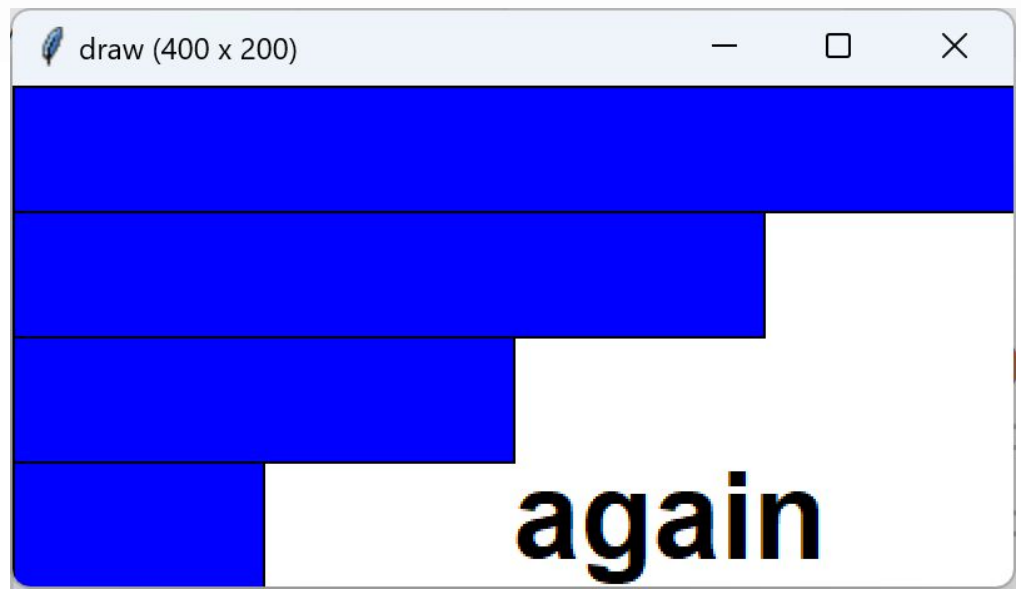
Week 5: Agenda

- Mentor meetings:
  - Officially starting this week
  - Do not miss a meeting
- HW4: due **Wednesday Feb 5**
- Regret Period for HW3: Deadline **Tonight 10pm**
  - **e-mail and,**
  - **in-person meeting tomorrow**

Code Tracing (Graphics)

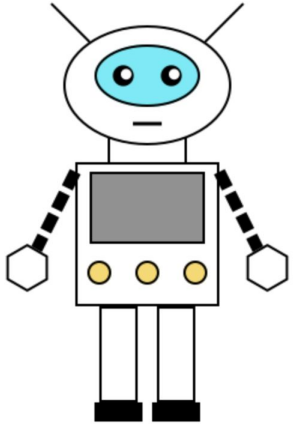
```
def drawCT1(app, m, n):
    d = app.width
    x = 0
    y = 0
    for i in range(n):
        if i%2 == 0:
            color = 'black'
        else:
            color = 'white'
        drawRect(x, y, d, d, fill=color, border='black')
        d -= m
        x += m//2
        y += m//2
    drawLabel("112 Rocks", app.width//2, app.height//2,
                align='center', fill='black')

def redrawAll(app):
    drawCT1(app, 100, 4)

runApp()
```

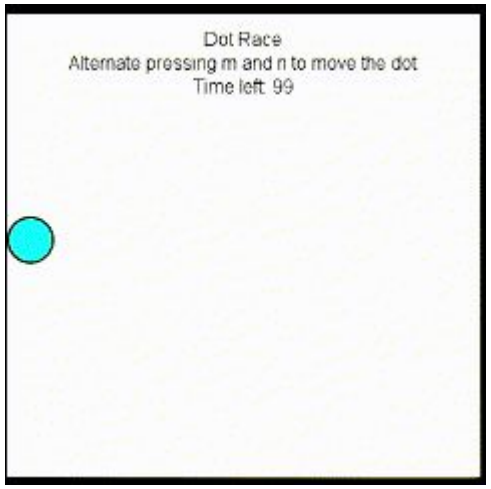
```
from cmu_graphics import *

def redrawAll(app):
    h = 50 # Height of first rectangle

    for i in range(4):
        x0 =     # _____________________ (A)
        y0 =     #_____________________ (B)
        w  =     #_____________________ (C)
        drawRect(x0, y0, w, h, fill="blue")
    drawLabel("again", 200, 180, align='bottom-left',size=h, bold=True)

runApp(400, 200)
```



