CS65: Introduction to Computer Science

Drawing on a Canvas using graphics library

Md Alimoor Reza
Assistant Professor of Computer Science

Topics

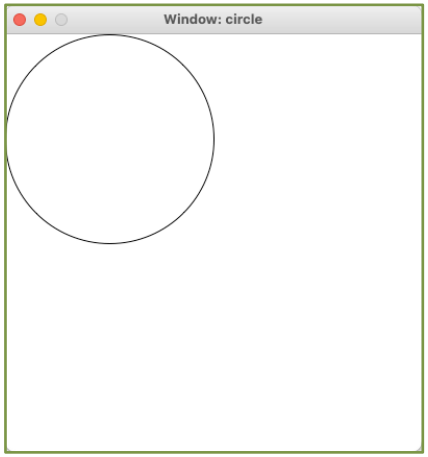
- Drawing shapes inside the window
  - Circle
  - Rectangle
  - Line, Text, and combinations of these shapes
- Changing coordinate system
- Mouse interaction inside graphics window

Drawing inside the window

- Step 1: Construct a circle
  - Step 1.1: construct a point —> the center of the circle
  - Step 1.2: fix the radius
  - Step 1.3: put them together

- Step 2: **Draw** the newly constructed circle inside the window

```
from graphics import *


def create_simple_window_v1():

    window = GraphWin("New window", 400, 400)

    point  = Point(100, 100)       # step 1.1
    radius = 100                   # step 1.2
    circle = Circle(point, radius) # step 1.3

    circle.draw(window)            # step 2

    return window


w1 = create_simple_window_v1()
```



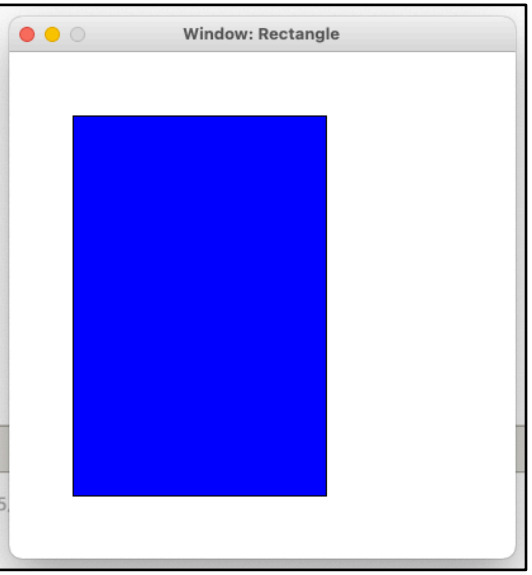
Drawing rectangle

- Step 1: Construct a rectangle
  - Step 1.1: construct a point —> one corner
  - Step 1.2: construct a point —> opposite corner
  - Step 1.3: put them together

- Step 2: **Draw** the newly constructed rectangle inside the window

```
from graphics import *


def create_simple_window_w_rect():
    window = GraphWin("Window: Rectangle", 400, 400)

    point1 = Point(50,50)
    point2 = Point(250, 350)

    rect   = Rectangle(point1, point2)
    rect.setFill("blue")
    rect.draw(window)

    return window


w1 = create_simple_window_w_rect()
```

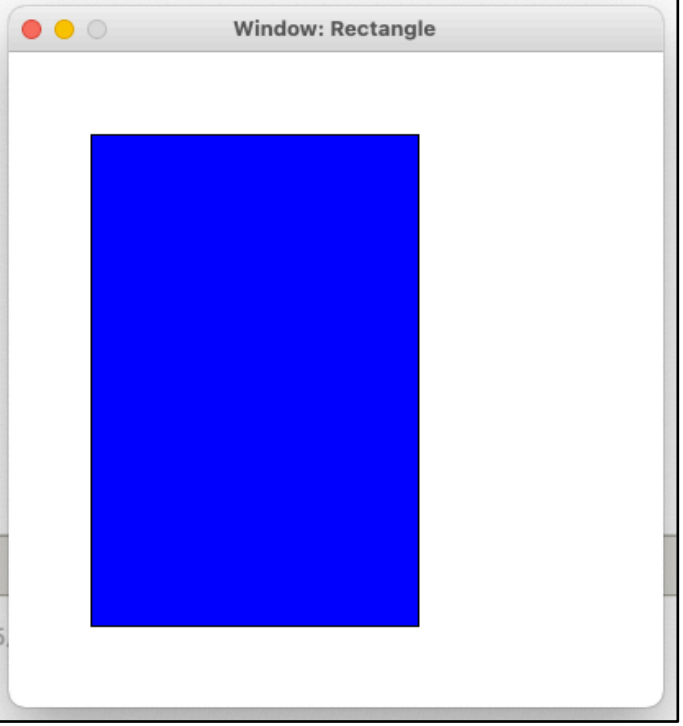
```
n 3.7.9 (bundled)
cd /Users/reza/Class_and_Research/drake_teaching/CS65
Run lec6_rect.py
Run lec6_rect.py
```



