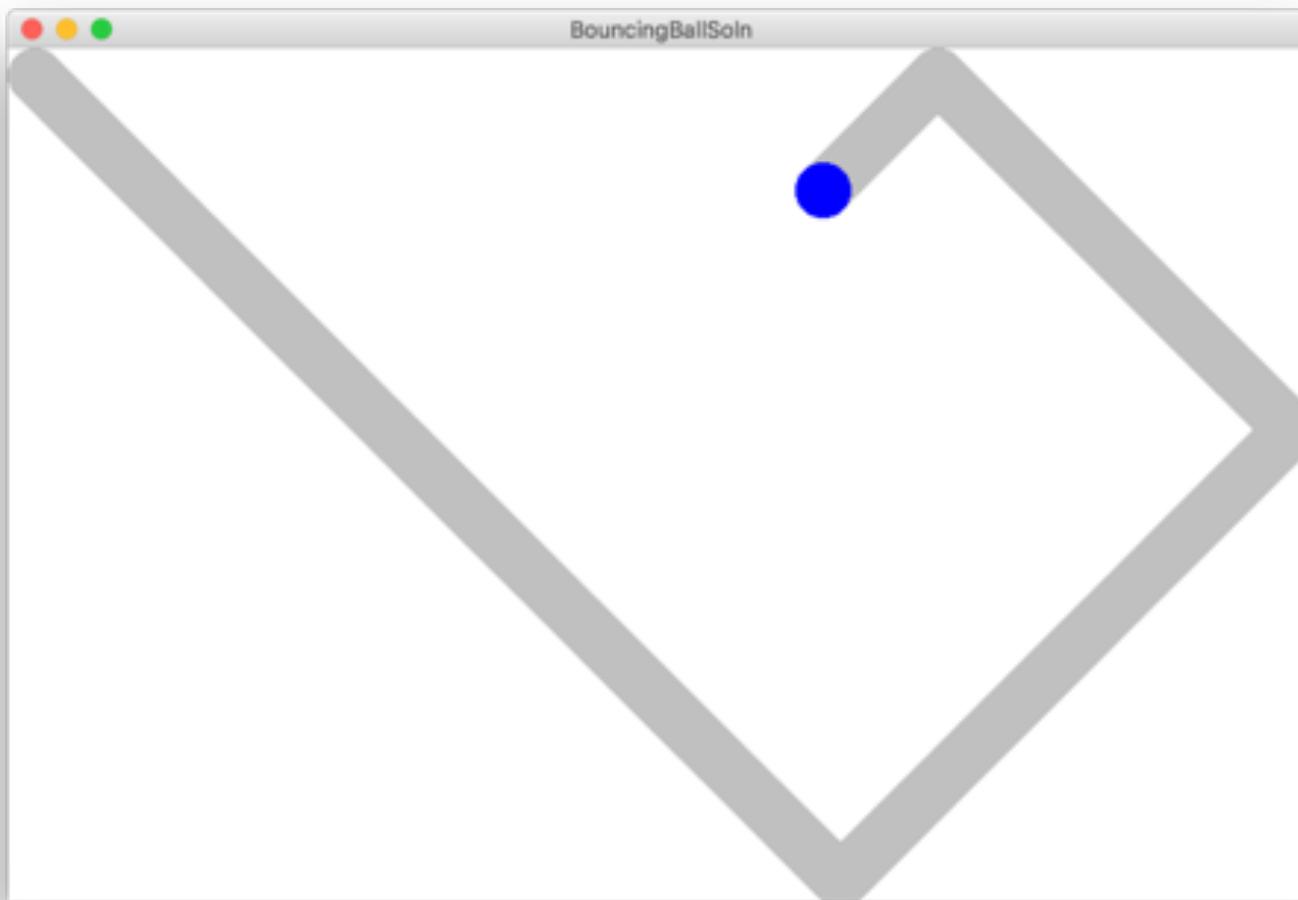
- **Turing Award** is like the nobel prize in CS
- Professor here at Stanford (CS107E, CS348)
- Founding employee at Pixar
- Wrote RenderMan, won 3 Academy Awards
- And just a really wonderful human.

Learning Goals

1. Write animated programs



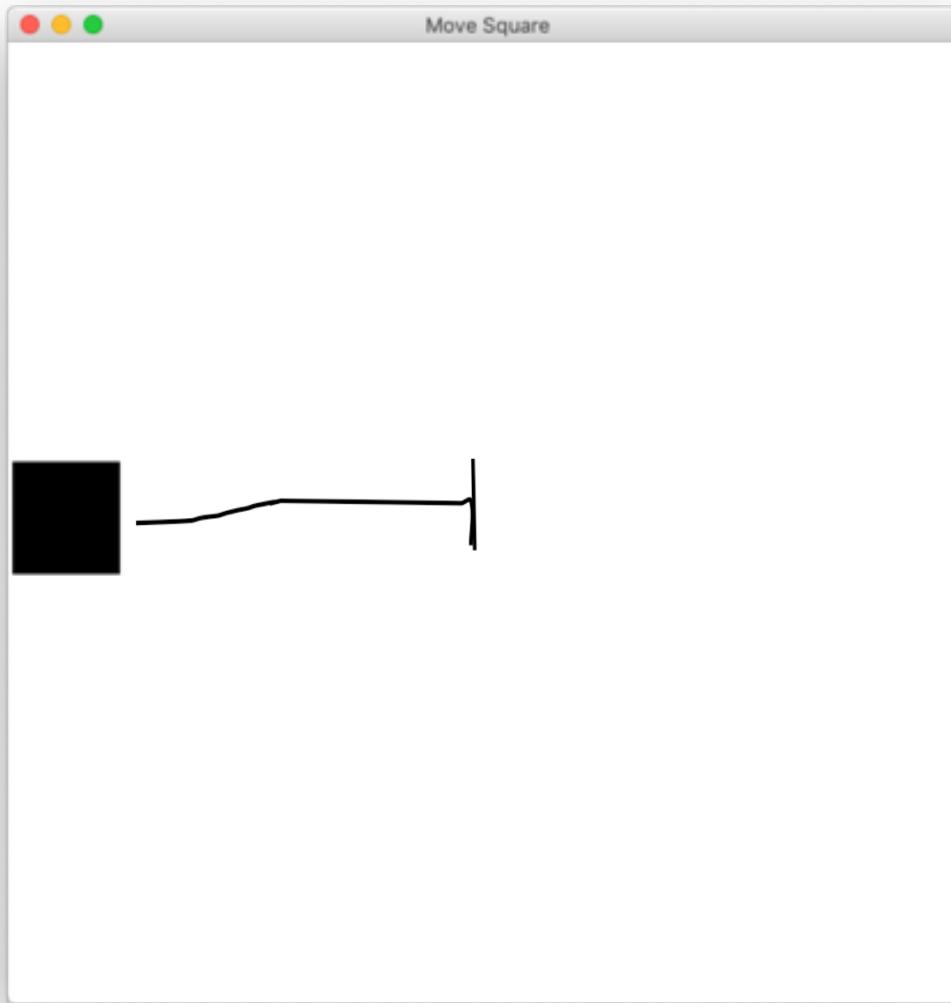
You will be able to write Bouncing Ball



Great foundation



Move to Center



In our last episode...

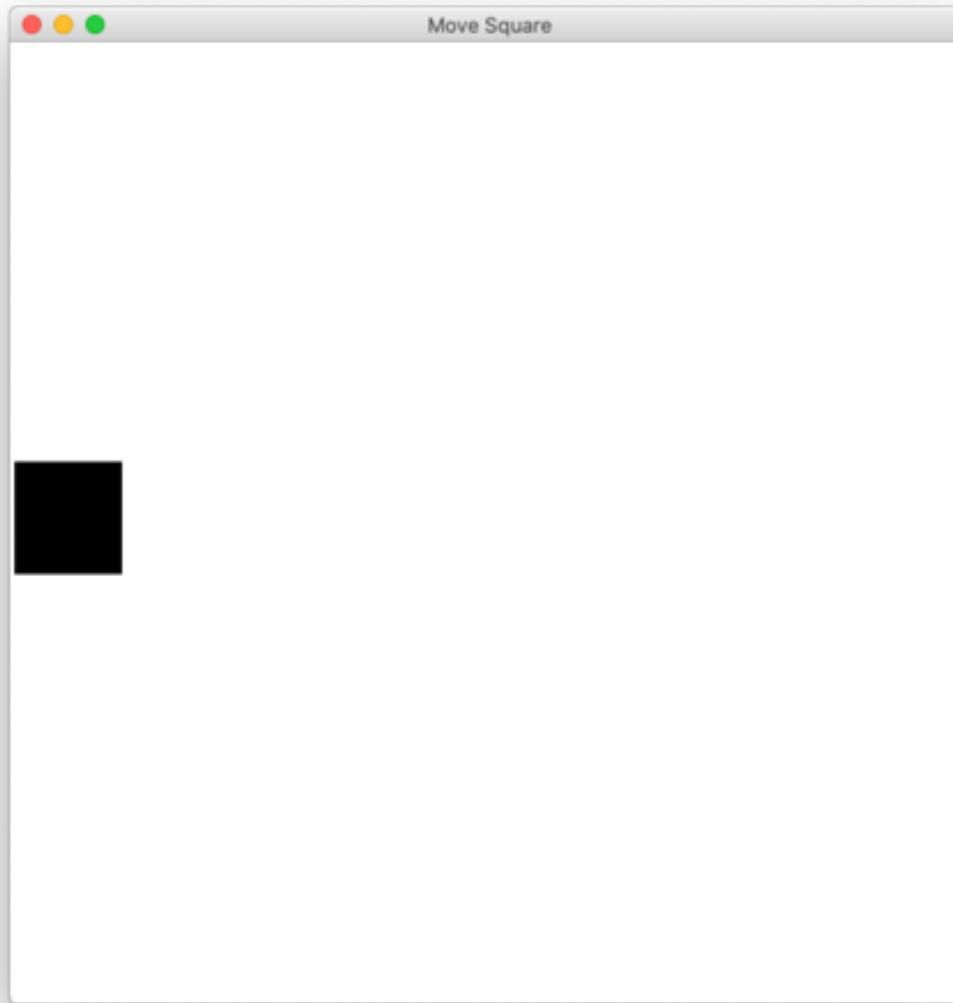
Graphics from tkinter (aka tk)

```
import tkinter
```

```
# we write this for you, and include it
# in all of your projects!
canvas = Canvas(width, height, title)
```



Add square



Graphics from tkinter (aka tk)

```
import tkinter
```

```
# we write this for you, and include it
# in all of your projects!
def Canvas(width, height, title):
```



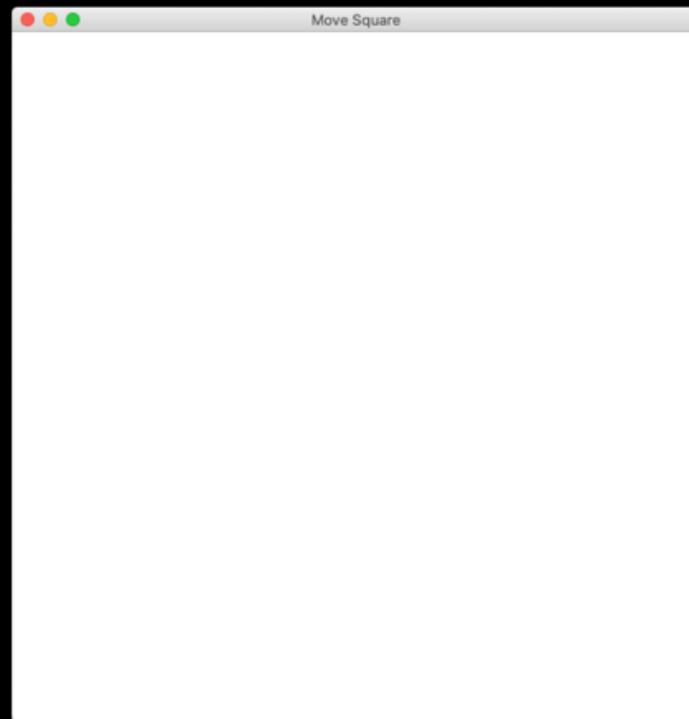
```
def main():
    canvas = Canvas(CANVAS_WIDTH, CANVAS_HEIGHT, 'Move Square')
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y, fill='black')
    canvas.mainloop()
```



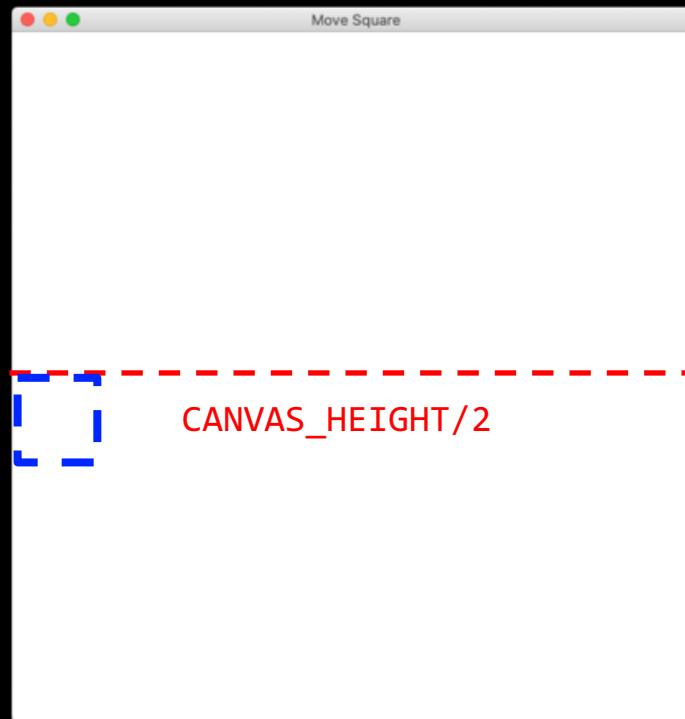
```
def main():
    canvas = Canvas(CANVAS_WIDTH, CANVAS_HEIGHT, 'Move Square')
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y, fill='black')
    canvas.mainloop()
```