Graphics and Animation Tasks

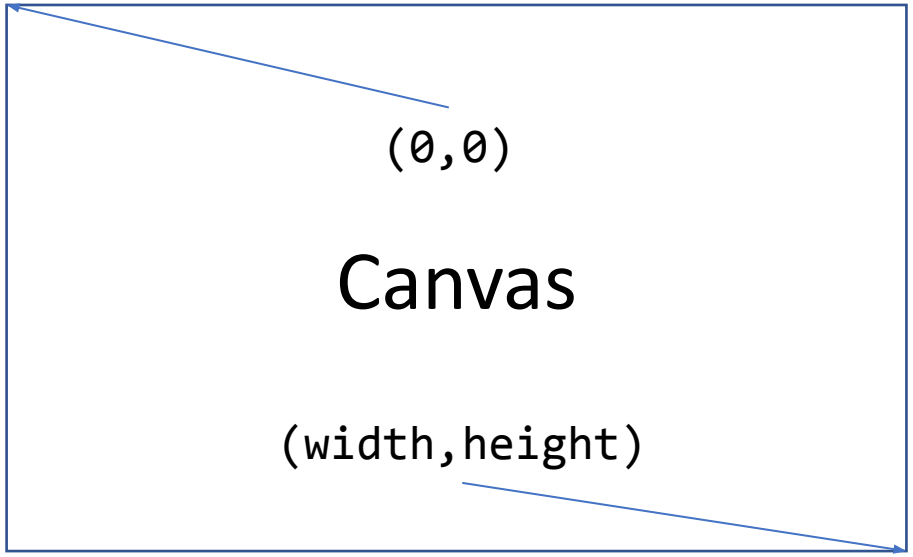
- Drawing: Create or replicate visuals by combining shapes, colors, and attributes.
- Animating
  - Capture and respond to mouse and keyboard events.
  - Use timers to update drawings over time, creating dynamic effects.





cmu_graphics

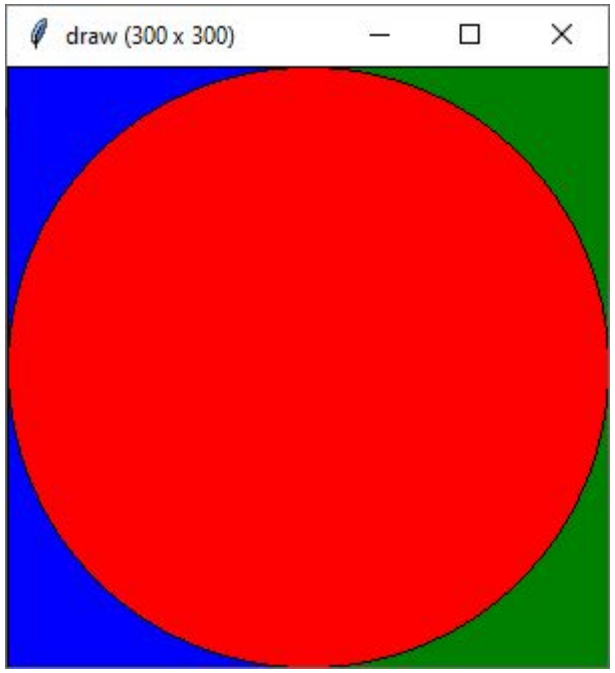
```
1  from cmu_graphics import *
2
3  def redrawAll(app):
4      # < your code >
5
6  runApp()
```

