Chapter 4: Objects and Graphics Part 2

Jan 23, 2020

Today’s Outline

- Review:
  - Object-oriented programming concepts
  - Simple graphics programming with graphics.py
- Interactive Graphics
- GUIs

Quiz 2

Quizzable topics:

Mostly will be focusing on theory from Chapter 3 Lectures.

Format: 10 Multiple Choice questions

Deadline: Sunday (Jan 26) at 11:55pm

Lab 2

Topic: Working with Numbers

Posted: Today after class

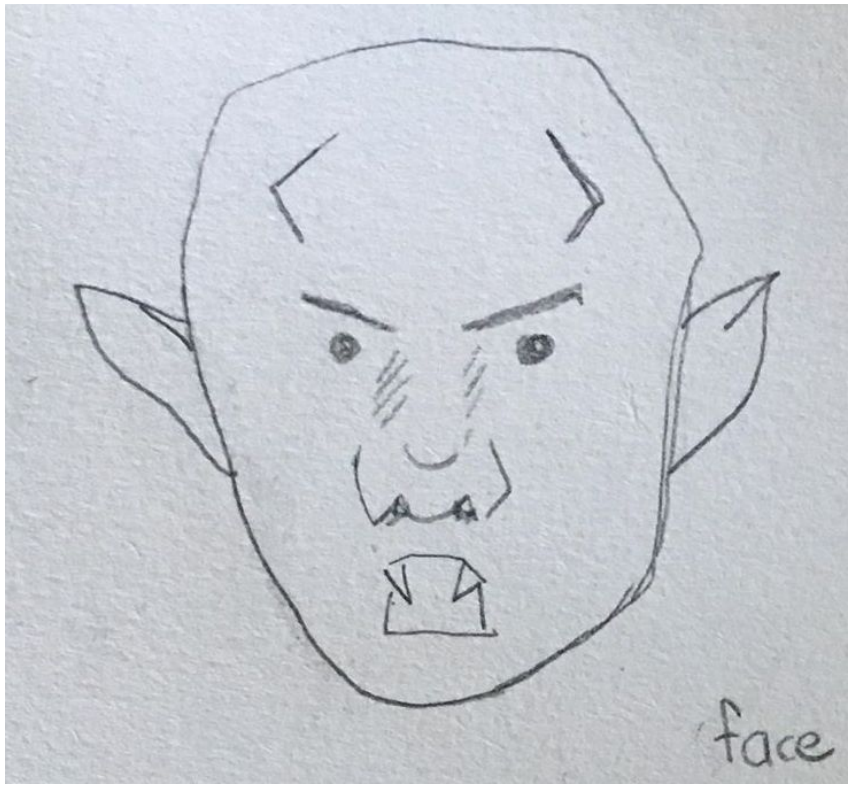
Tutorial: Tomorrow (Friday) 8:30-11:30am, MacOdrum Library

Deadline: next Friday (Jan 31) in Lab 3

*Remember to submit Lab 1 by 9:00am tomorrow!

Character Design Bonus Assignment

Determine how you would draw a representation of your fantasy character using the given classes, (Point, Line, Circle, Oval, Rectangle, Polygon, and Text).



It is not too late to submit!
Email me your drawing.

Object-oriented Programming

- Object-oriented programming is a programming paradigm based on the concept of “objects”

Purpose: build more complex programs
build maintainable programs

Class

A **class** is a blueprint of how to make an object

An **object** is an instance of a class.

There can be many objects/instances of a class.

Class

A **class** is a blueprint of how to make an object

An **object** is an instance of a class.

There can be many objects/instances of a class.