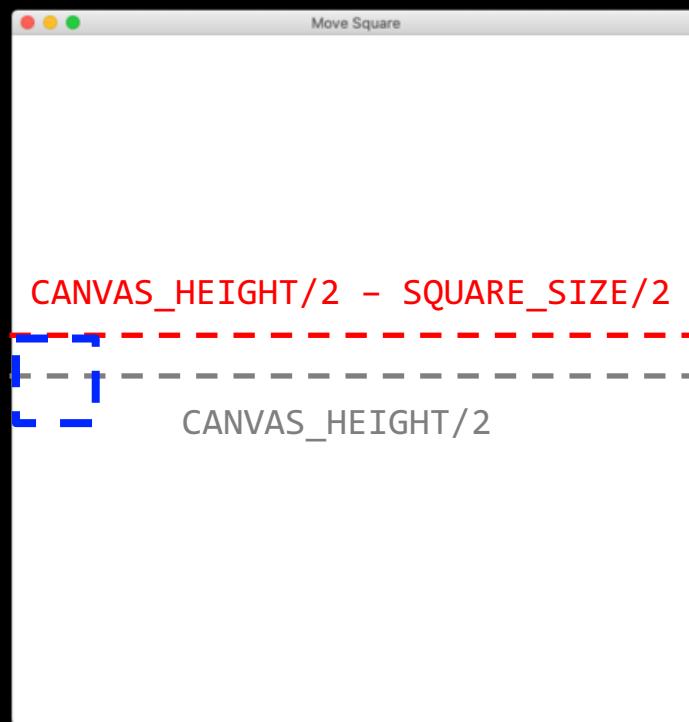
```
def main():
    canvas = Canvas(CANVAS_WIDTH, CANVAS_HEIGHT, 'Move Square')
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y, fill='black')
    canvas.mainloop()
```



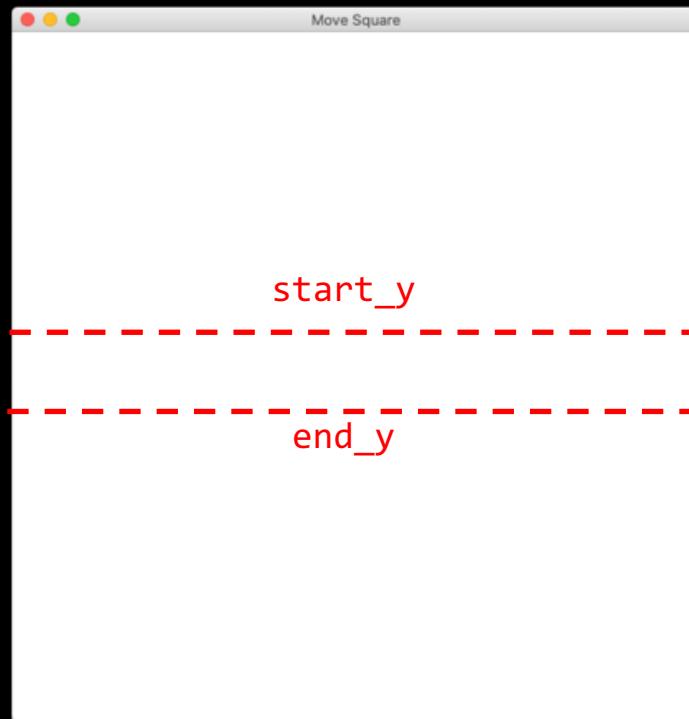
```
def main():
    canvas = Canvas(CANVAS_WIDTH, CANVAS_HEIGHT, 'Move Square')
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y, fill='black')
    canvas.mainloop()
```



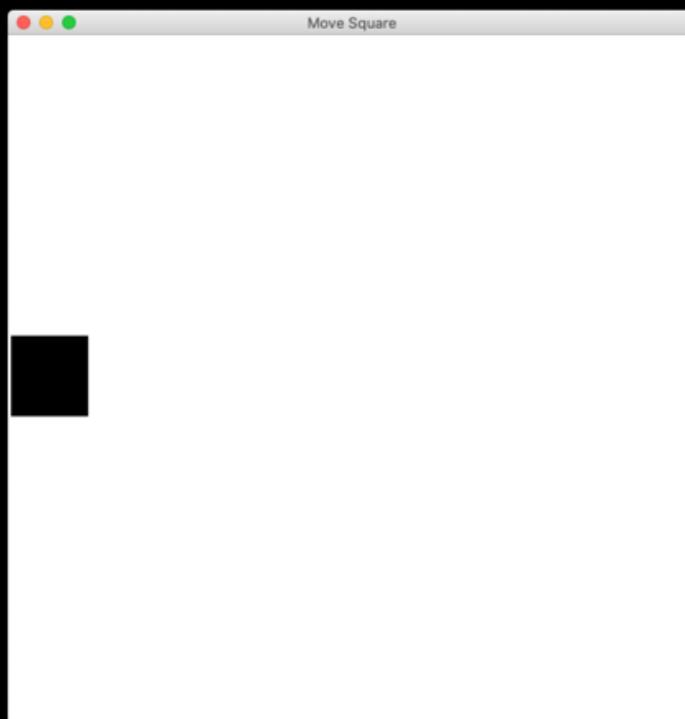
```
def main():
    canvas = Canvas(CANVAS_WIDTH, CANVAS_HEIGHT, 'Move Square')
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y, fill='black')
    canvas.mainloop()
```



```
def main():
    canvas = Canvas(CANVAS_WIDTH, CANVAS_HEIGHT, 'Move Square')
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y, fill='black')
    canvas.mainloop()
```

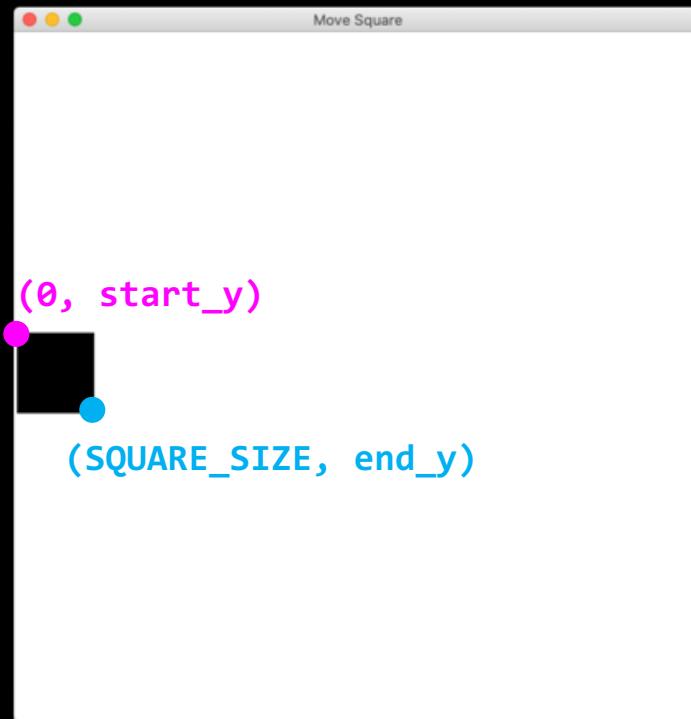


```
def main():
    canvas = Canvas(CANVAS_WIDTH, CANVAS_HEIGHT, 'Move Square')
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y, fill='black')
    canvas.mainloop()
```



Some “heavy duty” variables allow you to call functions on them

```
def main():
    canvas = Canvas(CANVAS_WIDTH, CANVAS_HEIGHT, 'Move Square')
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y, fill='black')
    canvas.mainloop()
```



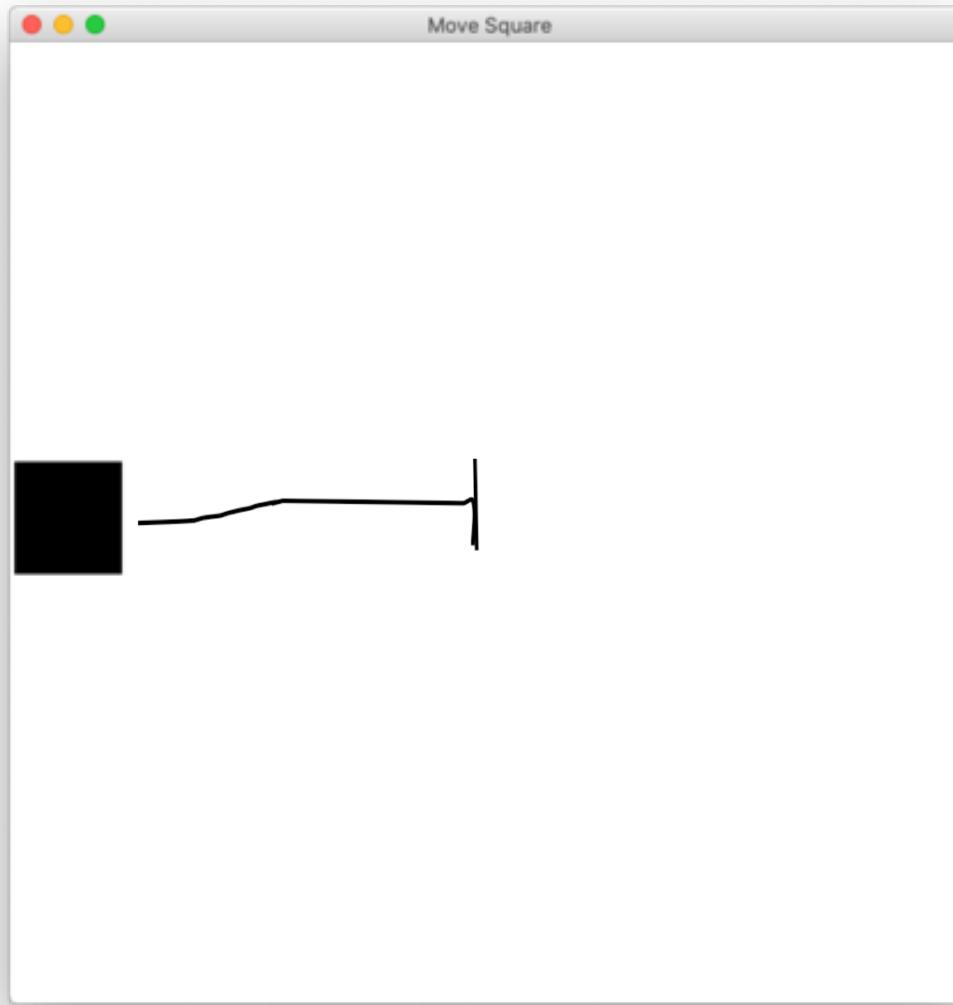
You're now all graphics programmers!



End review...

How do movies or games
animate?

Move to Center



* That's not quite toy story, but it is a start...



Animation Loop

```
def main():
    # setup

    while True:
        # update world

        # pause
        time.sleep(DELAY)
```



Animation Loop

```
def main():
    # setup

    while True:
        # update world

        # pause
        time.sleep(DELAY)
```

Make all the variables you need.



Animation Loop

```
def main():
    # setup
    while True:
        # update world
        # pause
        time.sleep(DELAY)
```

The animation loop is a repetition of heartbeats



Animation Loop

```
def main():
    # setup

    while True:
        # update world
        # pause
        time.sleep(DELAY)
```

Each heart-beat, update
the world forward one
frame



Animation Loop

```
def main():
    # setup

    while True:
        # update world

        # pause
        time.sleep(DELAY)
```

If you don't pause,
humans won't be able
to see it



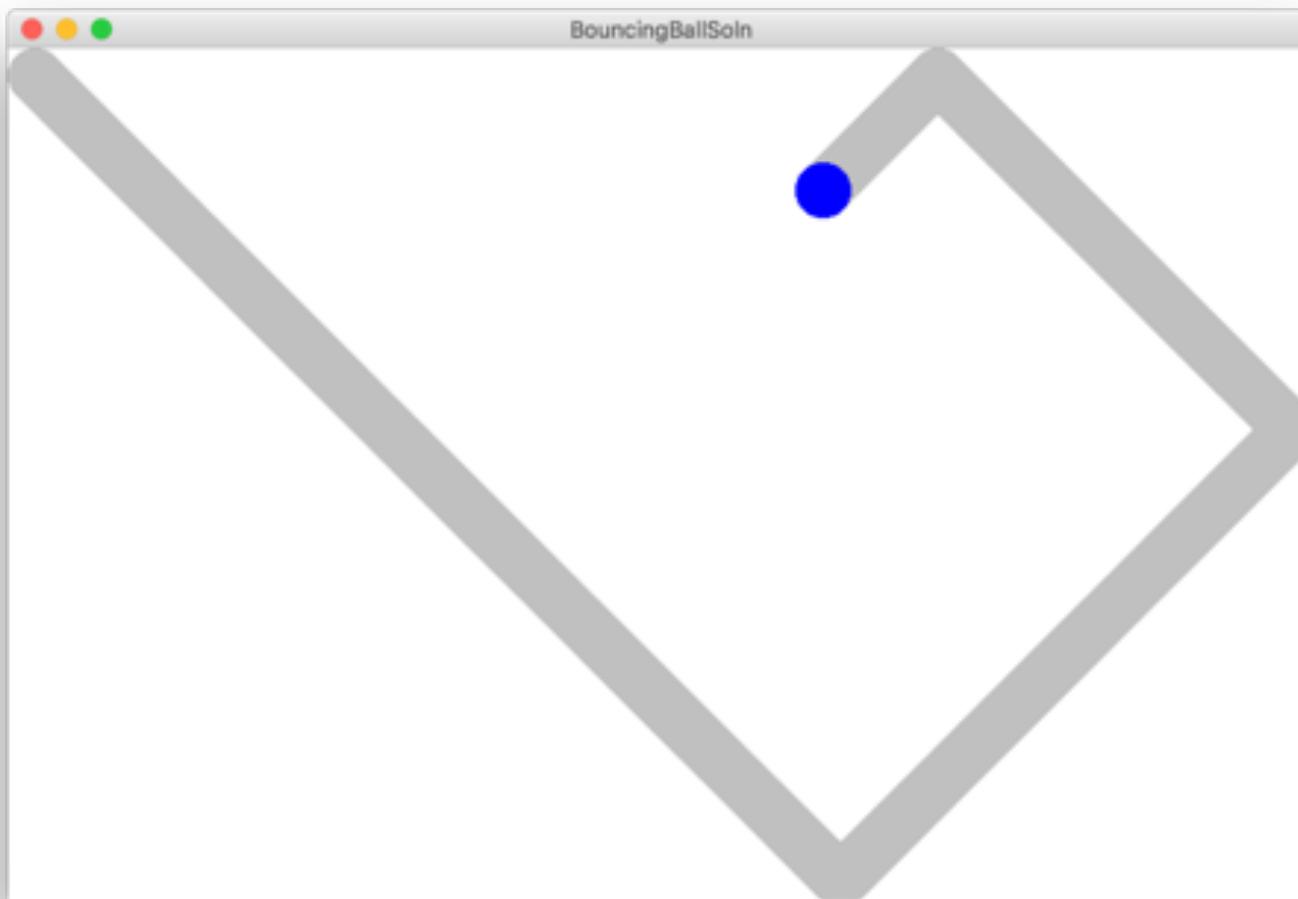
Move To Center

```
def main():
    # setup
    canvas = Canvas(CANVAS_WIDTH, CANVAS_HEIGHT)
    rect = canvas.create_rectangle(0, 0, 100, 100)
    while not is_past_center(canvas, r):
        # update world
        canvas.move(rect, 1, 0)
        canvas.update()
        # pause
        time.sleep(DELAY)
```