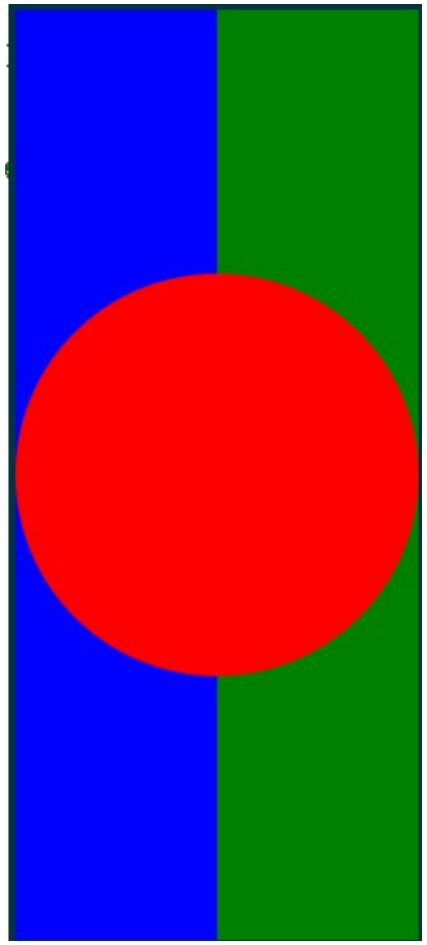


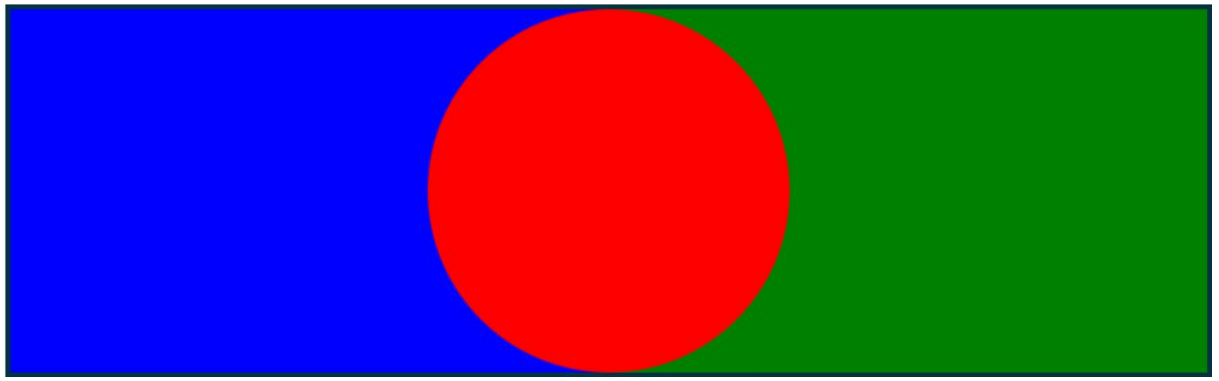
Free Response (again)

The figure should span the entire window

The circle must be centered and touch the border of the window







How to center shapes?

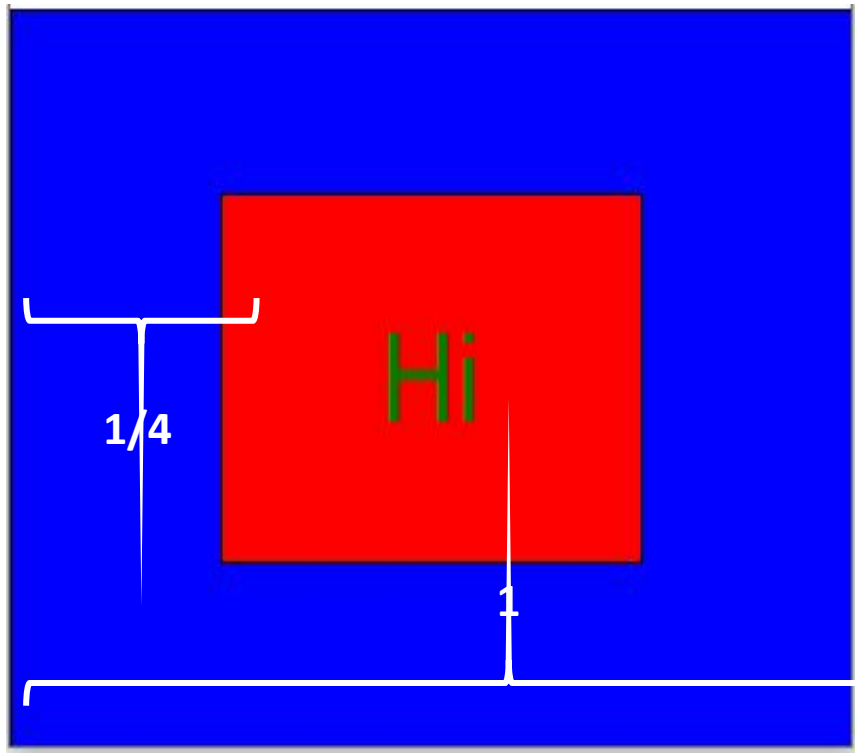
- Option 1: Add margin to top/left, subtract margin from bottom/right

```
from cmu_graphics import *

def redrawAll(app):
    marginH = 300
    marginV = 250
    rectW = 200
    rectH = 100
    drawRect( marginH,   marginV, rectW, rectH, fill="orange")

runApp(800, 600)
```

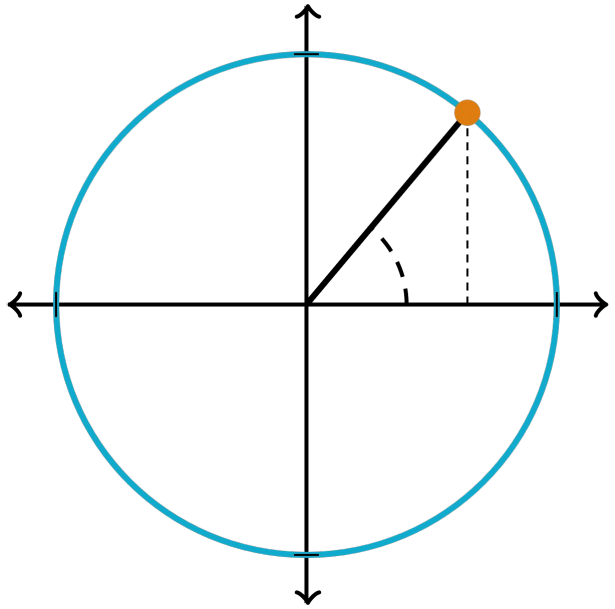
- Option 2: Add/subtract width/height from the location of the shape

```
from cmu_graphics import *

def redrawAll(app):
    rectW = 200
    rectH = 100
    drawRect( app.width//2 - rectW//2,   app.height//2 - rectH//2, rectW, rectH, fill="orange")

runApp(800, 600)
```