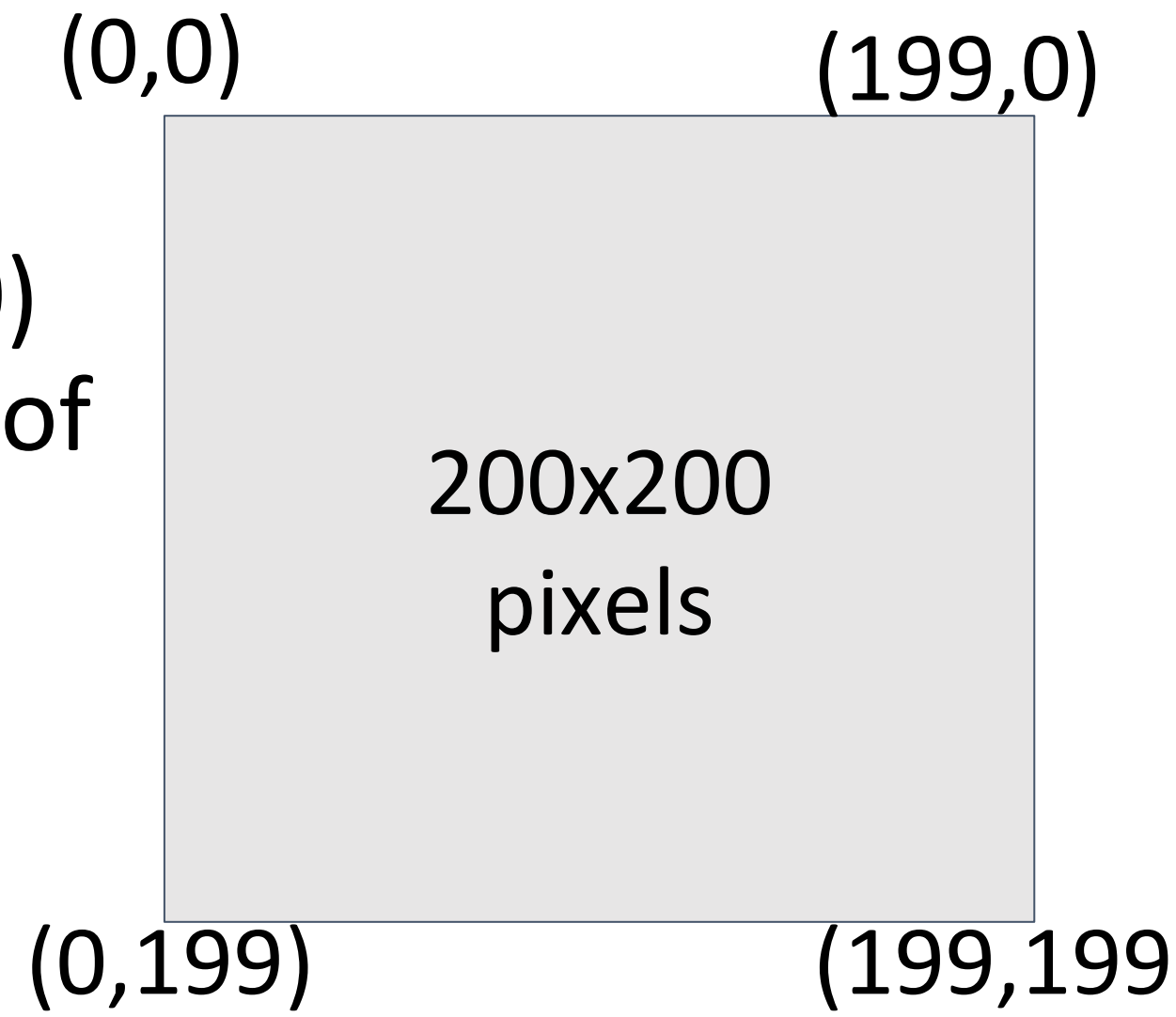
Graphics library classes:

Point, Line, Circle, Rectangle, Oval, Polygon, Text

Graphics Library objects:

Cartesian Coordinates

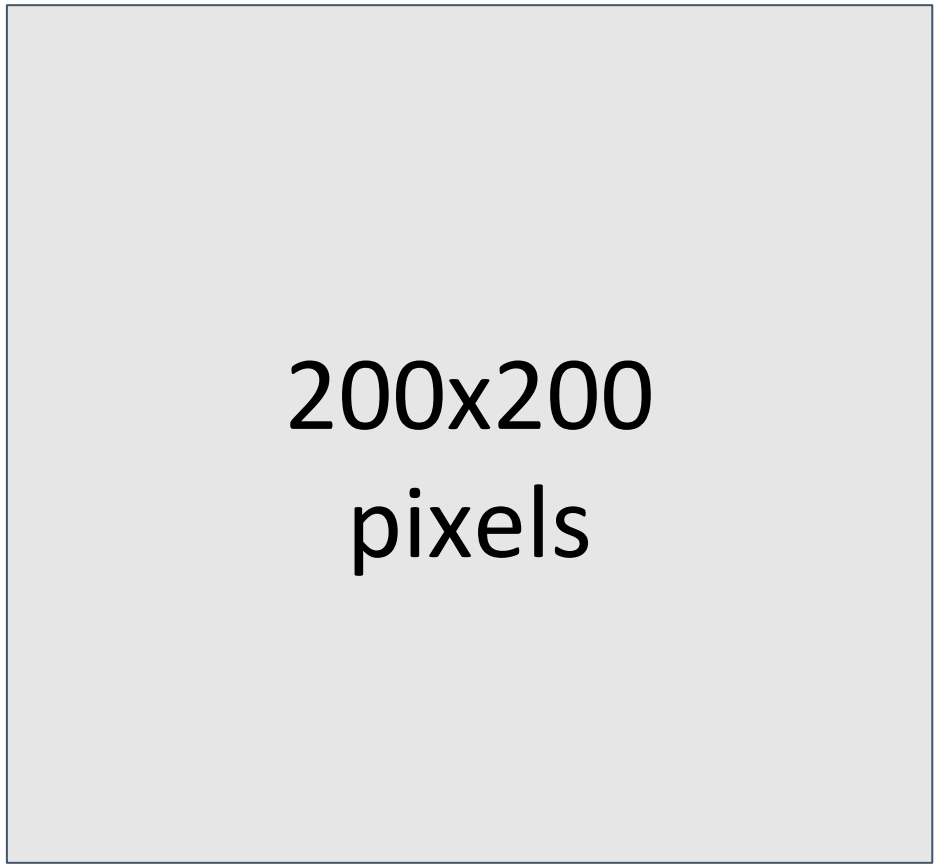
Traditionally (x,y) = (0,0) is in the top left corner of the graphics window.