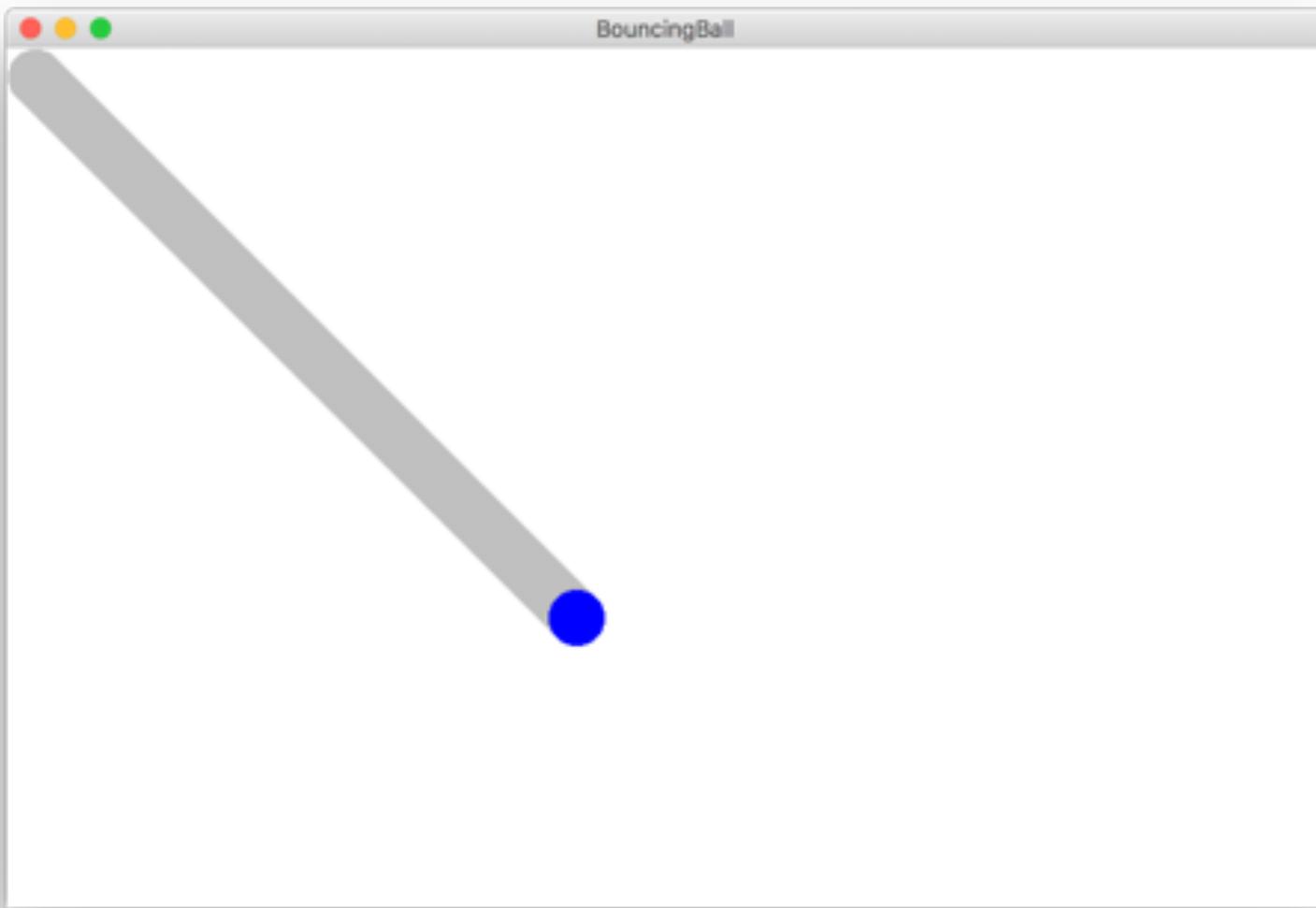


We are ready...

Bouncing Ball

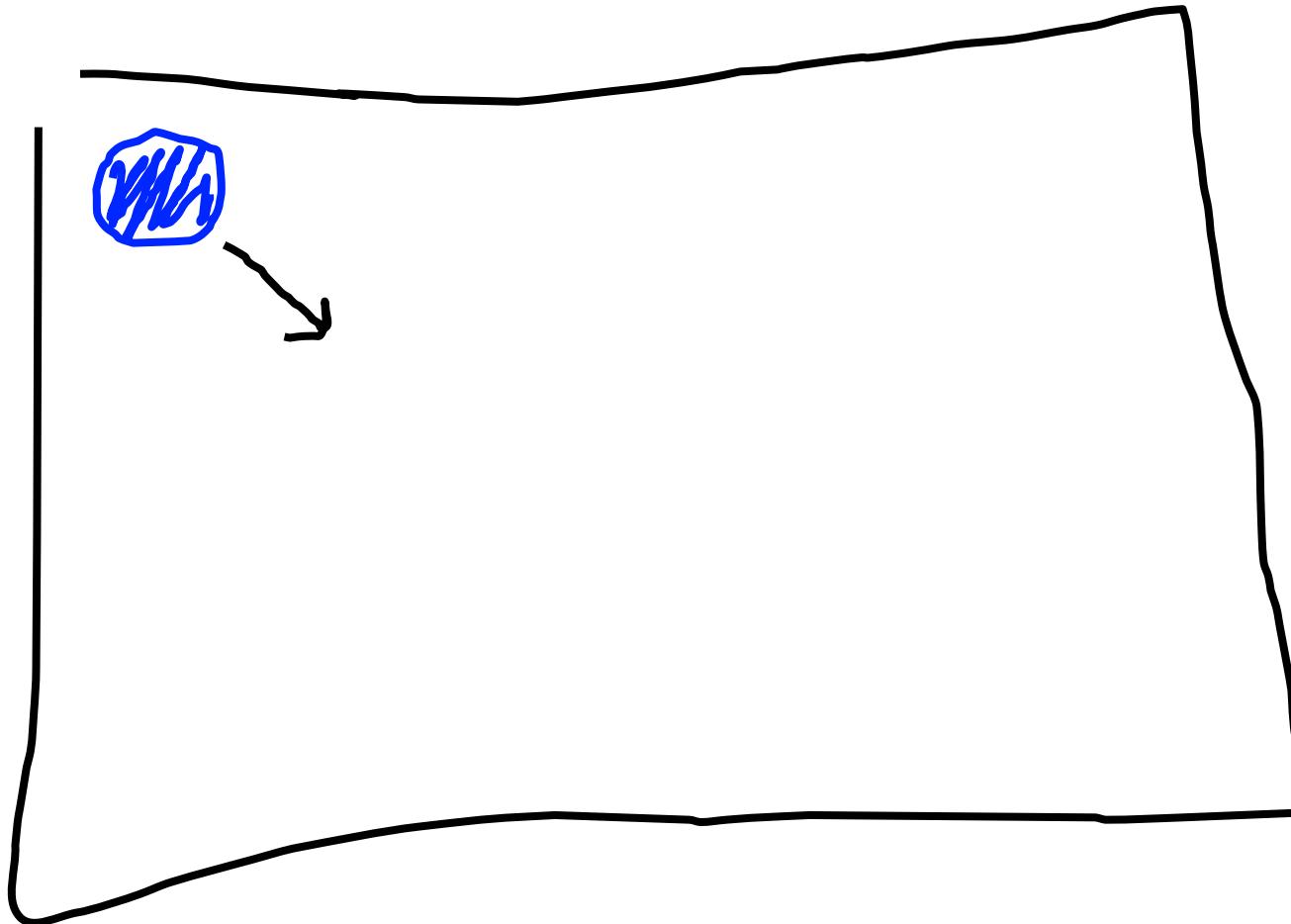


Milestone #1



Bouncing Ball

First heartbeat

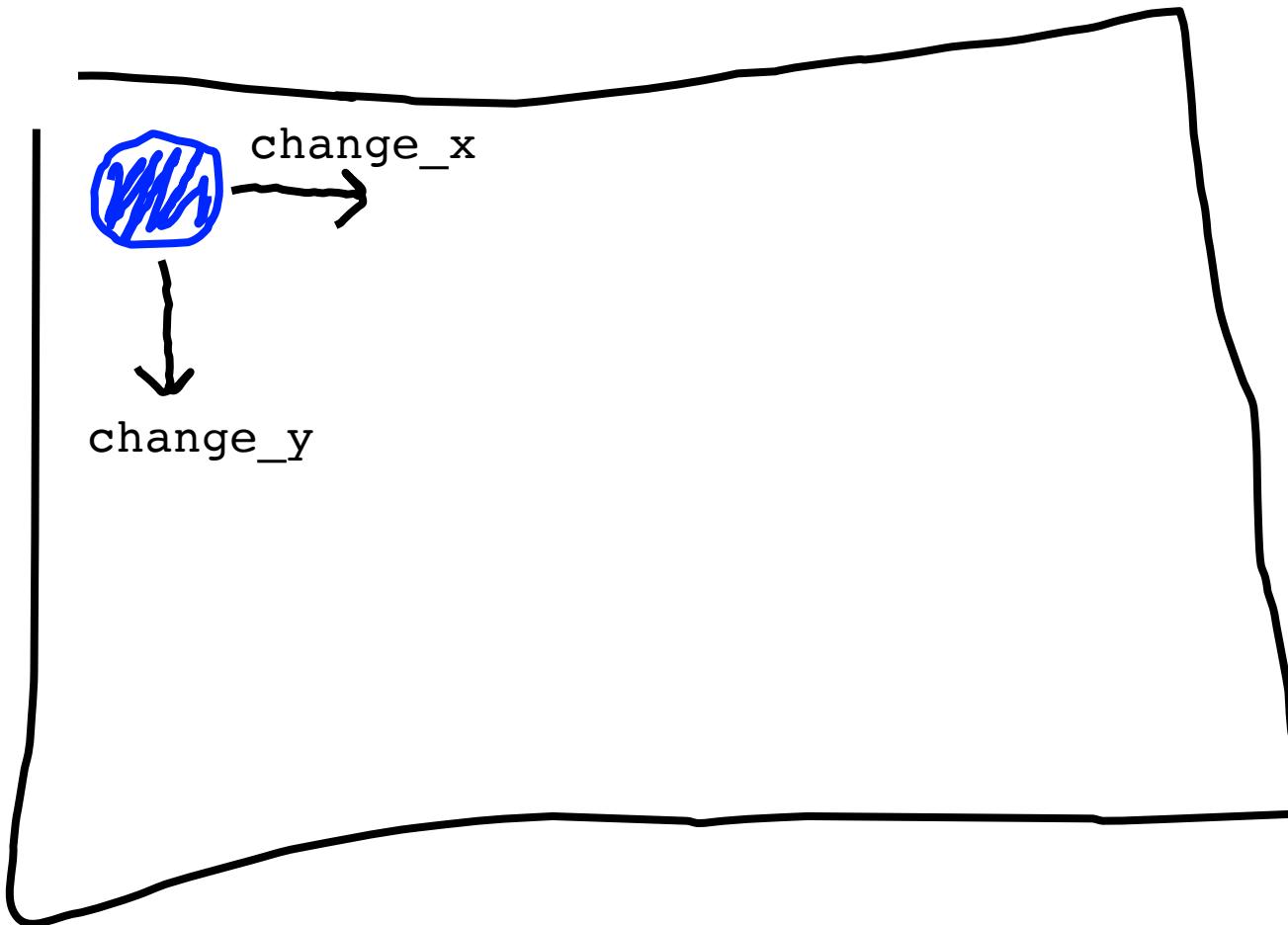


Key variable: how much the ball position
change each heartbeat?