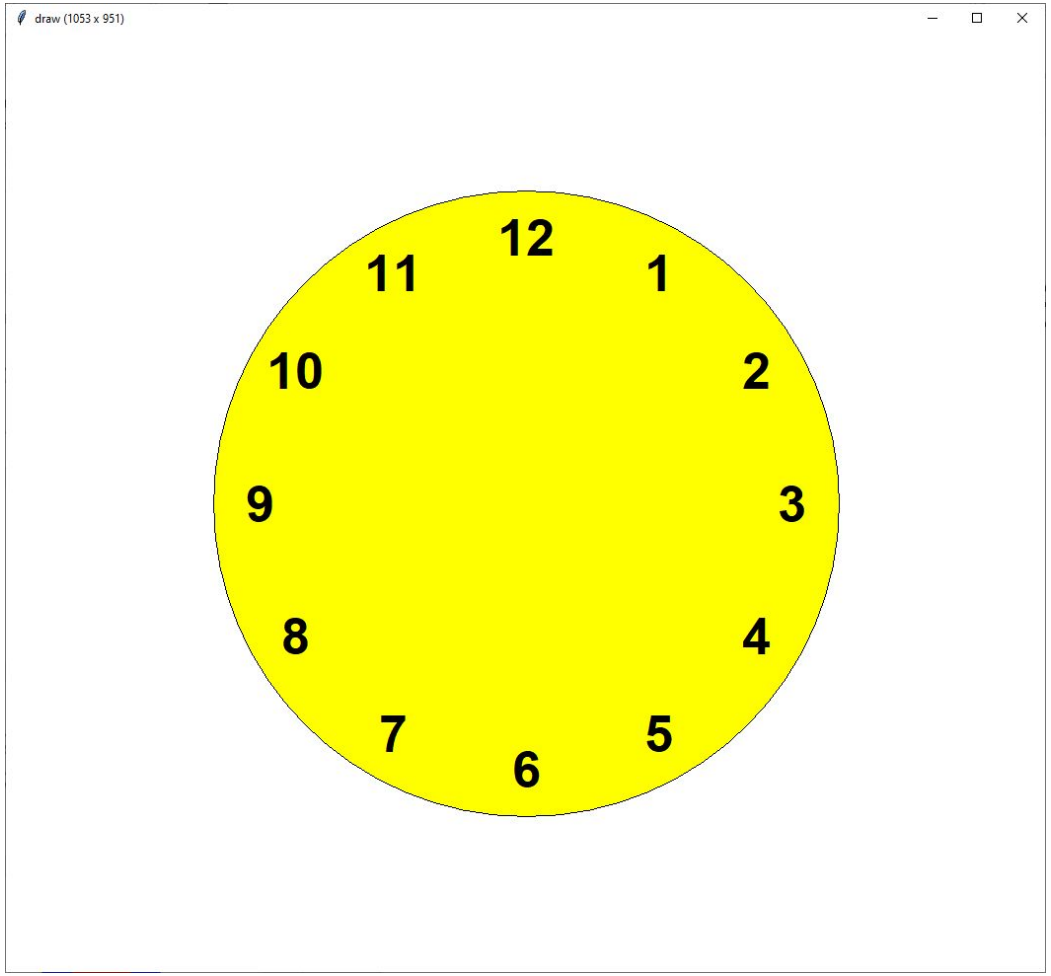
Free Response



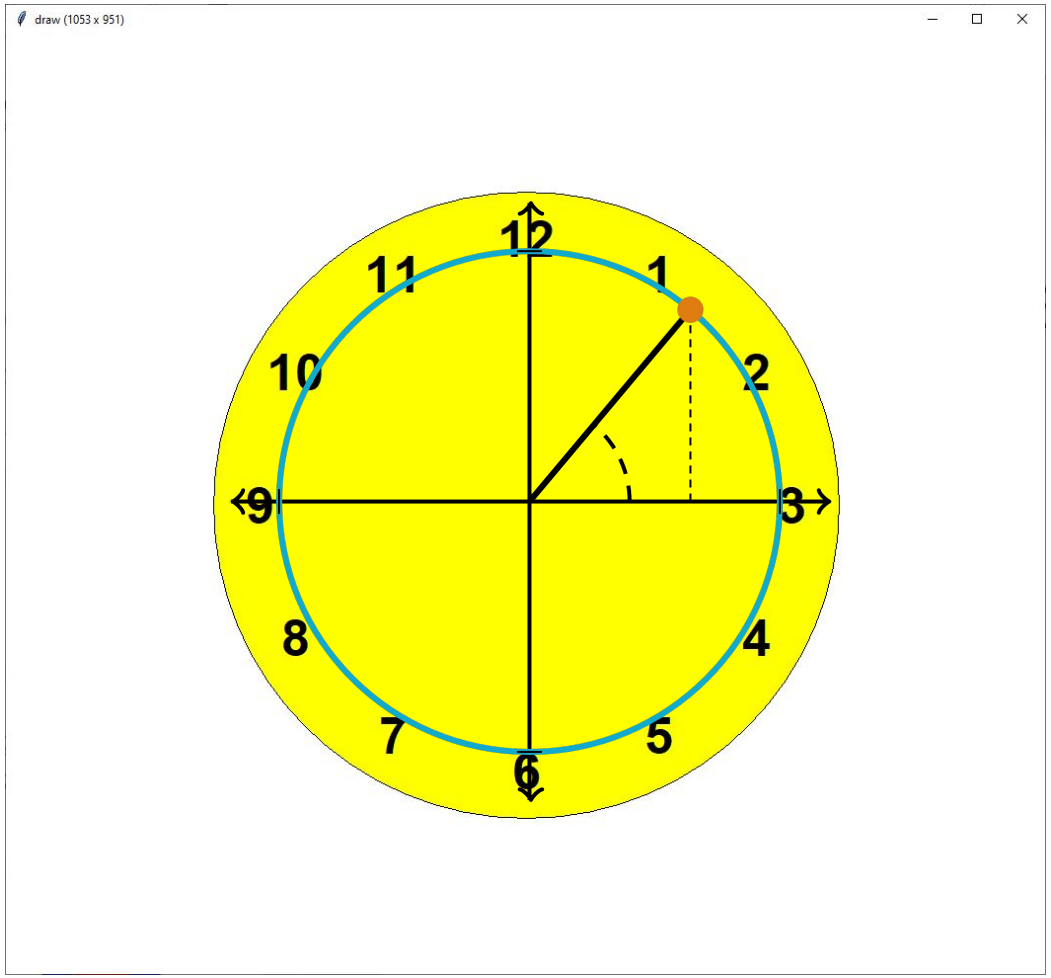