Coding demo

```
from graphics import *


def create_simple_window_w_rect():
    window = GraphWin("Window: Rectangle", 400, 400)

    point1 = Point(50,50)
    point2 = Point(250, 350)

    rect   = Rectangle(point1, point2)
    rect.setFill("blue")
    rect.draw(window)

    return window


w1 = create_simple_window_w_rect()
```

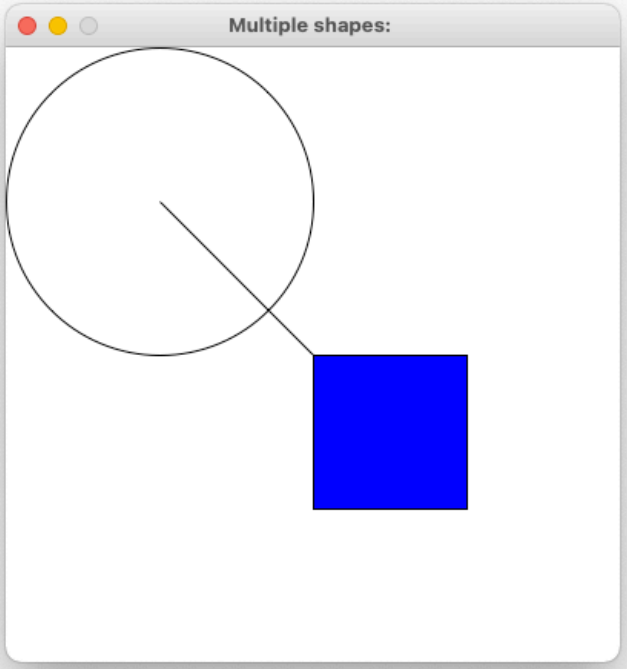
```
n 3.7.9 (bundled)
cd /Users/reza/Class_and_Research/drake_teaching/CS65
Run lec6_rect.py
Run lec6_rect.py
```



Draw multiple shapes

- Drawing circle, rectangle, and a line connecting them
- Demo

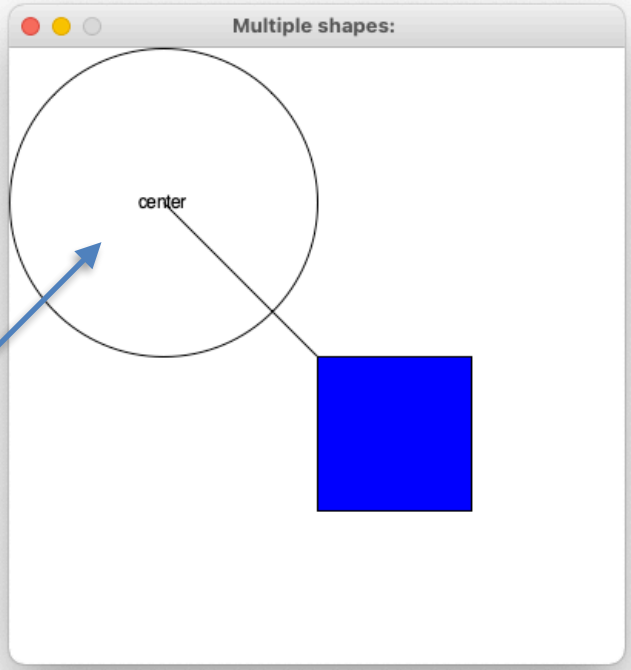
```
from graphics import *

def create_random_shapes():

    win = GraphWin("Multiple shapes: ", 400, 400)

    # create circle
    cir_center = Point(100, 100)
    cir_radius = 100
    my_cir     = Circle(cir_center, cir_radius)


    # create rectangle
    rect_point_a = Point(200, 200)
    rect_point_b = Point(300, 300)
    my_rect      = Rectangle(rect_point_a, rect_point_b)
    my_rect.setFill("blue")


    # line connecting circle and rectangle
    my_line      = Line(cir_center, rect_point_a)

    # draw all the components
    my_cir.draw(win)
    my_rect.draw(win)
    my_line.draw(win)

    return win


new_window = create_random_shapes()
```