Bouncing Ball

First heartbeat

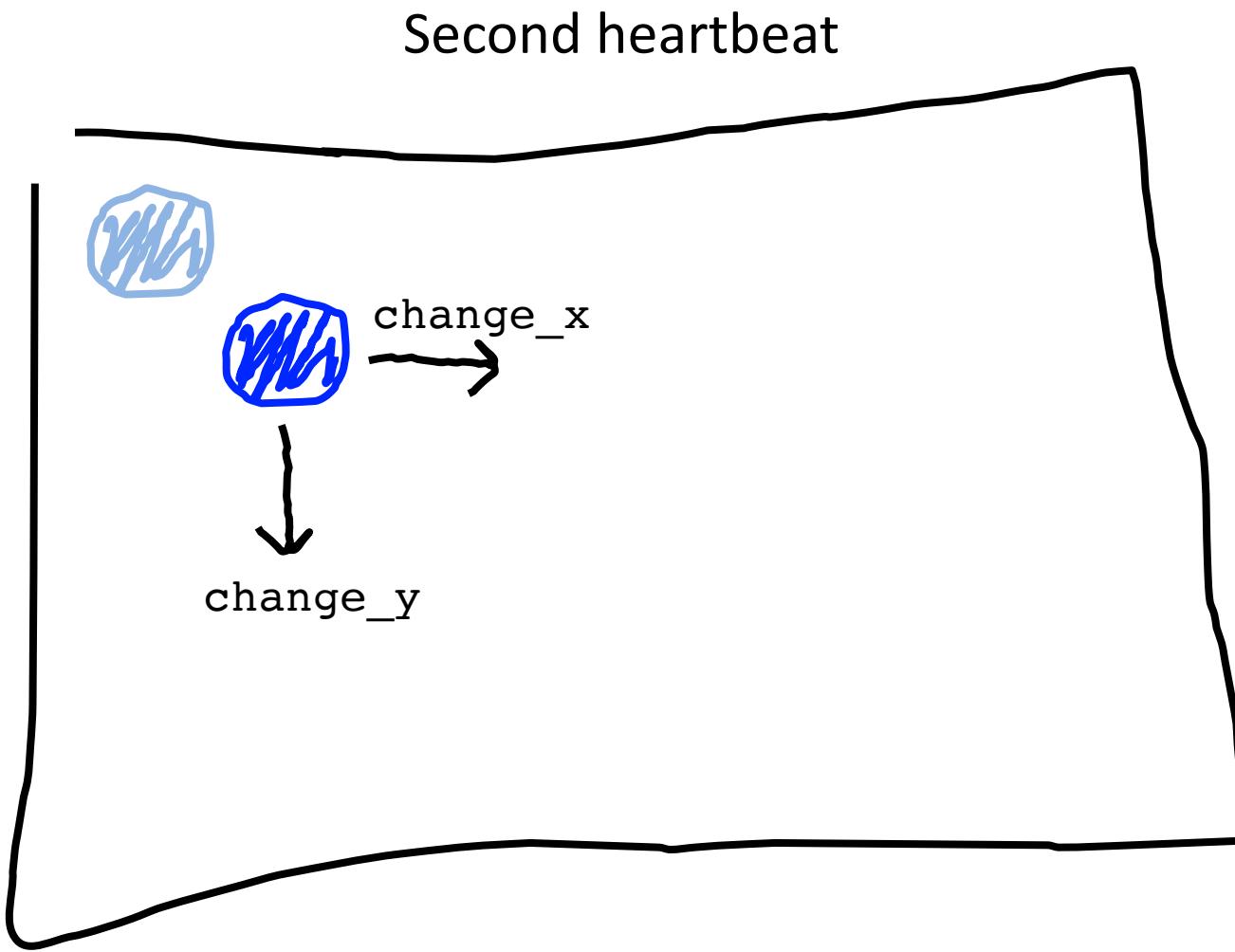


The **move** function takes in
a change in x and a change in y

Pi

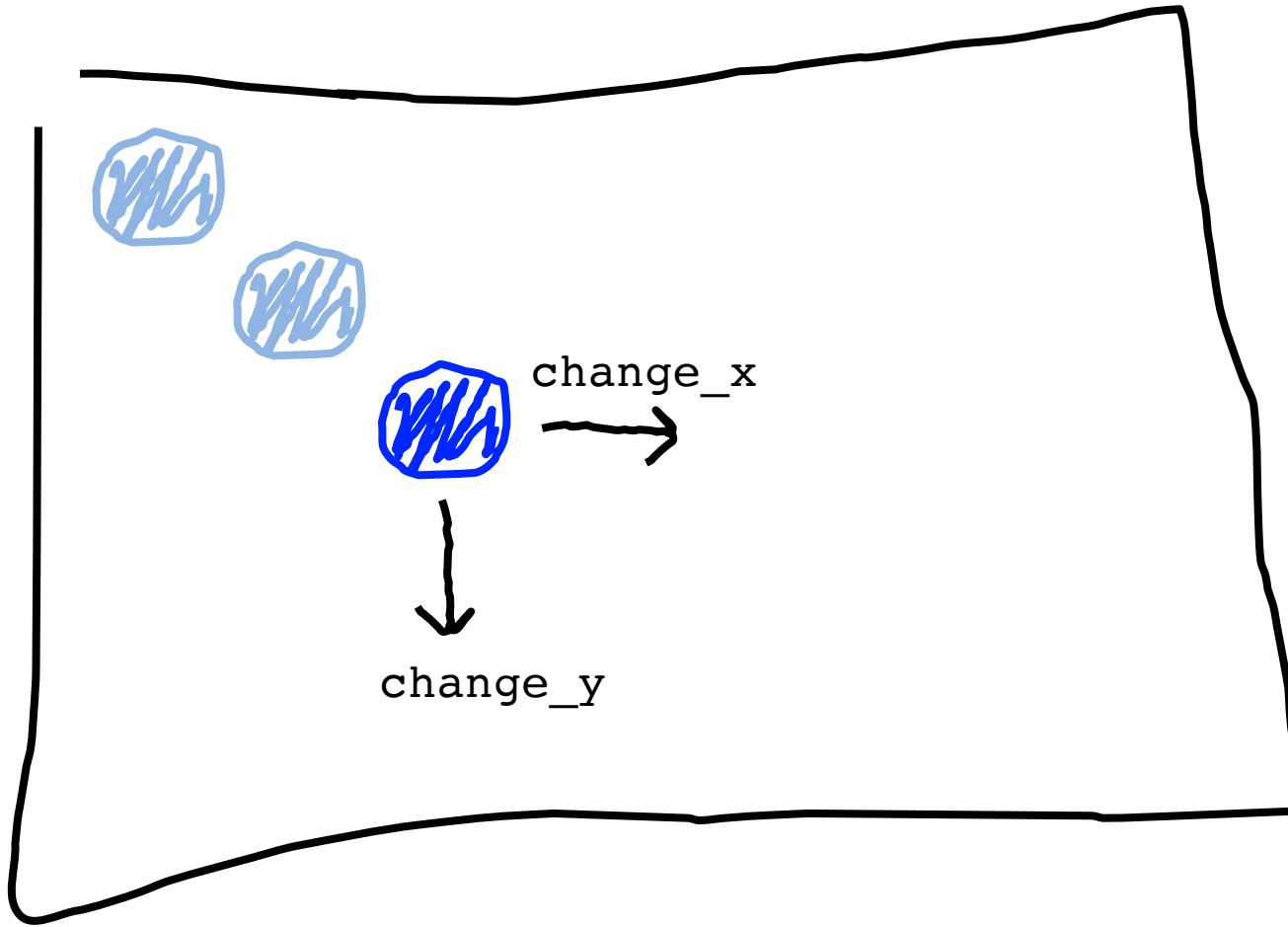


Bouncing Ball



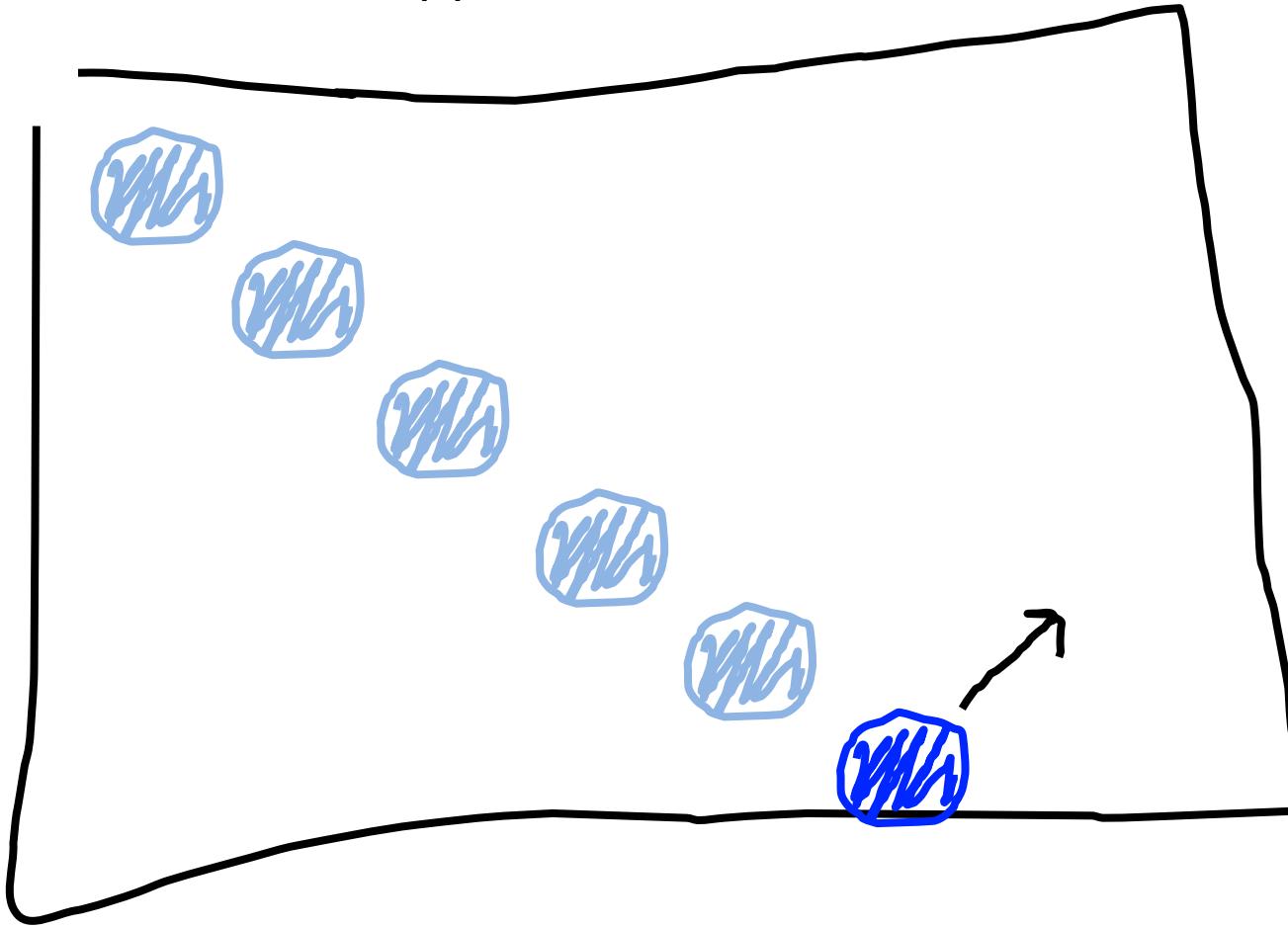
Bouncing Ball

Third heartbeat



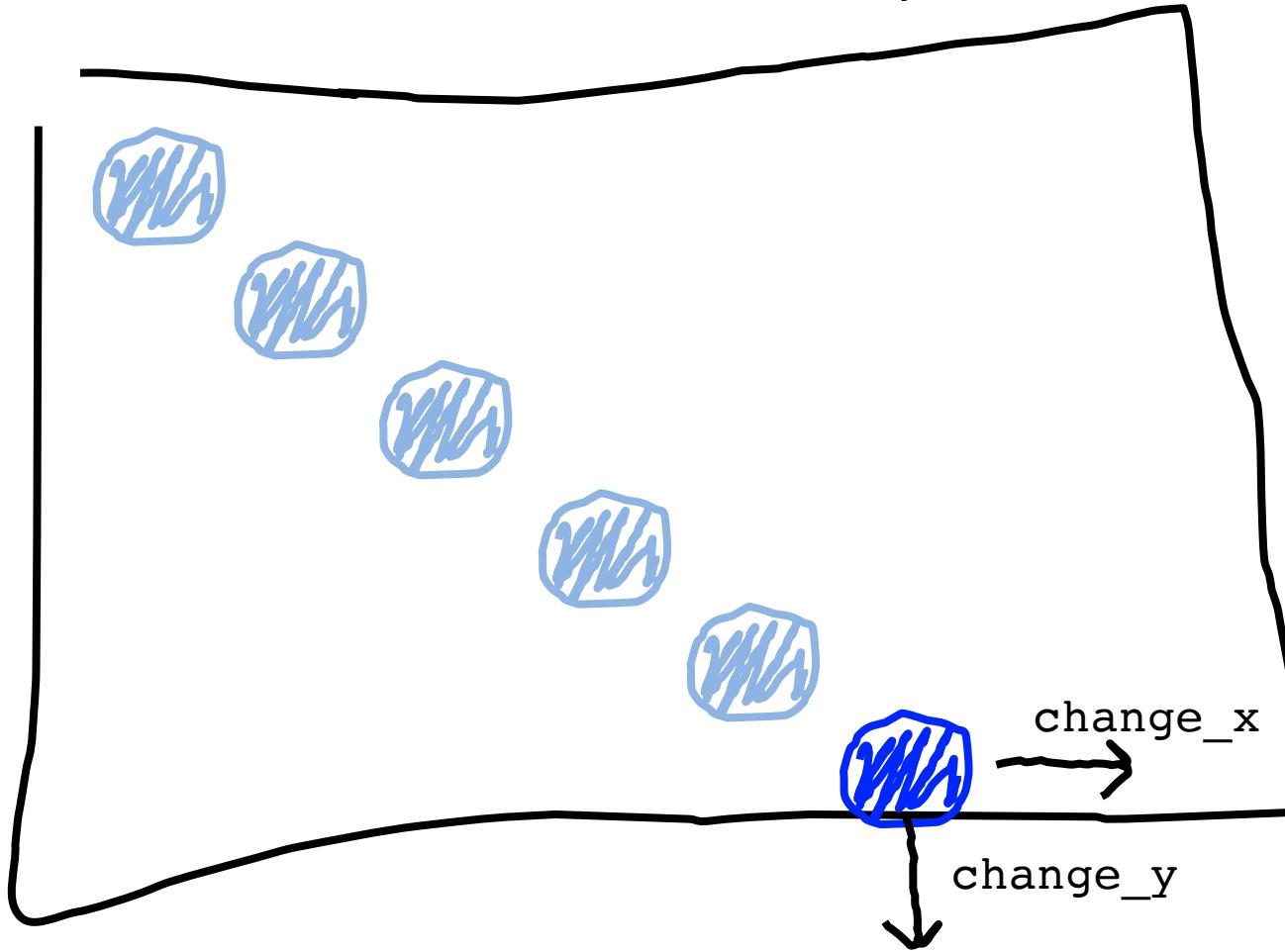
Bouncing Ball

What happens when we hit a wall?