Free Response



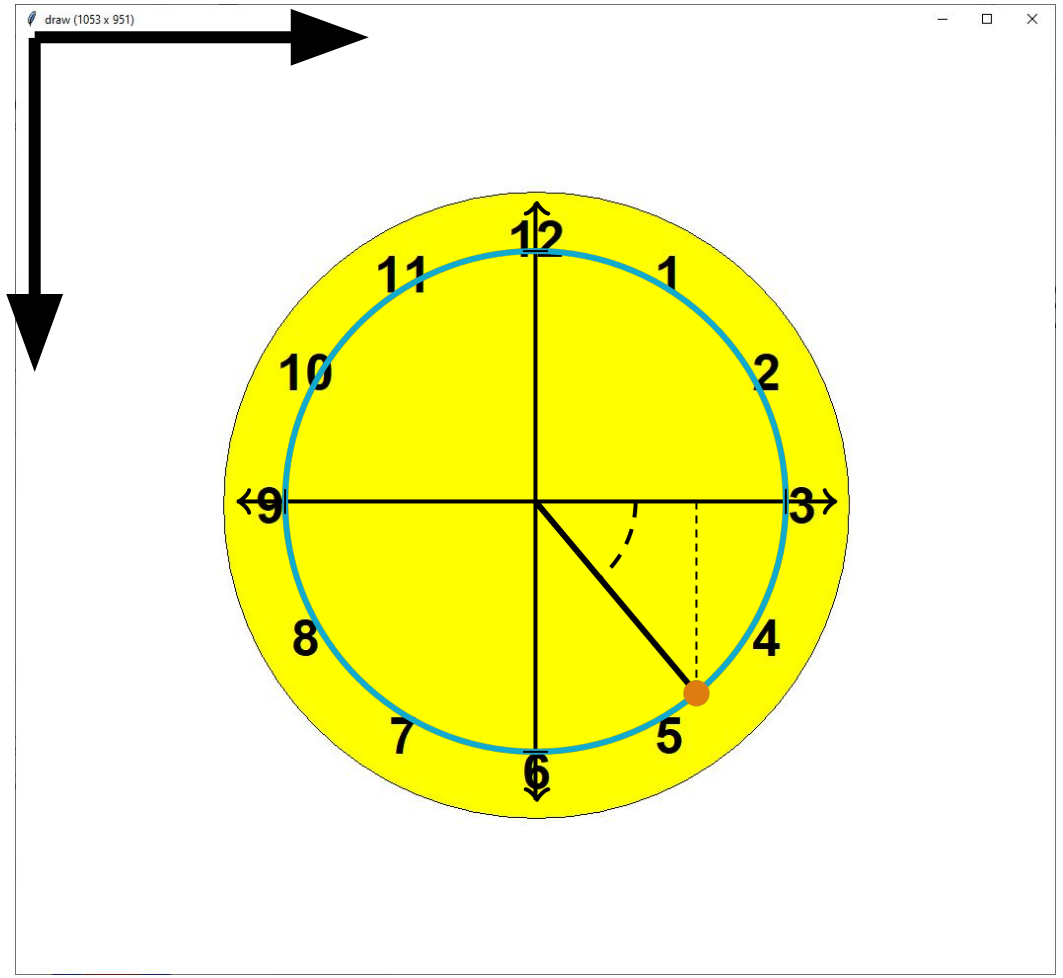
Resizable!