Point Objects

```
#What will the result be?

import graphics
win = graphics.GraphWin()
p1 = graphics.Point(150,100)
p2 = graphics.Point(10,70)
p1.draw(win)
p2.draw(win)
```



Object Interactions

How many objects are involved in this code?

```
win = graphics.GraphWin()
circ = graphics.Circle(graphics.Point(100,100),30)
circ.draw (win)
```

Object Interactions

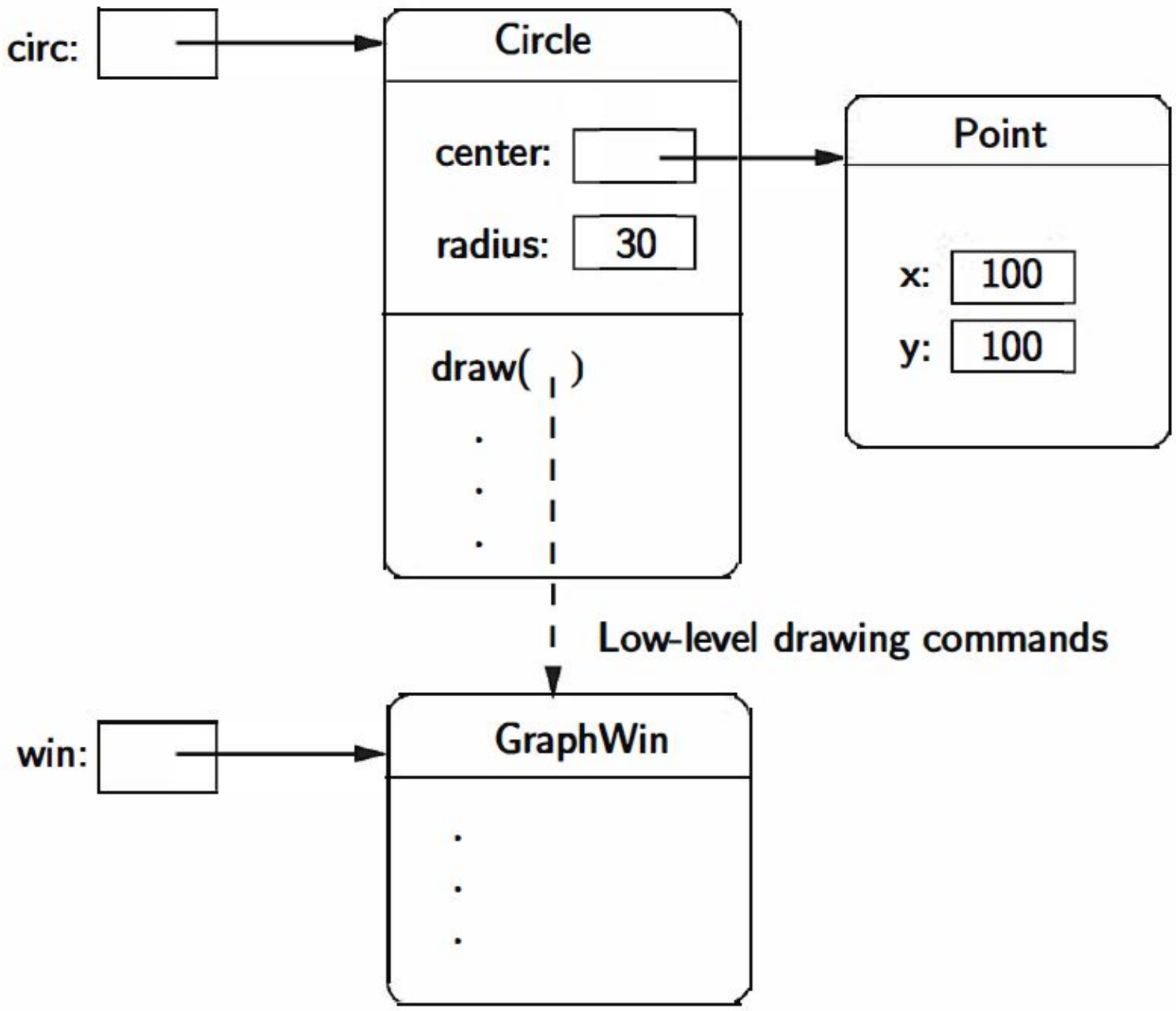
Where are the objects?