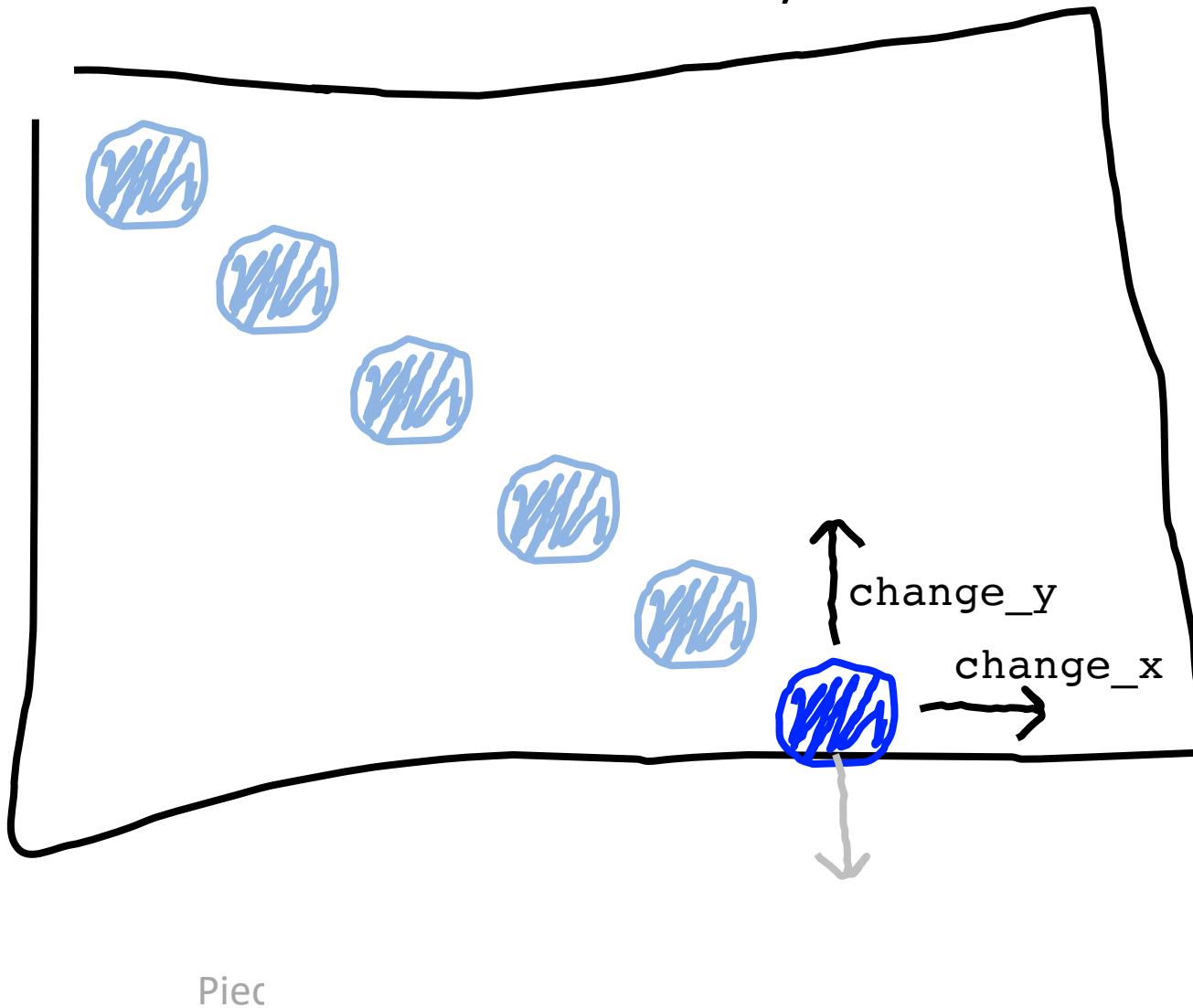
Bouncing Ball

We have this velocity



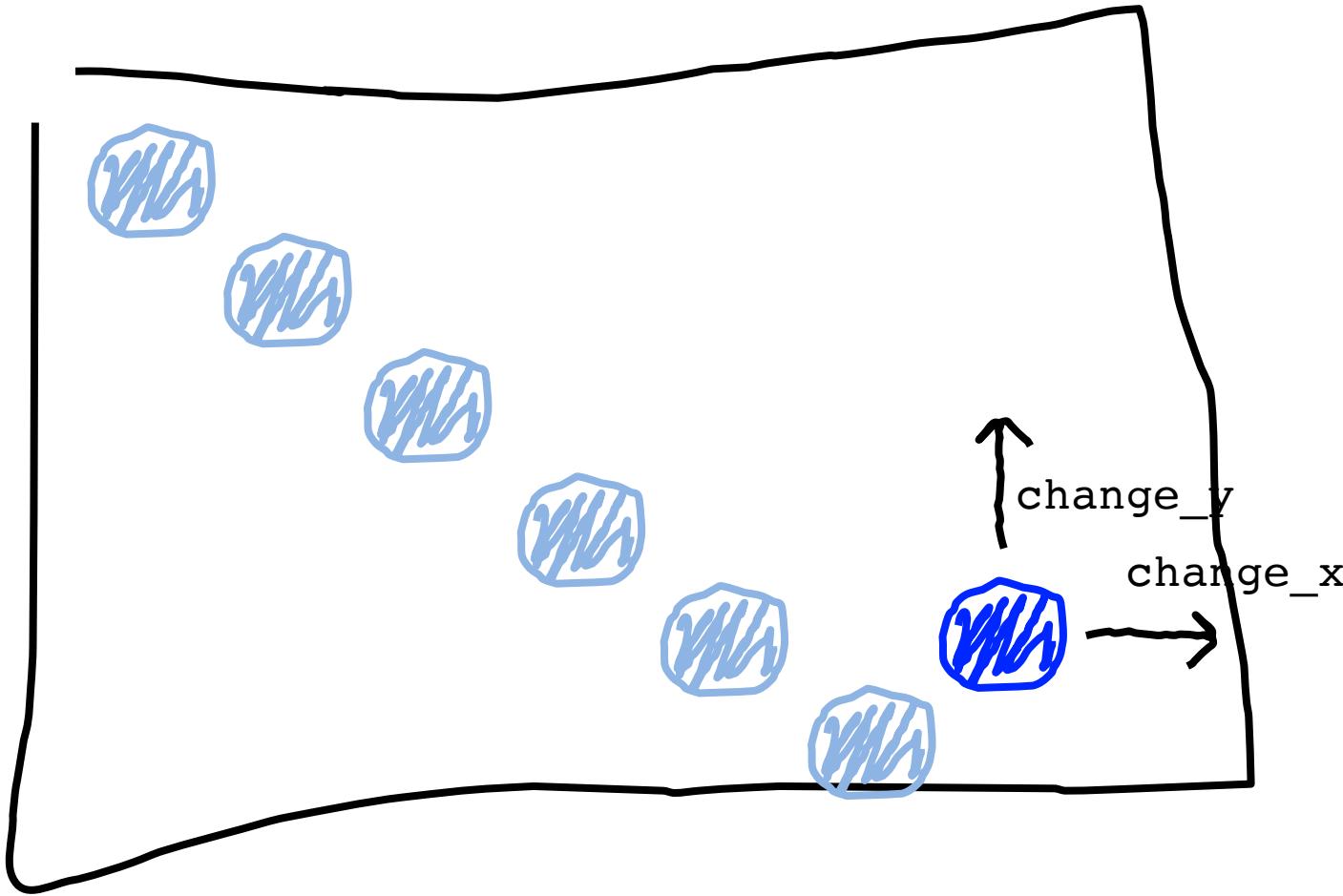
Bouncing Ball

Our new velocity



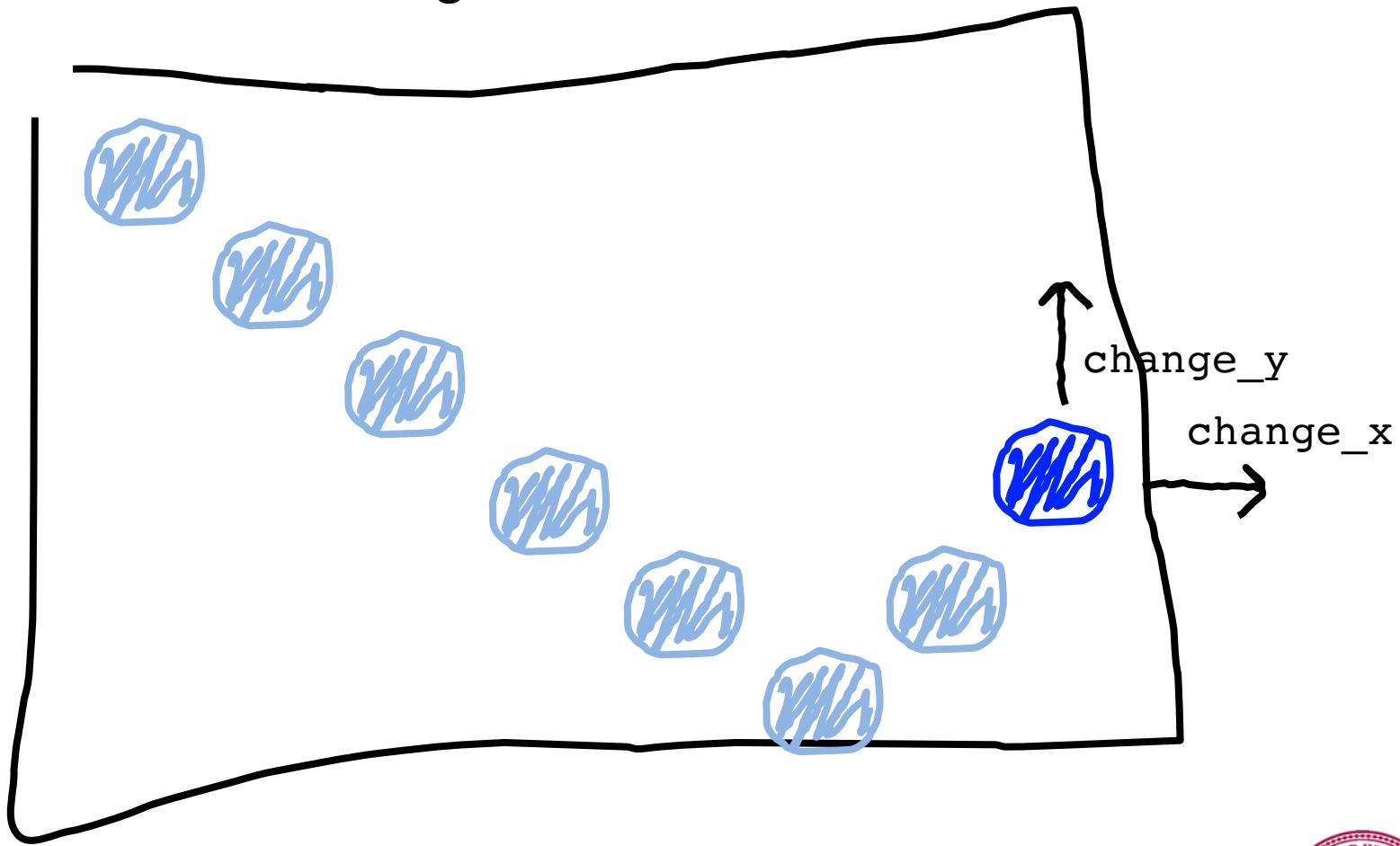
Bouncing Ball

Seventh heartbeat



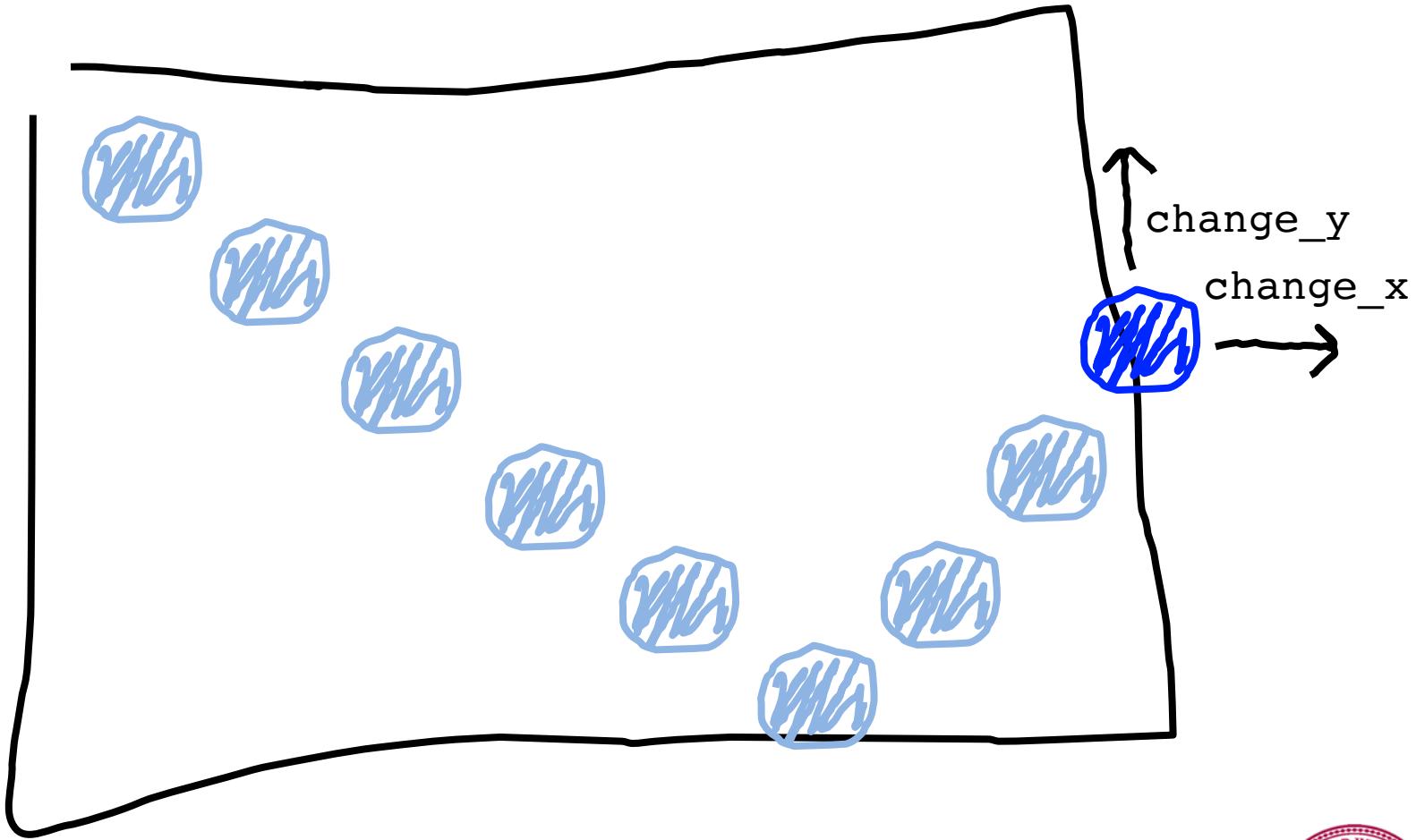
Bouncing Ball

Eighth heartbeat



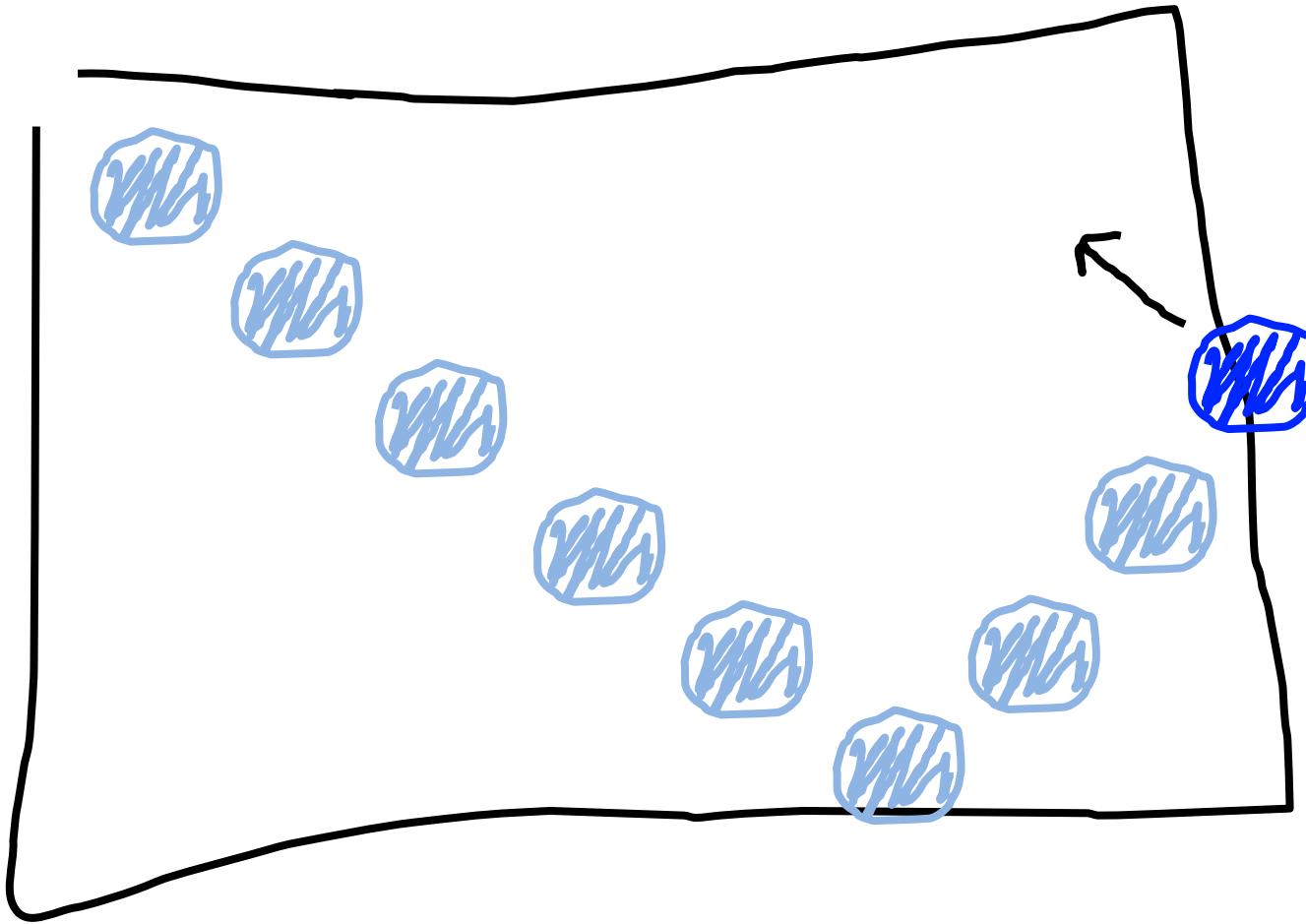
Bouncing Ball

Ninth heartbeat



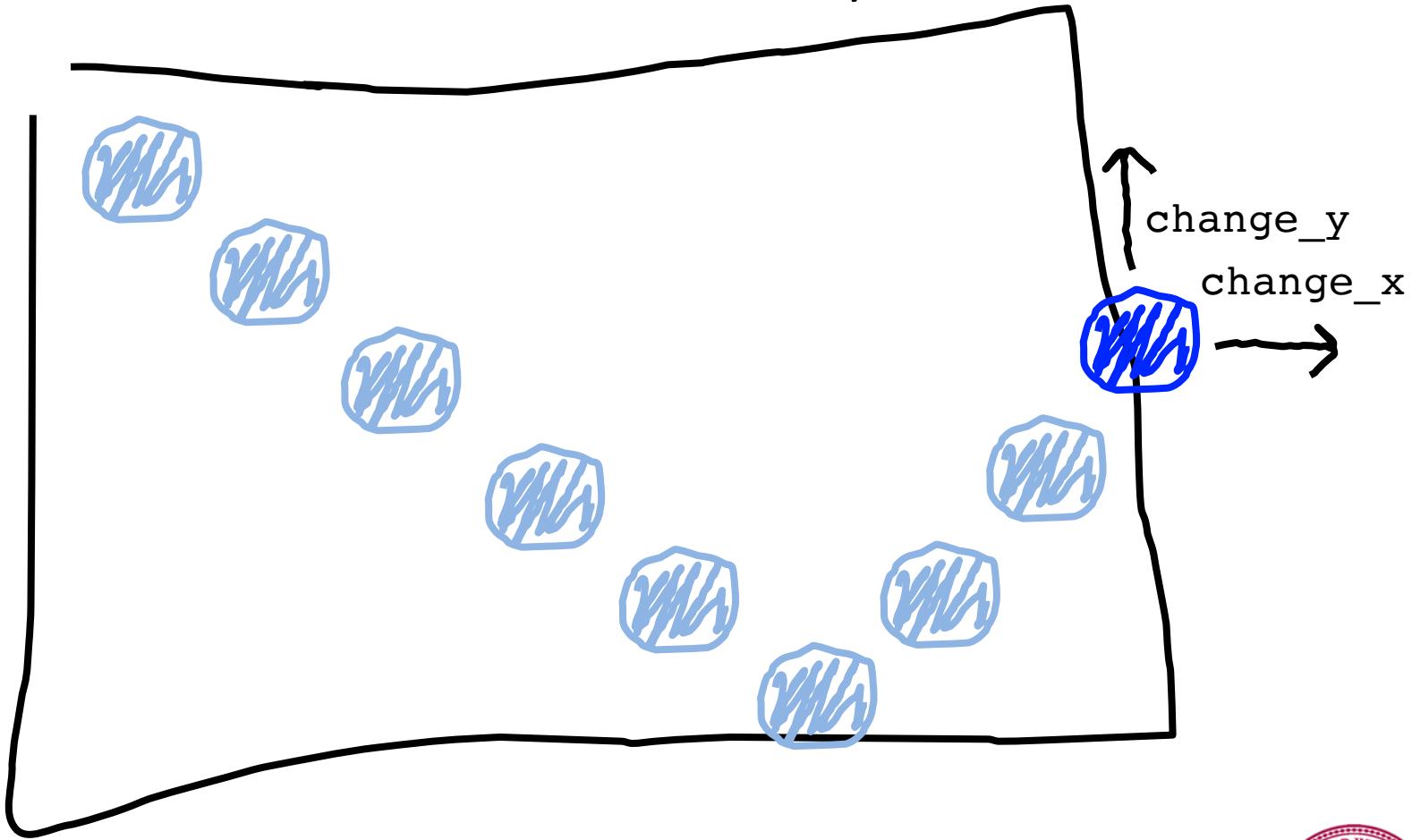
Bouncing Ball

We want this!