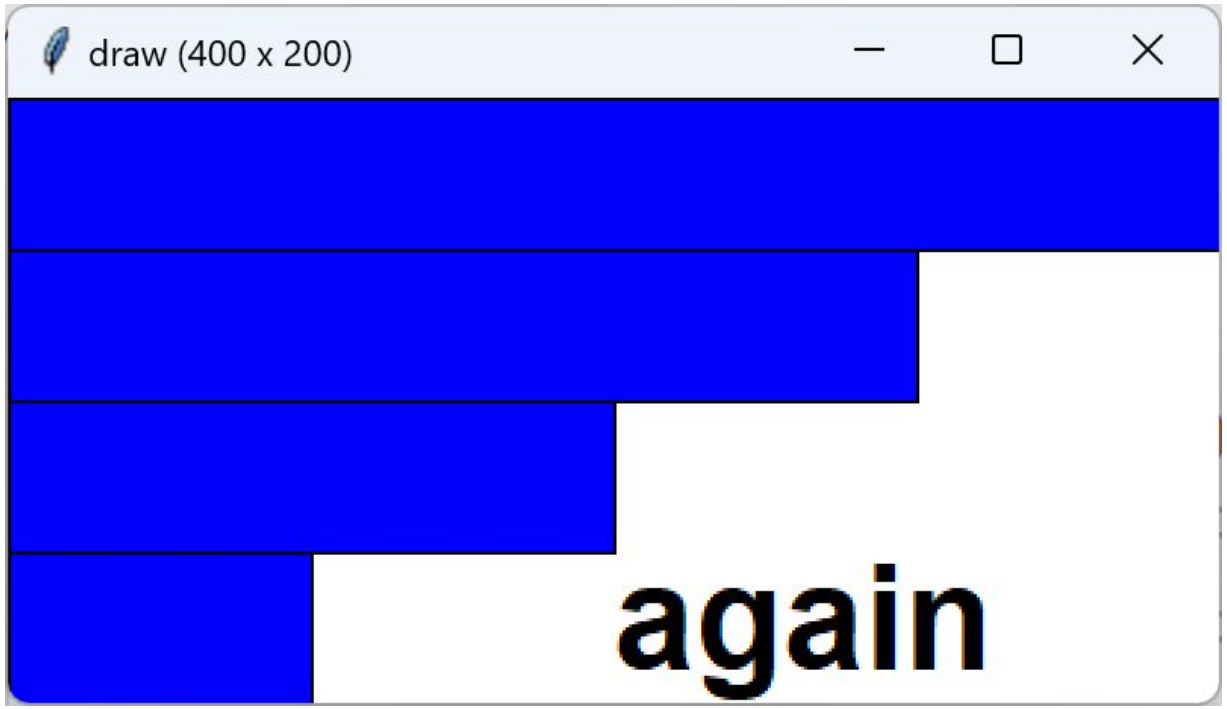


Free Response



Free Response




draw (400 x 200)
again

Animations!

- The “Model”
- Understand MVC
  - Don’t worry if you don’t get it now, you will understand it eventually
- Usual Tasks
  - Moving objects around
    - Timers
  - Different animation “states”: e.g., start screen, paused
    - Keyboard events
    - Using the model to track the “state”
  - Mouse click inside objects
  - …

```
from cmu_graphics import *

# This is called when the program starts
def onAppStart(app):
    pass

# This is called every time one key is pressed
def onKeyPress(app, key):
    pass

# This is called every time a mouse button is pressed
def onMousePress(app, x, y):
    pass

# This is called many times to refresh the window
def redrawAll(app):
    pass

# This is called "often" (def. by app.stepsPerSecond)
def onStep(app):
    pass

# This is how you run the program
runApp(width=800, height=800)
```

appStarted

- Runs once when the animation starts
- Useful to initialize values used in the animation

The “Model”

Responsible for managing the data, logic, and state of the application
For **now**, think of it as a container of information.