Draw multiple shapes

- Drawing **Text** inside the window
- Demo

```
from graphics import *

def create_random_shapes():

    win = GraphWin("Multiple shapes: ", 400, 400)

    # create circle
    cir_center = Point(100, 100)
    cir_radius = 100
    my_cir     = Circle(cir_center, cir_radius)


    # create rectangle
    rect_point_a = Point(200, 200)
    rect_point_b = Point(300, 300)
    my_rect      = Rectangle(rect_point_a, rect_point_b)
    my_rect.setFill("blue")


    # line connecting circle and rectangle
    my_line      = Line(cir_center, rect_point_a)

    # text
    text_point   = Point(100, 100)
    my_text      = Text(text_point, "center")

    # draw all the components
    my_cir.draw(win)
    my_rect.draw(win)
    my_line.draw(win)
    my_text.draw(win)

    return win
```



Topics

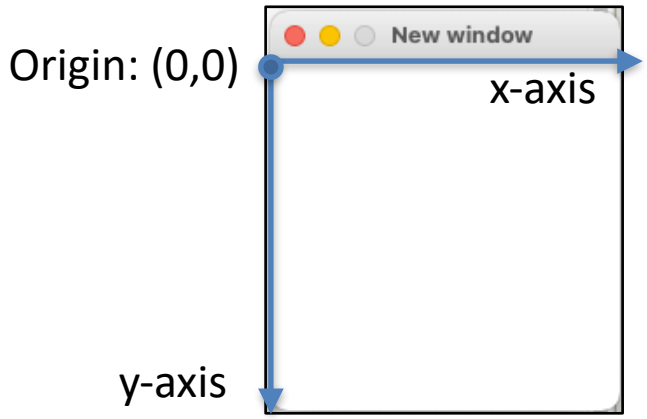
- Drawing shapes inside the window
  - Circle
  - Rectangle
  - Line, Text, and combinations of these shapes

- Changing coordinate system

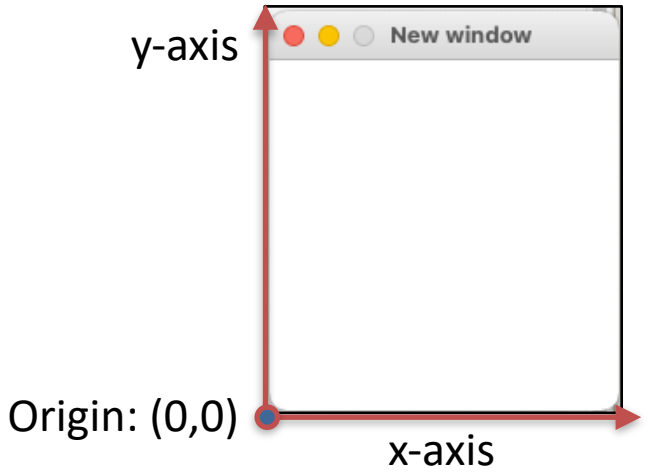
- Mouse interaction inside graphics window

Changing coordinate system

- Default coordinate system:
  - x: top-left —> top-right
  - y: top-left —> bottom-left