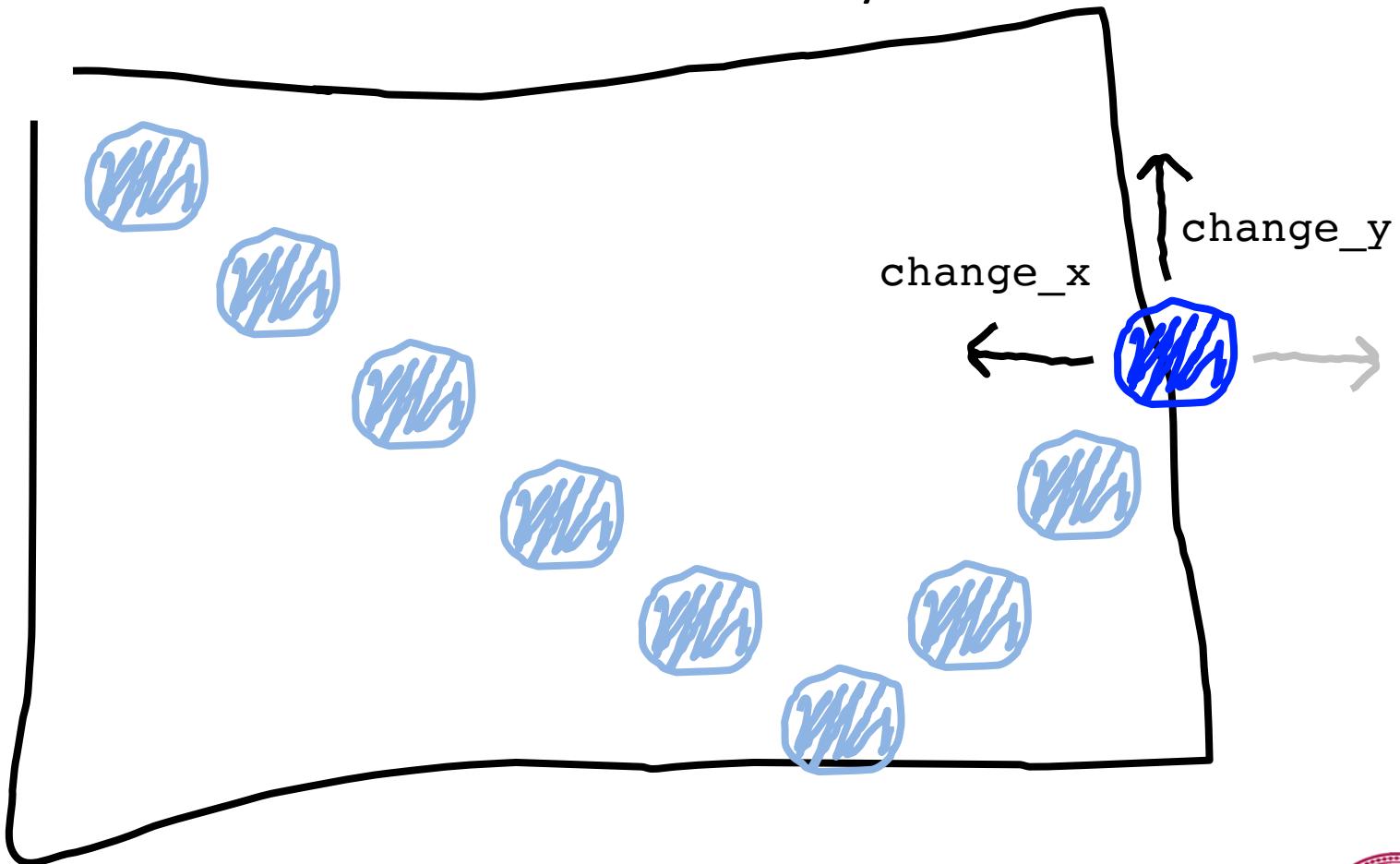
Bouncing Ball

This was our old velocity



Bouncing Ball

This is our new velocity

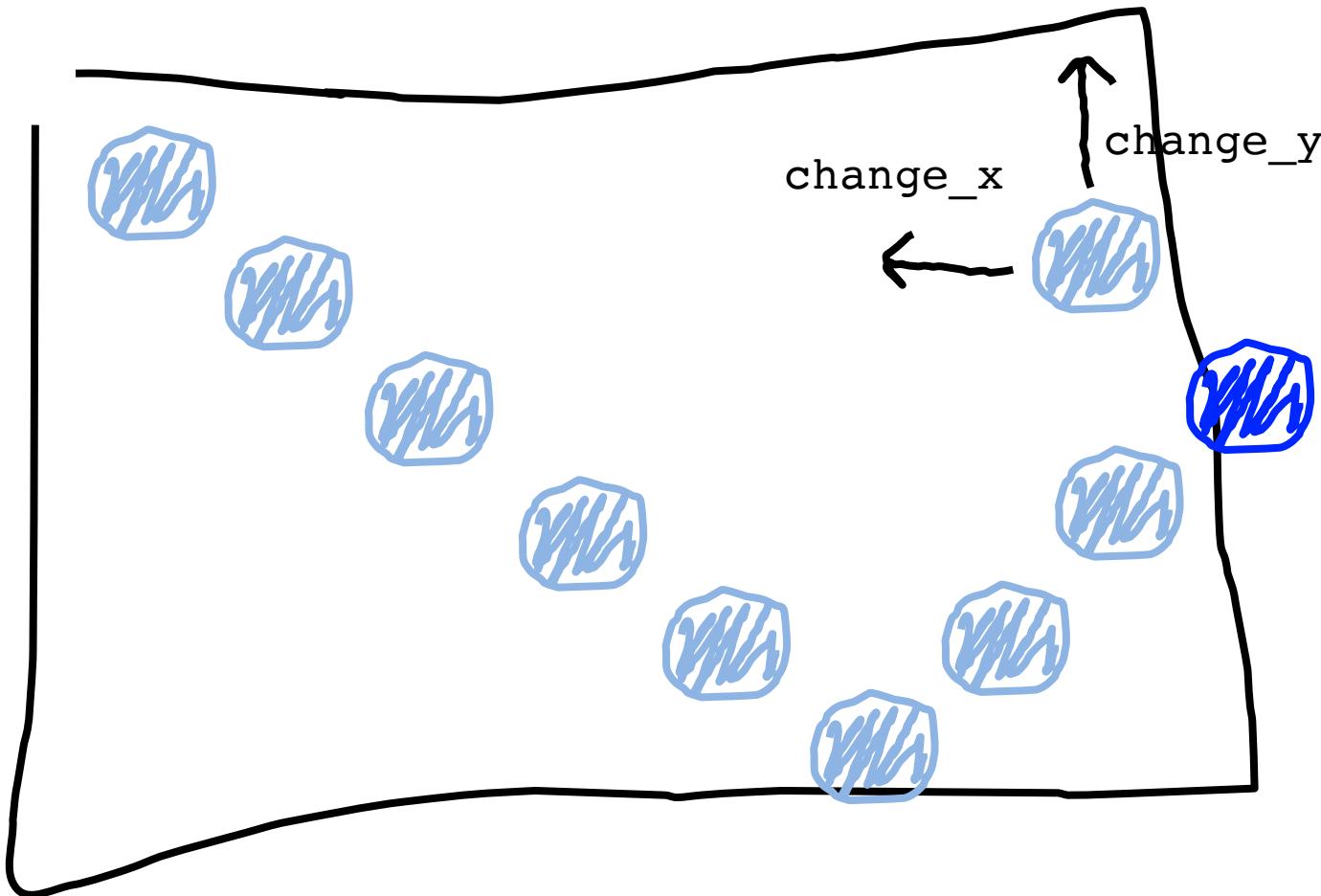


When reflecting horizontally: $change_x = -change_x$



Bouncing Ball

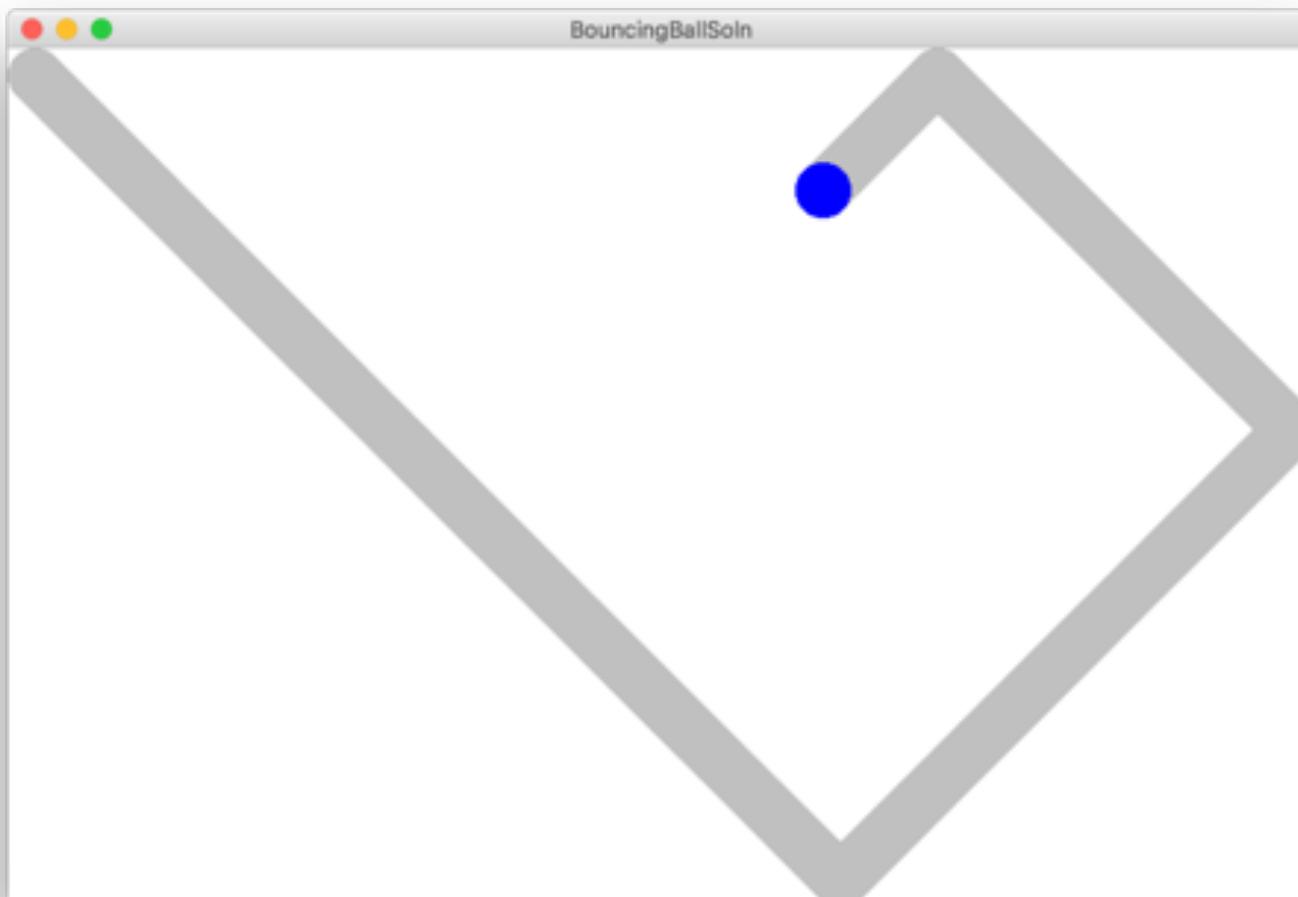
Tenth heartbeat



When reflecting horizontally: $change_x = -change_x$



Bouncing Ball



Hold up!

```
def make_ball(canvas):
```

If you get a copy when you pass a parameter. Does this copy the canvas??!!

*Python variables are stored using a reference. Which is like a **URL**. The URL gets copied when you pass the variable*



How do you share google docs?



The screenshot shows a Google Docs interface with the following details:

- Title:** Apollo 11 research
- Toolbar:** Includes File, Edit, View, Insert, Format, Tools, Table, Add-ons, Help, and a note that the last edit was 3 hours ago.
- Header:** Alexandre Sadi, Comments, Share
- Left Sidebar (Document Outline):** Lists sections including Apollo 11, Summary, The Spacecraft, Design, Command module, Service module, The People, Neil Armstrong, Buzz Aldrin, Mission Highlights, The Launch, The Landing, and Return Trip. The "Summary" section is currently selected.
- Content Area:**
 - Apollo 11:** A section titled "Apollo 11" with a sub-section "Summary". The text describes the Apollo 11 moon mission, mentioning Neil Armstrong, Buzz Aldrin, and Michael Collins landing at Tranquility Base on July 20, 1969, with the message "Tranquility Base here. The Eagle has landed."
 - The Spacecraft:** A section titled "The Spacecraft" describing the three spacecraft: Command Module Columbia, Service Module, and Lunar Module Eagle. Columbia was the only part to return to Earth.
 - Design:** A section titled "Design" mentioning the key NASA spacecraft involved: a Saturn V rocket, an Apollo CSM-107 (Command/Service Module), and an Apollo LM-5 (Lunar Module, AKA "The Eagle").
 - Command module:** A section titled "Command module" describing the CSM as one of two spacecraft used for the United States Apollo program, built by North American Aviation, and its use in suborbital and low Earth orbit test missions with the Saturn IB launch vehicle, and three times by itself and nine times with the Lunar Module as part of the Apollo spacecraft assembly on the larger Saturn V launch vehicle.