app

app.message = “Hello”
app.userName = “Eduardo”
app.ballVelocity = 30.5
…

Store Data

`app.<variable name> = <value>`

Retrieve Data

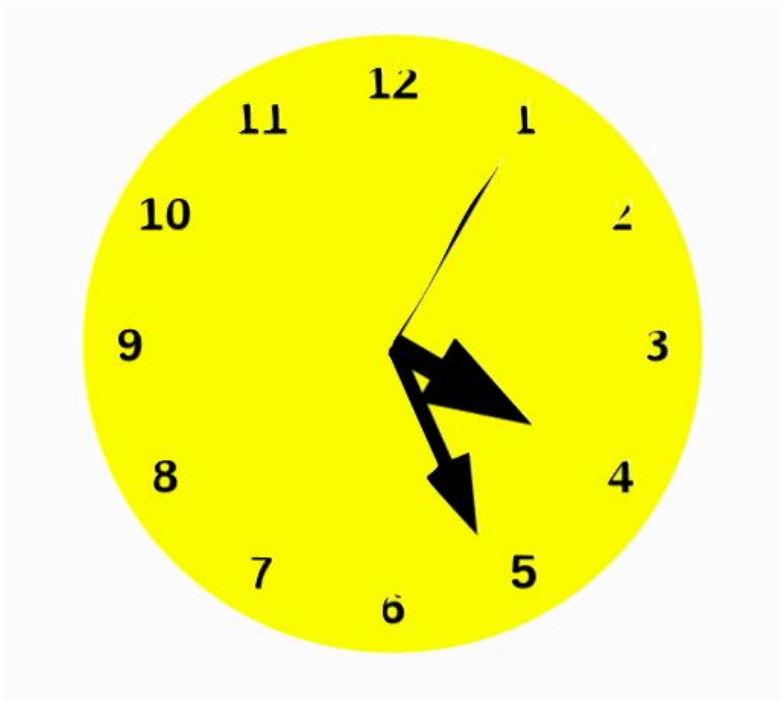
Simply use `app.<variable name>`

Timed events: `onStep`

```
def onStep(app):
```

- The library calls this function `app.stepsPerSecond` times per second
- Useful for **updating the model** according to the animation task
  - Changing the position of elements (simulating movement)
  - Implementing timers: e.g., countdown

Example: Animated clock hands



Capturing events

- onKeyPress(app, key)
  - Example: Press ‘p’ to pause the clock
  - Useful for debugging:
    - Press a key to advance the animation one step further

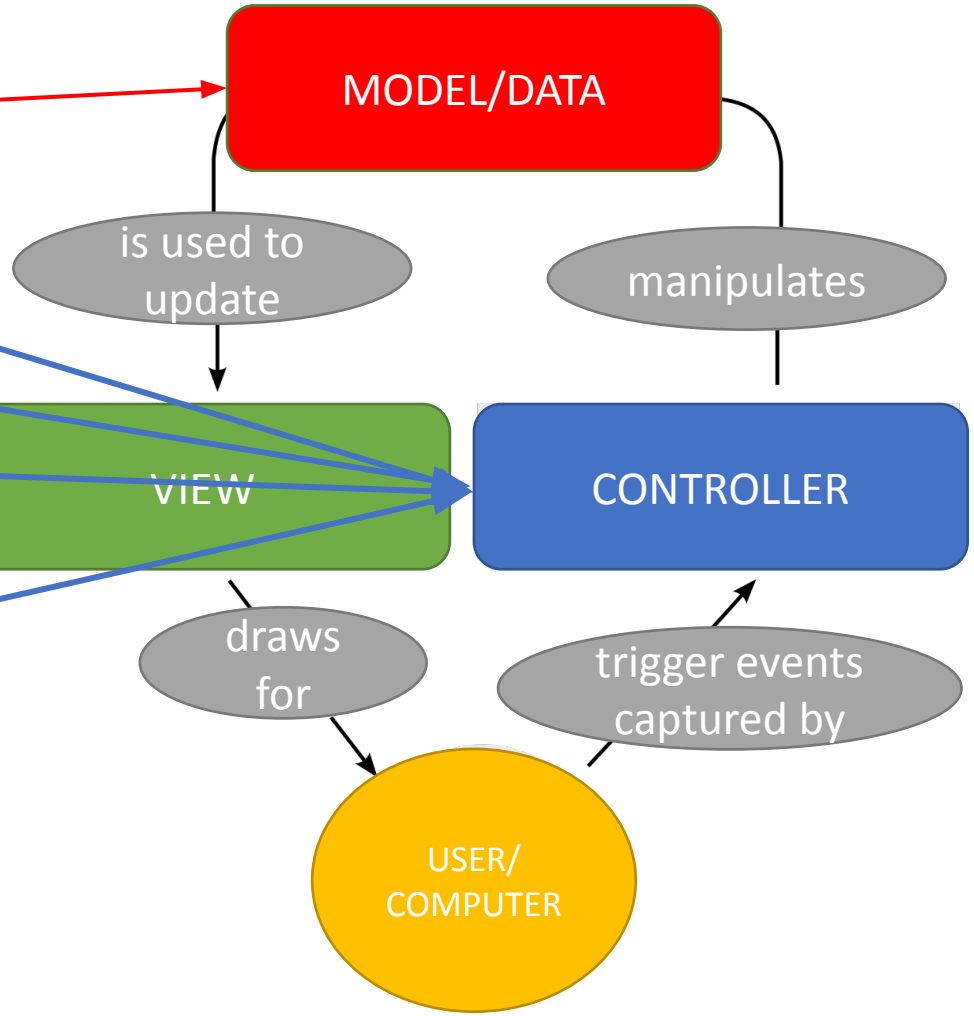
Capturing events

- onMousePress(app, ev)
  - Example: Mouse press over the clock to change its color