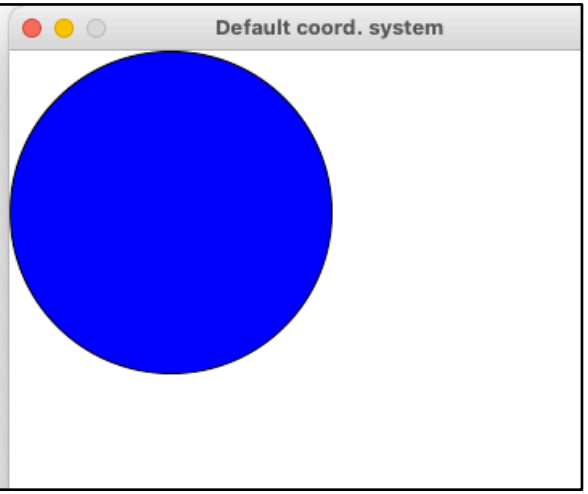


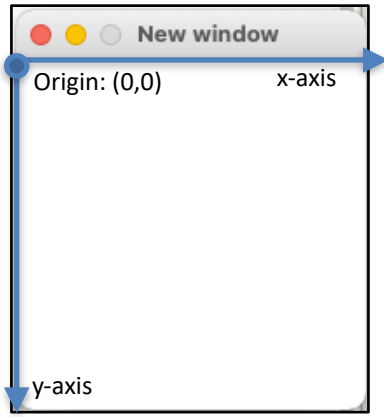
- Transforming into traditional coordinate system
  - x: bottom-left —> bottom-right
  - y: bottom-left —> bottom-up



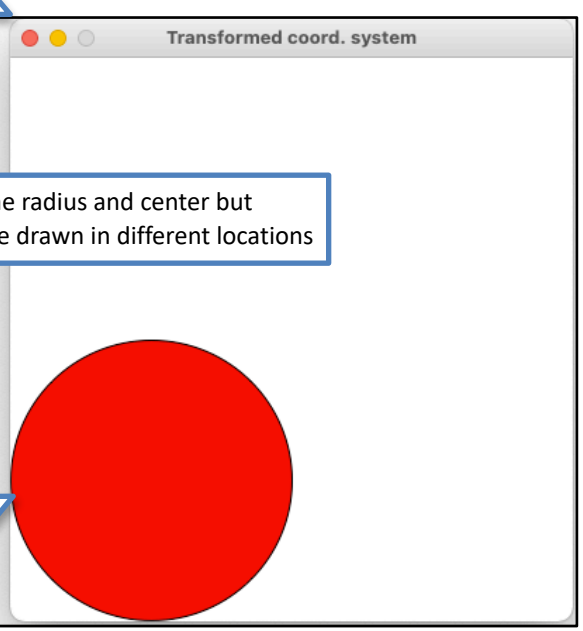
Changing coordinate system

```
from graphics import *

def create_circle_default_coord(px, py, radius):

    window = GraphWin("Default coord. system", 400, 400)

    # create circle
    cir_center = Point(px, py)
    cir_radius = radius
    my_cir     = Circle(cir_center, cir_radius)
    my_cir.setFill("blue")

    my_cir.draw(window)

    return window

win_default = create_circle_default_coord(100, 100, 100)
```

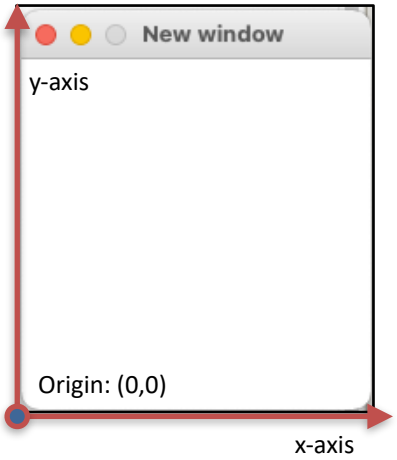





```
from graphics import *

def create_circle_transformed_coord(px, py, radius):


    window = GraphWin("Transformed coord. system", 400, 400)

    #---------- changing the coordinate system ----------
    # lower left  corner becomes 0,0
    # upper right corner becomes 400, 400
    window.setCoords(0,0, 400, 400)
    #----------------------------------------------------

    # create circle
    cir_center = Point(px, py)
    cir_radius = radius
    my_cir     = Circle(cir_center, cir_radius)
    my_cir.setFill("red")

    my_cir.draw(window)

    return window


# ------ lower left corner (0,0) and upper left corner (400, 400) ------
win_trans = create_circle_transformed_coord(100, 100, 100)
```