 - Service module:** A section titled "Service module" mentioning its role in providing oxygen, water, and electric power for the command module, and housing the engine injection system.

<https://docs.google.com/document/d/1eBtnEii3KHe fFS-kSAOpXqeSXpbfTTMlmOgj6I9dvk/>



```
def main():
```

```
    canvas = Canvas(...)
```

```
    make_ball(canvas)
```

```
def make_ball(canvas):
```

```
    canvas.create_rectangle( ... , fill='blue')
```

stack

heap

```
main
```



```
def main():
```

```
    canvas = Canvas(...)
```

```
    make_ball(canvas)
```

```
def make_ball(canvas):
```

```
    canvas.create_rectangle( ... , fill='blue')
```

stack

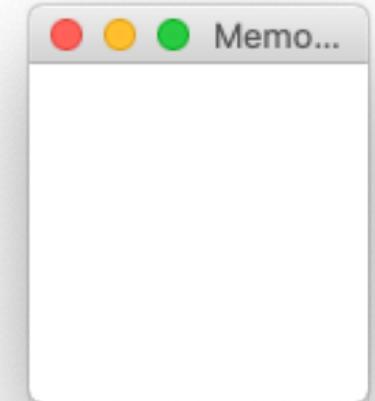
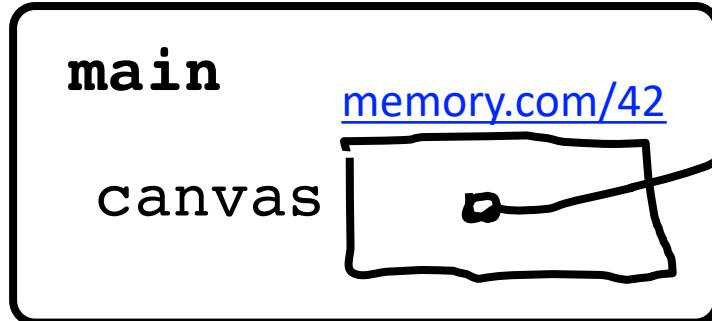
main

<memory.com/42>

canvas

heap

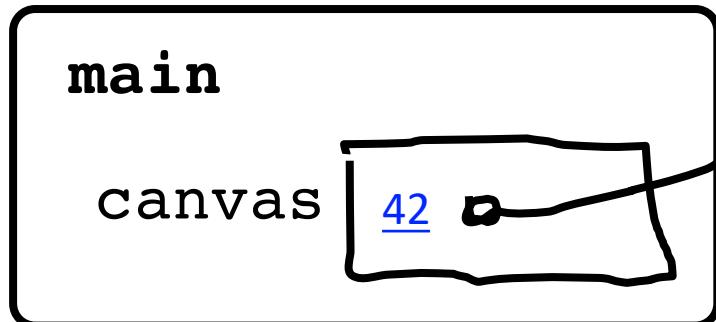
<memory.com/42>



```
def main():
    canvas = Canvas(...)
    make_ball(canvas)
```

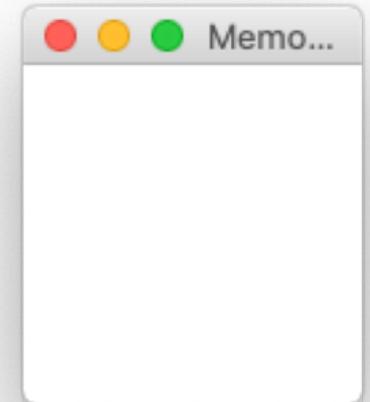
```
def make_ball(canvas):
    canvas.create_oval( ... , fill='blue')
```

stack



heap

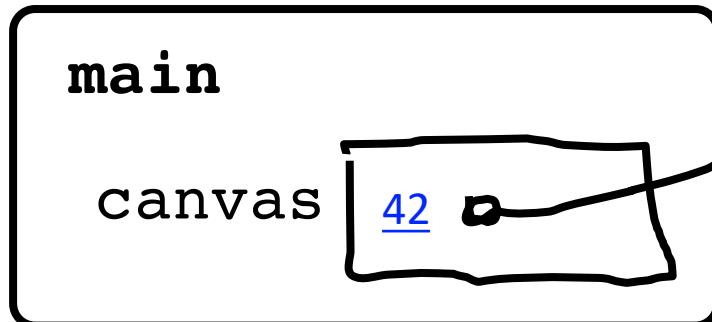
memory.com/42



```
def main():
    canvas = Canvas(...)
    make_ball(canvas)
```

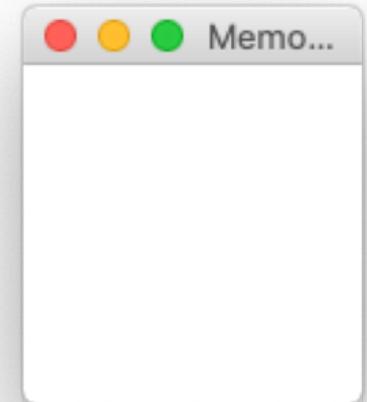
```
def make_ball(canvas):
    canvas.create_oval( ... , fill='blue')
```

stack



heap

memory.com/42



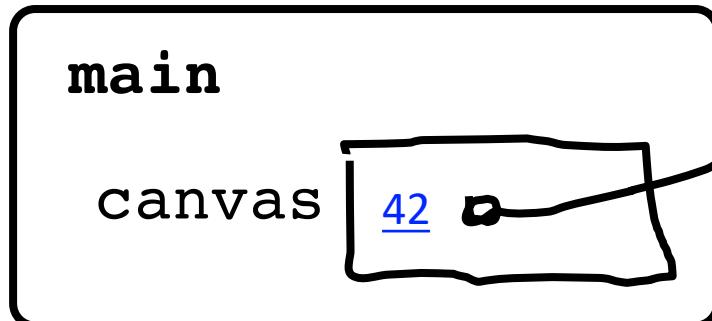
make_ball



```
def main():
    canvas = Canvas(...)
    make_ball(canvas)
```

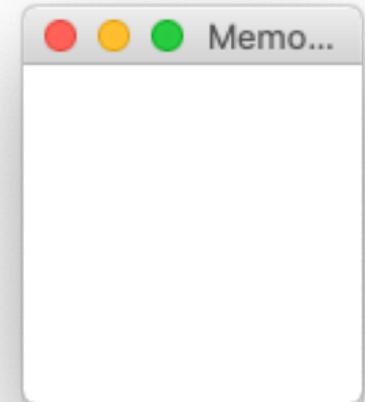
```
def make_ball(canvas):
    canvas.create_oval( ... , fill='blue')
```

stack

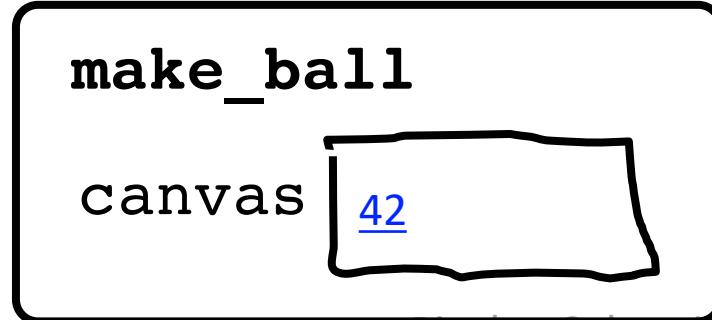


heap

memory.com/42



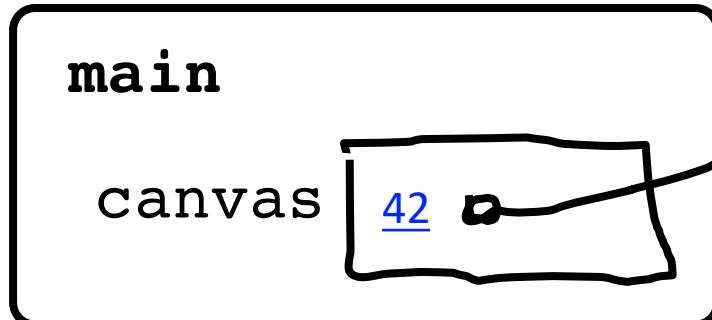
make_ball



```
def main():
    canvas = Canvas(...)
    make_ball(canvas)
```

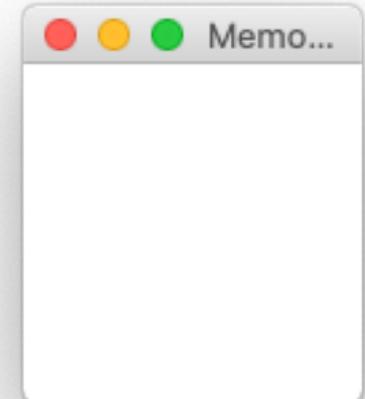
```
def make_ball(canvas):
    canvas.create_oval( ... , fill='blue')
```

stack

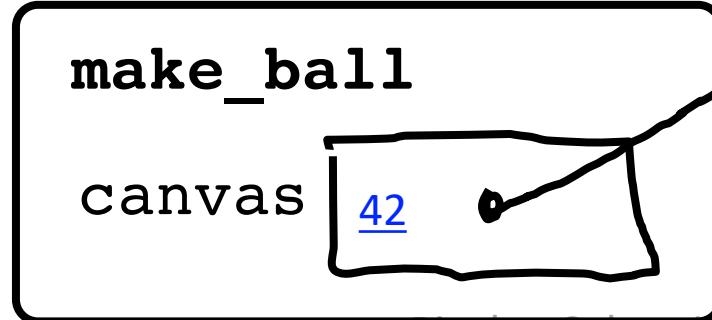


heap

memory.com/42



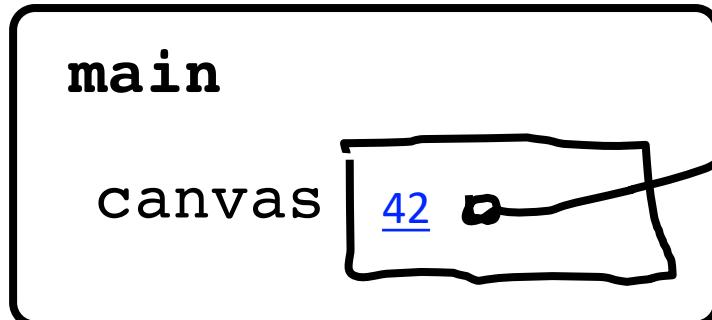
make_ball



```
def main():
    canvas = Canvas(...)
    make_ball(canvas)
```