Common Animation Features

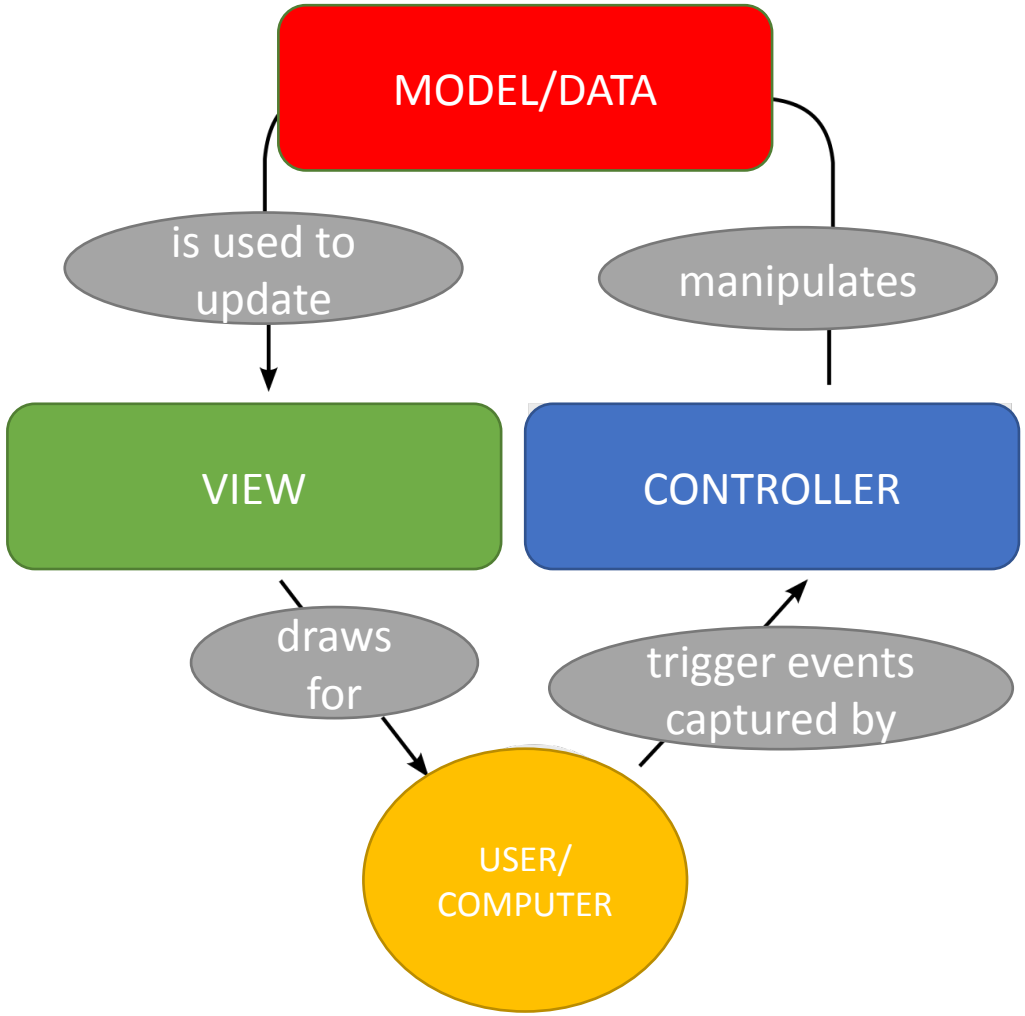
- Tracking state and changing the screen based on it
  - Example: Welcome screen, pause screen, game over, etc
- Moving objects, bounding, collisions between objects

```python
from cmu_graphics import *

# This is called when the program starts
def onAppStart(app):
    pass

# This is called every time one key is pressed
def onKeyPress(app, key):
    pass

# This is called every time a mouse button is pressed
def onMousePress(app, x, y):
    pass

# This is called many times to refresh the window
def redrawAll(app):
    pass

# This is called "often" (def. by app.stepsPerSecond)
def onStep(app):
    pass

# This is how you run the program
runApp(width=800, height=800)
```



The Three (3) RULES of MVC

- You never call the view or the controllers. The animation framework calls these for you.
- Controllers can only update the model. They cannot update the view.
- The view can never update the model.



The Three (3) RULES of MVC

- You never call the view or the controllers. The animation framework calls these for you.
- Controllers can only update the model. They cannot update the view.
- The view can never update the model.