window object

point object

```
win = graphics.GraphWin()
circ = graphics.Circle(graphics.Point(100,100),30)
circ.draw (win)
```

circle object


circ:
Circle
center:
radius:
30
draw( )
Point
x:
100
y:
100
Low-level drawing commands
win:
GraphWin

Methods for Graphics Objects

setFill (colour)

setOutline (colour)

draw (GraphWin)

undraw()

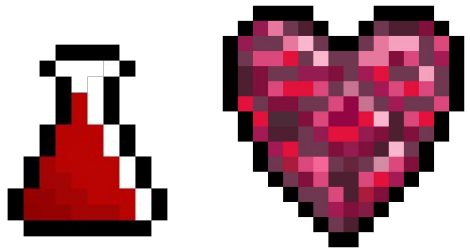
move (dx,dy)

Methods

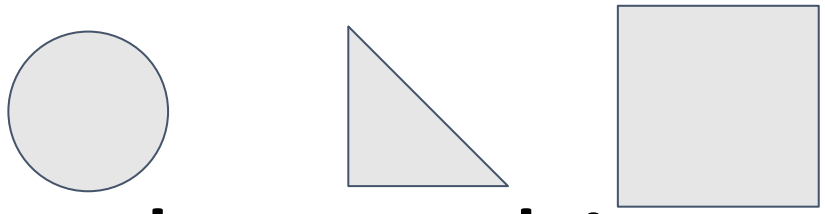
- Methods are used to help objects communicate with each other
- There are two categories of methods that can be performed on objects
- **Accessors** are used to find out information about an object:
  - getX(), getY(), draw()
- **Mutators** are used to change the internal state of an object:
  - move(), setFill(), setOutline()

4 Object Oriented Programming Principles

Inheritance



Encapsulation



Polymorphism

Abstraction

4 Object Oriented Programming Principles

1. Encapsulation
   a. to access object information we need to use an accessor method
   b. to draw an object we need to use a draw() method
2. Abstraction
   a. we don’t have to understand everything about how to draw a circle or oval to create and draw them
3. Inheritance
   a. some of the more shape classes could have been developed from a parent class for shapes
4. Polymorphism
   a. the draw() method works on all of the shapes even though each shape is different

RBG Colours

Additive colour model of (red, green, blue)

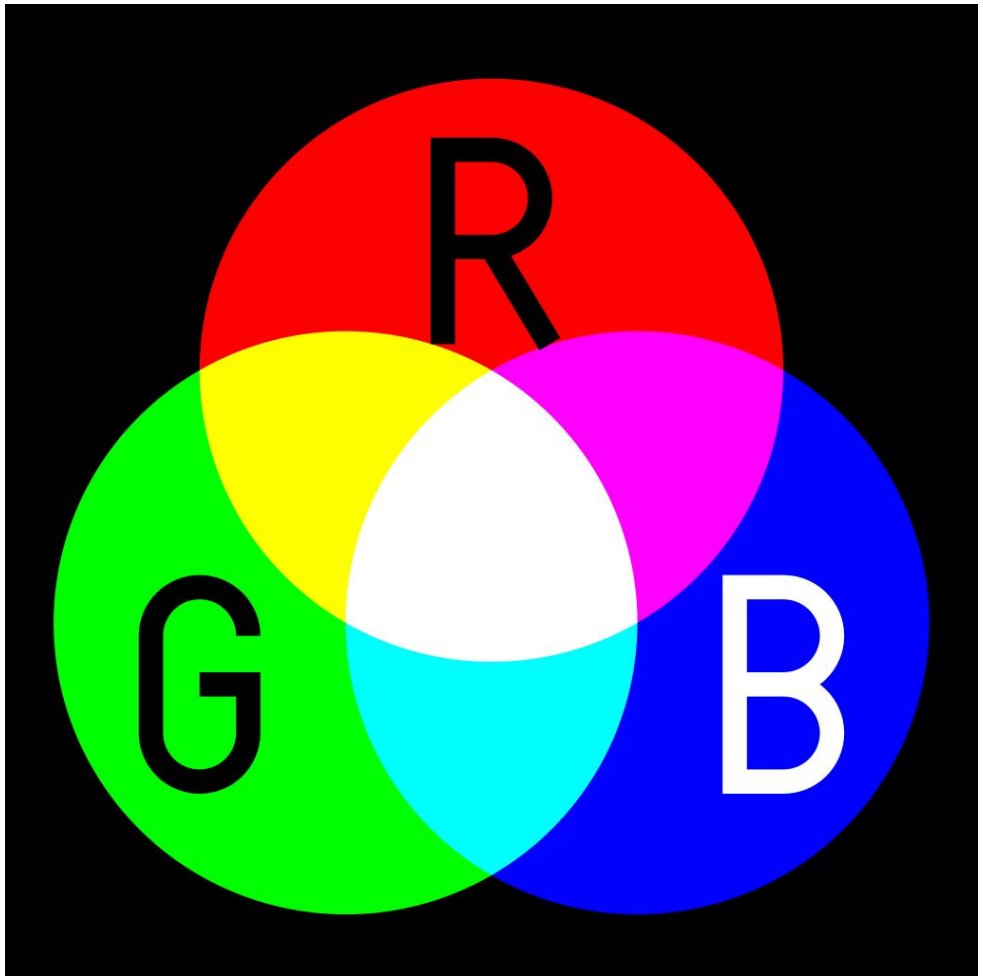
Ranges of colours are ints on a scale from 0-255