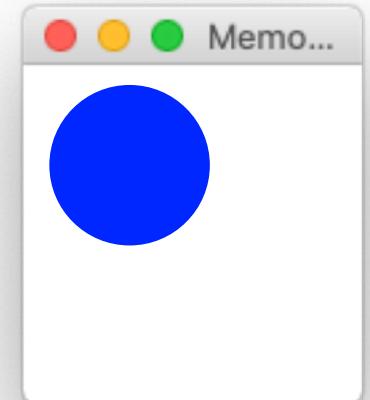
```
def make_ball(canvas):
    canvas.create_oval( ... , fill='blue')
```

stack

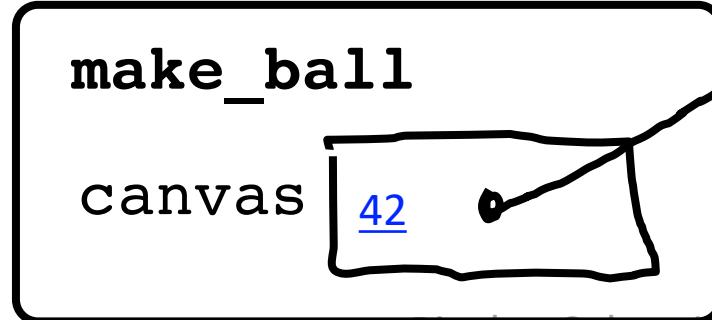


heap

memory.com/42



make_ball



```
def main():
```

```
    canvas = Canvas(...)
```

```
    make_ball(canvas)
```

```
def make_ball(canvas):
```

```
    canvas.create_oval( ... , fill='blue')
```



stack

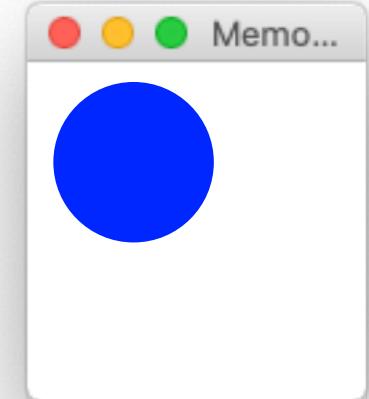
main

canvas

42

heap

memory.com/42



make_ball

canvas

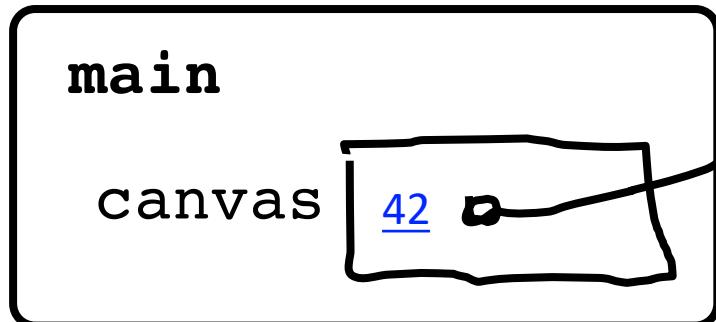
42



```
def main():
    canvas = Canvas(...)
    make_ball(canvas)
```

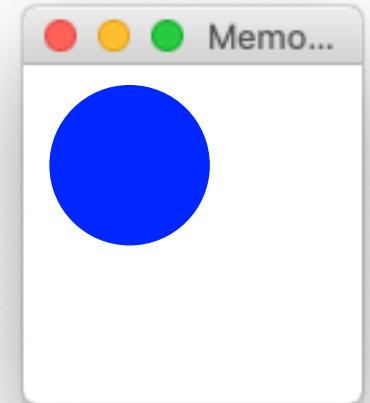
```
def make_ball(canvas):
    canvas.create_oval( ... , fill='blue')
```

stack



heap

memory.com/42





When passing variables,
some act just like you are
passing a URL.

That allows functions to
modify the variable

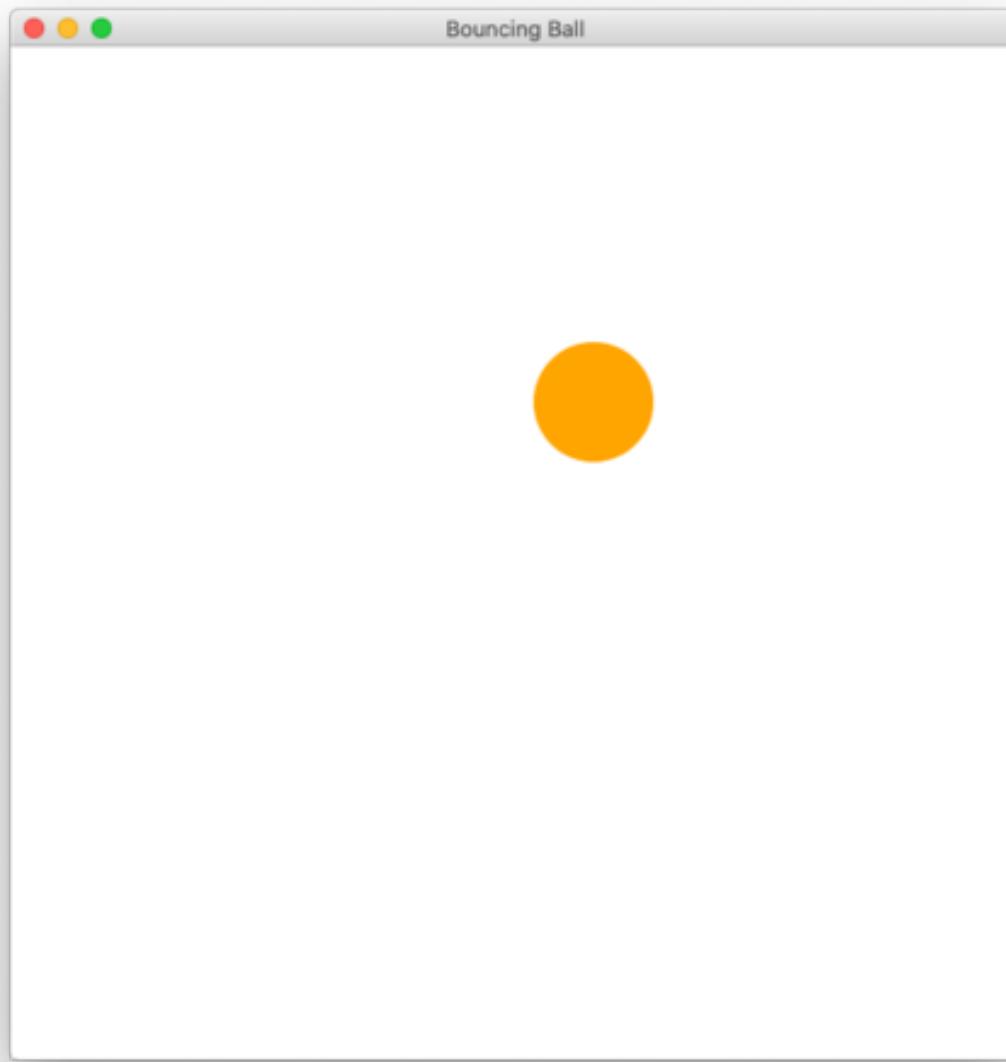


Learning Goals

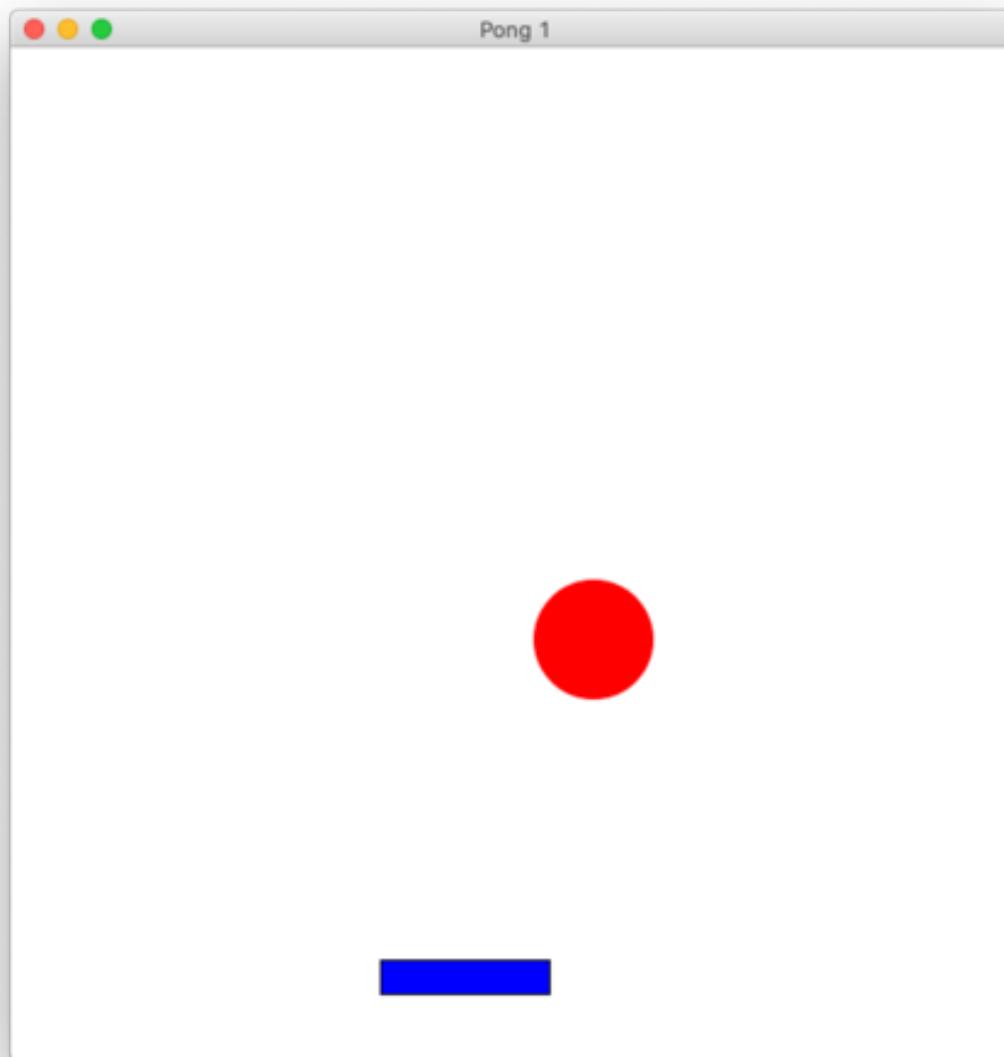
1. Write animated programs



ball_colors.py



pong.py



Special Graphics Functions

```
# get the x location of the mouse
mouse_x = canvas.get_mouse_x()
```

```
# move shape to some new coordinates
canvas.moveto(shape, new_x, new_y)
```

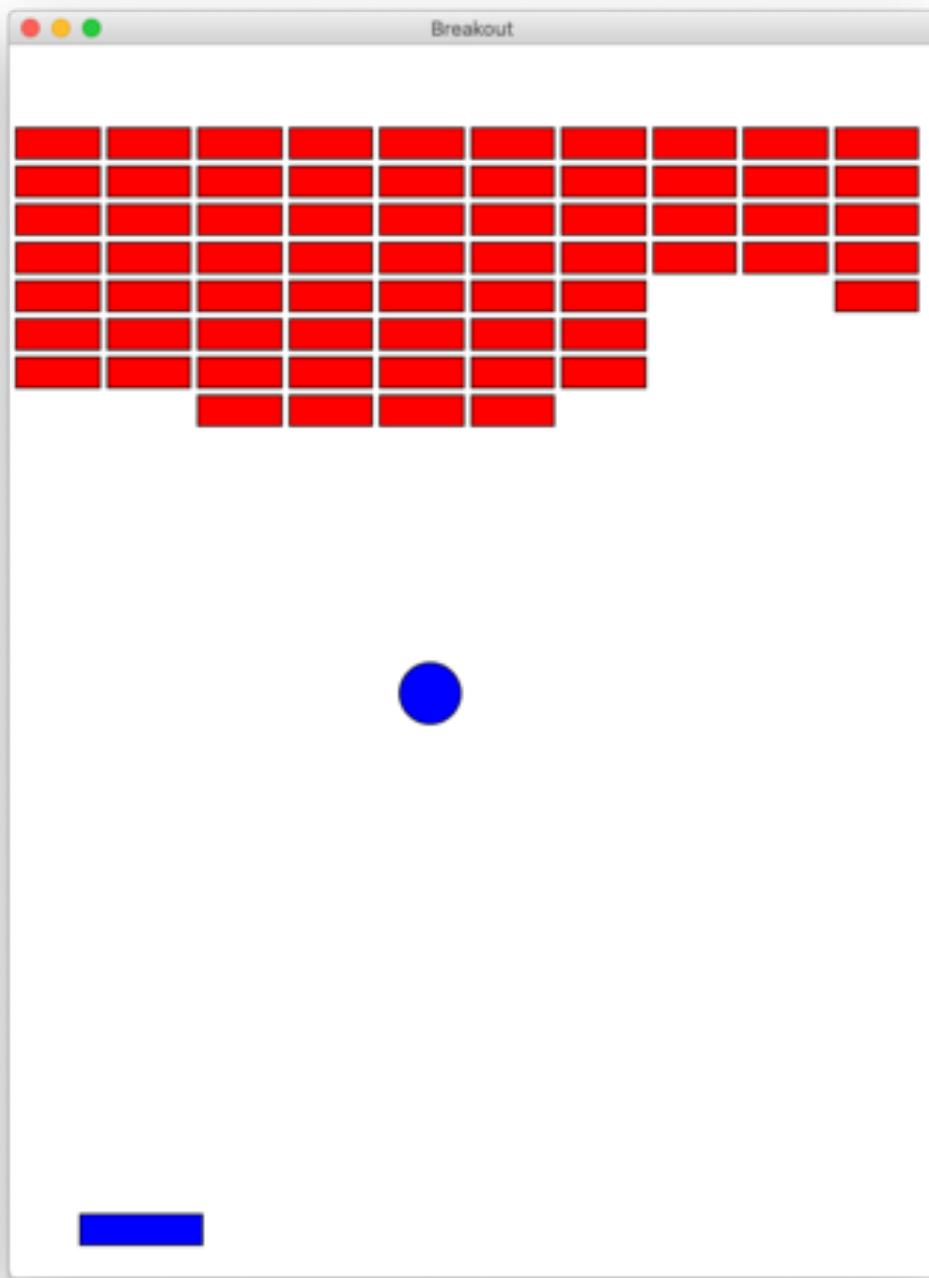
```
# move shape by a given change_x and change_y
canvas.move(shape, change_x, change_y)
```

```
# get the coordinates of a shape
top_y = canvas.get_top_y(shape)
left_x = canvas.get_left_x(shape)
coord_list = canvas.coords(shape)
```

```
# return a list of elements in a rectangle area
results = canvas.find_overlapping(x1, y1, x2, y2)
```

```
# you can change a shape's color too
canvas.set_fill_color(shape, new_color)
canvas.set_outline_color(shape, new_color)
```





Piech + Sahami, CS106A, Stanford University