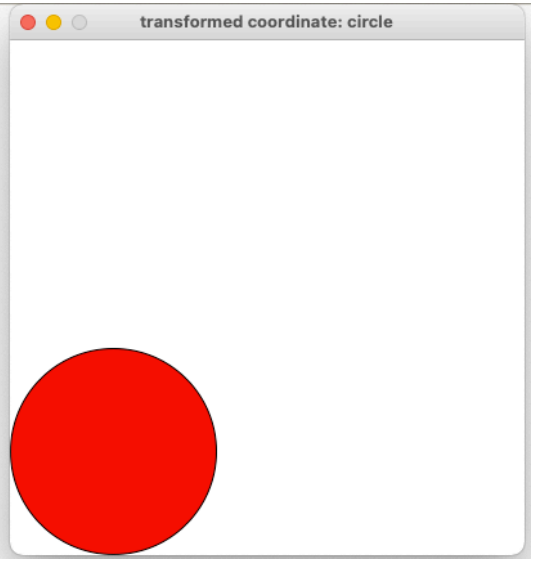


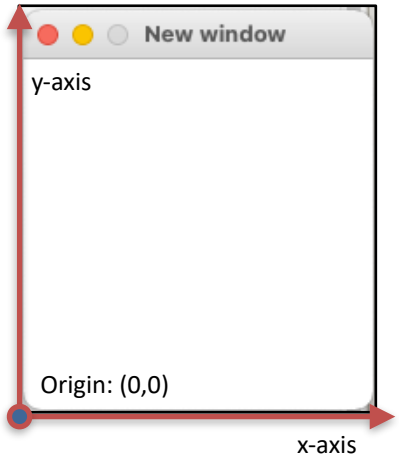
Same radius and center but
were drawn in different locations



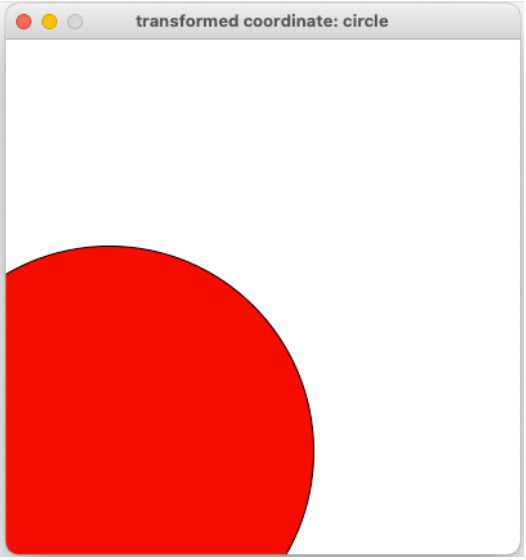
Changing coordinate system

```
from graphics import *

def create_transformed_window_v1(x, y, radius):

    window = GraphWin("transformed coordinate: circle", 400, 400)

    window.setCoords(-100, -100, 400, 400)

    circle_center = Point(x, y)
    circle_radius = radius
    tr_circle = Circle(circle_center, circle_radius)
    tr_circle.setFill("red")
    tr_circle.draw(window)

    return window

my_win = create_transformed_window_v1(0, 0, 100)
```





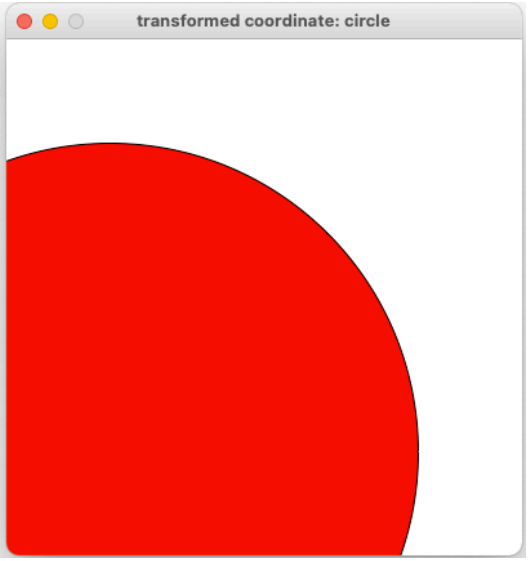

```
from graphics import *

def create_transformed_window_v1(x, y, radius):

    window = GraphWin("transformed coordinate: circle", 400, 400)

    window.setCoords(-100, -100, 400, 400)

    circle_center = Point(x, y)
    circle_radius = radius
    tr_circle = Circle(circle_center, circle_radius)
    tr_circle.setFill("red")
    tr_circle.draw(window)

    return window

my_win = create_transformed_window_v1(0, 0, 200)
```