#magenta

circ.setFill (color_rgb (255 , 0 , 255))

#yellow

circ.setFill (color_rgb (255 , 255 , 0))



Simple Example 1

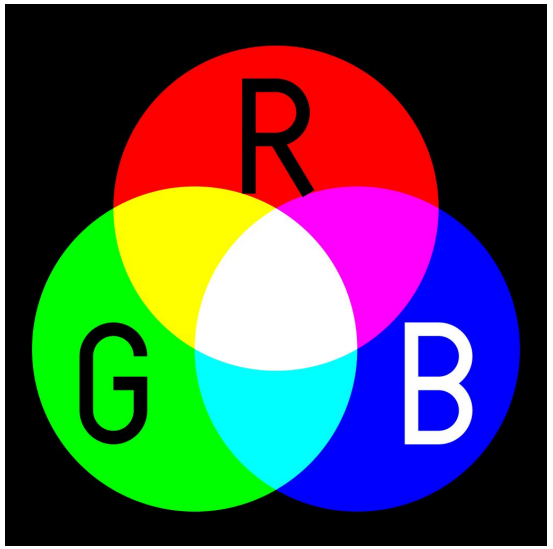
Describe what object will be created:

```
c = graphics.Circle (graphics.Point (30 , 40) , 25)
c.setFill ("blue")
c.setOutline ("red")
```

Simple Example 2

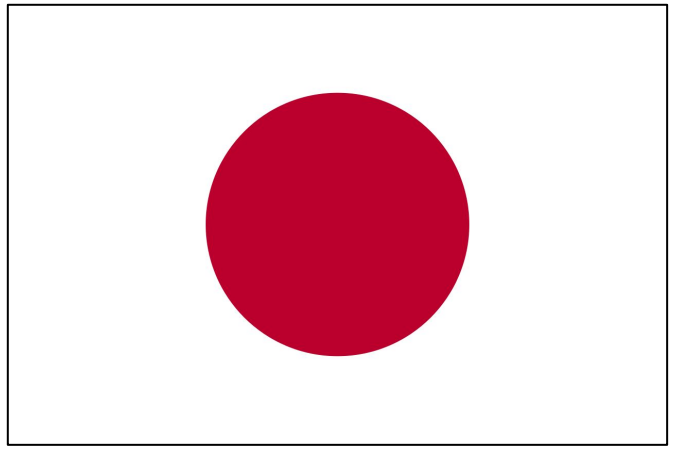
Describe what object will be created:

```
r = graphics.Rectangle (graphics.Point (20,20) , graphics.Point (140,140))
r.setFill (graphics.color_rgb (0,255,255))
```



Flag Problem

Write a program that can draw the flag of Japan.



Write another program that can draw the flag of France.

RGB: (0, 85, 164)

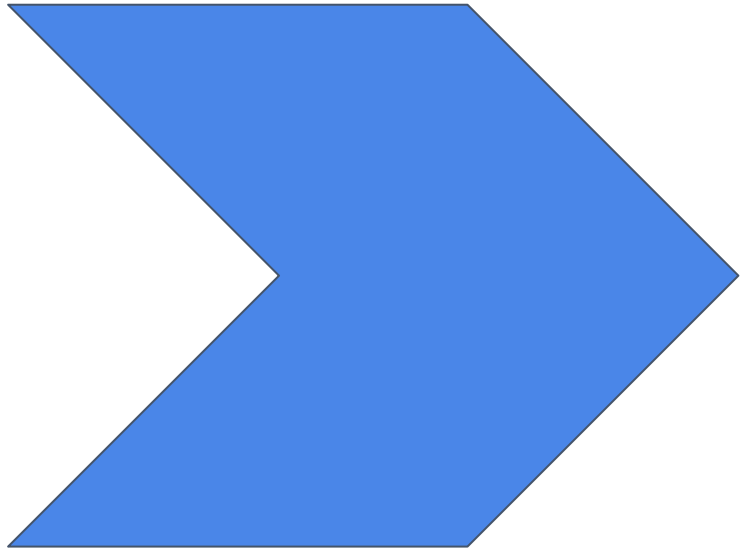
RBG: (255, 255,255)

RGB: (239, 65, 83)

Polygon Problem

Polygon (point 1, point2 , point3 , . . .) Constructs a polygon object with the given points as vertices.

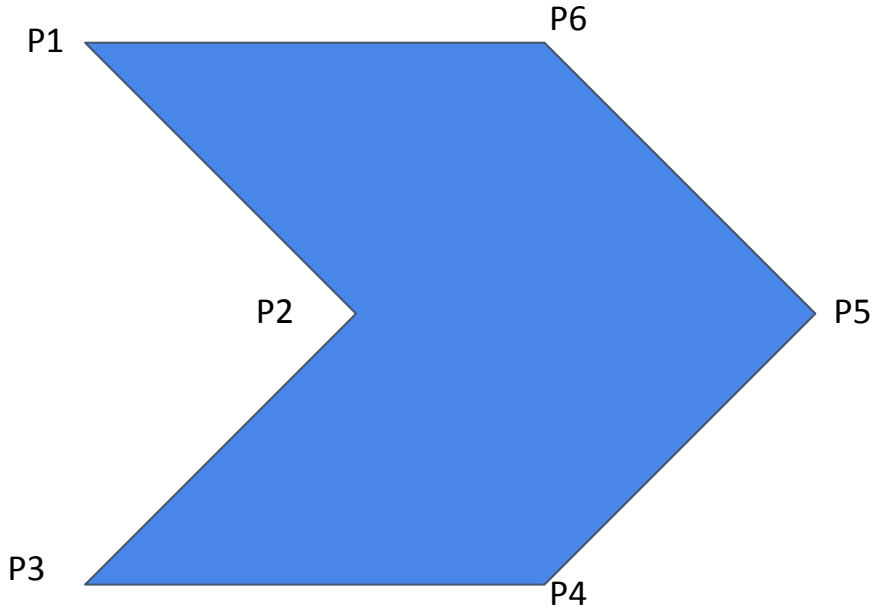
Use a polygon object to draw a chevron:



Polygon Problem

Polygon (point 1 , point2 , point3 , . . .) Constructs a polygon object with the given points as vertices.

Use a polygon object to draw a chevron:



Simple Example 3

Describe what object will be created:

```
shape = graphics.Polygon (graphics.Point (0,0), graphics.Point(199,199),
graphics.Point (0,199), graphics.Point (199,0))

shape.setFill ("orange")
shape.draw(win)
```

Drawing Eyes on a Face Code

```
leftEye= graphics.Circle(graphics.Point(60,70),8)
leftEye.setFill("red")
rightEye = leftEye
rightEye.move(70,0)
```

Aliasing

Problem: using rightEye = leftEye causes both variables to refer to the same object in the memory

Solution: the graphics.py library, the clone() function prevents this from happening

rightEye = leftEye.clone ( )

Graphics Window Size and Title

```
win = graphics.GraphWin ("title", width, height)
```