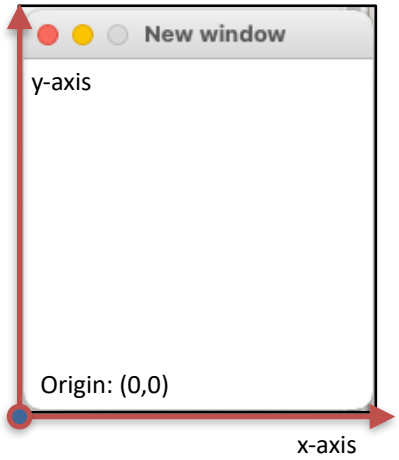


The red circle has been clipped

Changing coordinate system

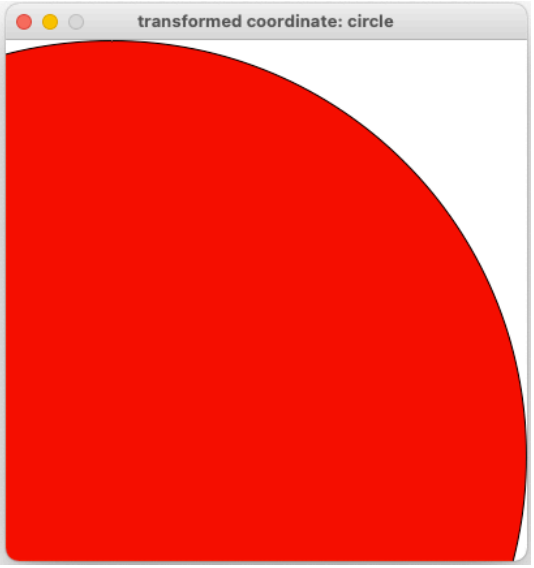
```
from graphics import *

def create_transformed_window_v1(x, y, radius):

    window = GraphWin("transformed coordinate: circle", 400, 400)

    window.setCoords(-100, -100, 400, 400)

    circle_center = Point(x, y)
    circle_radius = radius
    tr_circle = Circle(circle_center, circle_radius)
    tr_circle.setFill("red")
    tr_circle.draw(window)

    return window

my_win = create_transformed_window_v1(0, 0, 300)
```





The red circle has been clipped

```
from graphics import *

def create_transformed_window_v1(x, y, radius):

    window = GraphWin("transformed coordinate: circle", 400, 400)

    window.setCoords(-100, -100, 400, 400)

    circle_center = Point(x, y)
    circle_radius = radius
    tr_circle = Circle(circle_center, circle_radius)
    tr_circle.setFill("red")
    tr_circle.draw(window)

    return window

my_win = create_transformed_window_v1(0, 0, 400)
```



The red circle has been clipped

Topics

- Drawing shapes inside the window
  - Circle
  - Rectangle
  - Line, Text, and combinations of these shapes
- Changing coordinate system
- Mouse interaction inside graphics window

Interaction with the user using Mouse

- GraphWin() has a function that allows us to identify the location of your mouse-click
  - 2D coordinate
  - represented by Point object

```
from graphics import *


window = GraphWin("Mouse interaction", 400, 400)

mouse_point1 = window.getMouse()
mouse_point2 = window.getMouse()

print(mouse_point1)
print(mouse_point2)
```

More on function calls
Lab 3 (Task 5)

```
from graphics import *

def draw_one_rectangle(x1, y1, x2, y2, color, window):

    # your code here
    # ...
    # end of your code
    return None


def build_staircase():

    window = GraphWin("Staircase", 400, 400)
    window.setCoords(0, 0, 400, 400)


    # your code here
    # ...
    side_length_of_rect = 100
    x1 = 0
    y1 = 0
    x2 = 100
    y2 = 100
    draw_one_rectangle(x1, y1, x2, y2, "blue", window)

    # alternatively
    # draw_one_rectangle(0, 0, 100, 100, "blue", window)
    # ...
    # ...
    # end of your code

    return None

build_staircase()
```

Summary

- **Takeaway from this lecture**
  - Basics shapes are already defined, you just need to draw them according in specific ways
    - Circle, Rectangle, Triangle
  - Change of coordinates to draw in a different way
  - Mouse interaction with the user

- **To do:**
  - Read: https://mcsp.wartburg.edu/zelle/python/graphics/graphics/index.html

- **Announcements:**
  - Assignment 1 will be out soon! It will be due in 2 weeks.