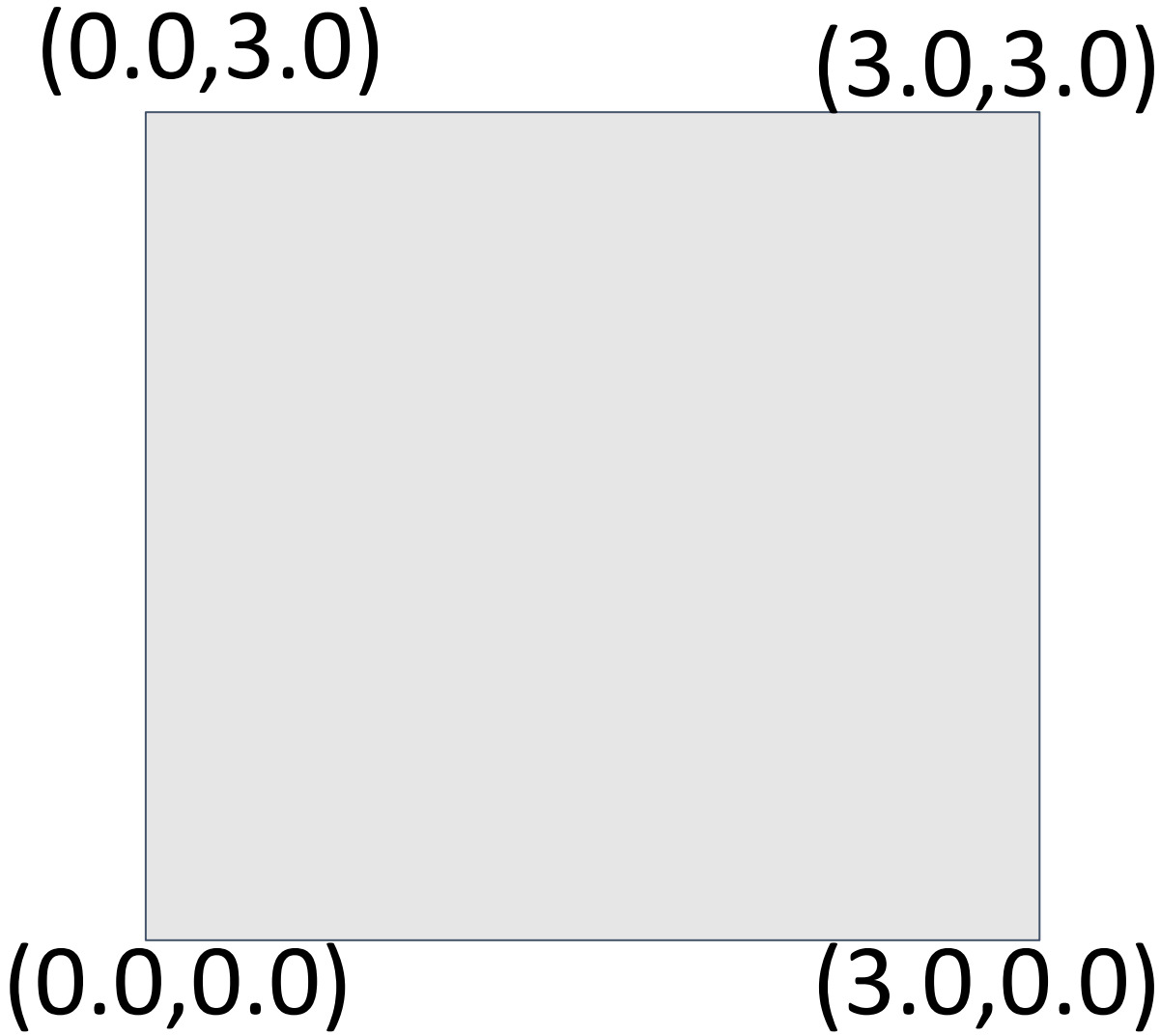
Example:

```
win = graphics.GraphWin ("My New Window", 250 , 750)
```

Coordinate Transformation Example

```
win = graphics.GraphWin()
win.setCoords(0.0, 0.0, 3.0 , 3.0 )
```

The .setCoords() method was used to set the coordinates of the lower left corner to (0.0, 0.0) and the upper right corner to (3.0, 3.0),

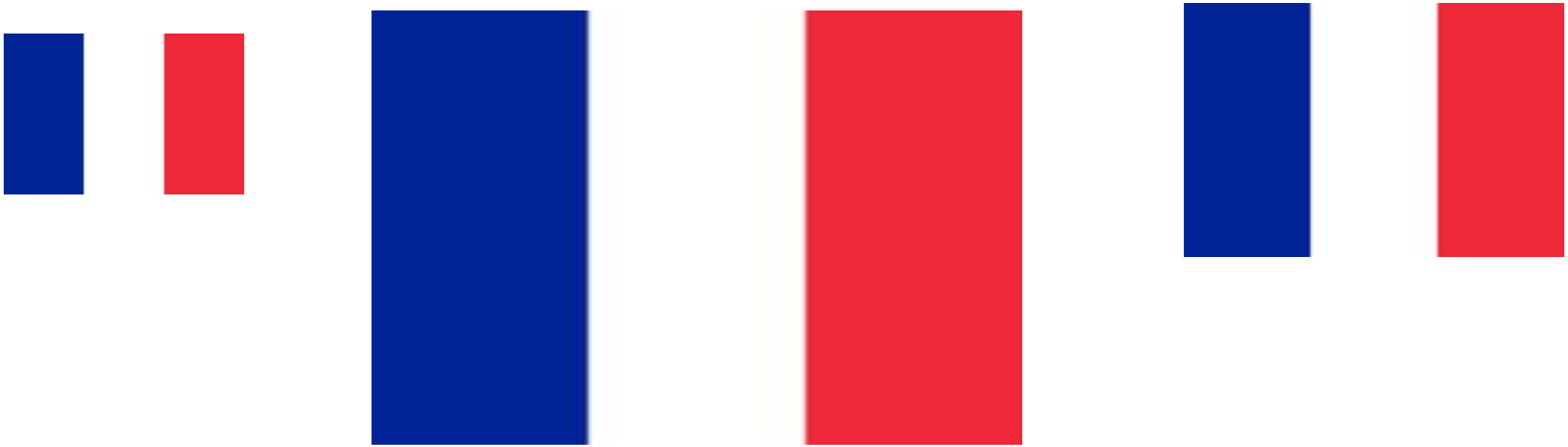


Coordinate Transformation Example

```
win = graphics.GraphWin("Tic-Tac-Toe",500,500)
win.setCoords(0.0, 0.0, 3.0 , 3.0 )
# Draw vertical lines
graphics.Line(graphics.Point(1,0), graphics.Point(1,3)).draw(win)
graphics.Line(graphics.Point(2,0), graphics.Point(2,3)).draw(win)
# Draw horizontal lines
graphics.Line(graphics.Point(0,1), graphics.Point(3,1)).draw(win)
graphics.Line (graphics.Point (0,2), graphics.Point (3,2)).draw(win)
```

Big French Flag

Modify the French flag program so that it can draw the flag on any window size. The length of the flag is 1.5x the width.



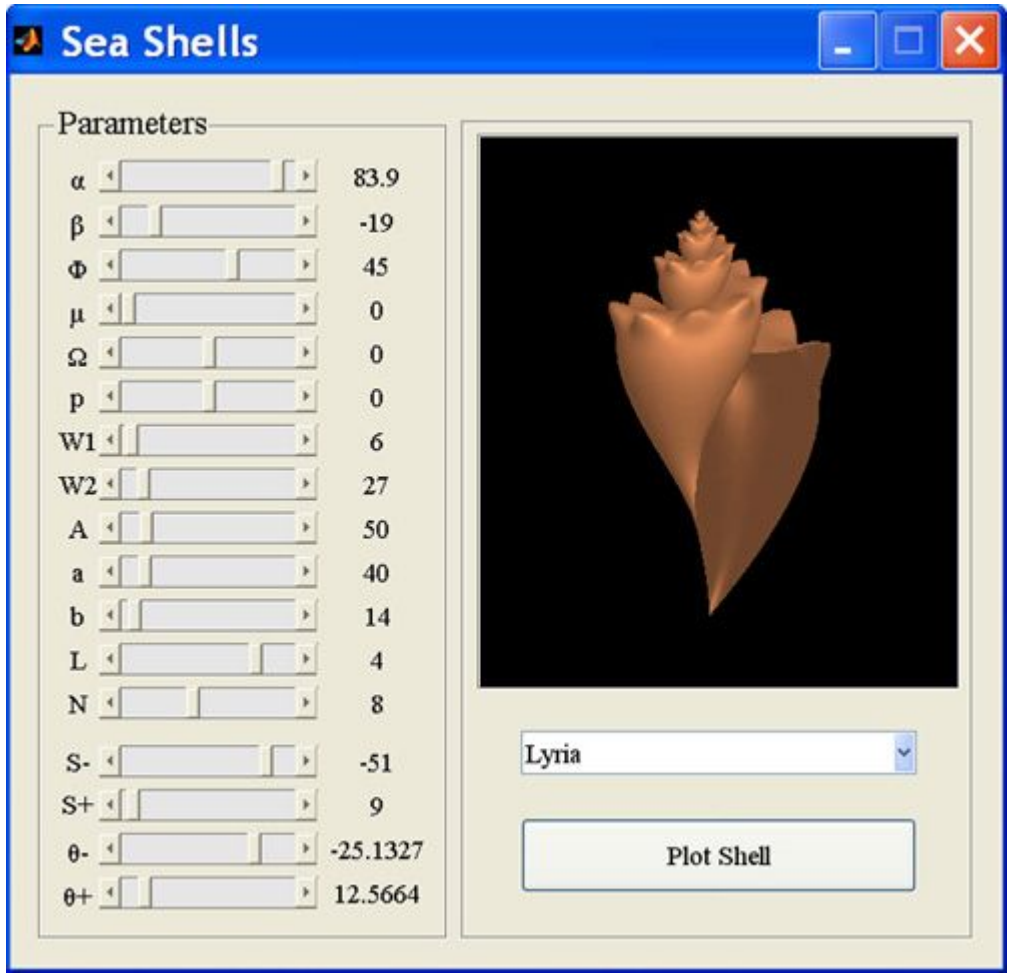
Negative Space

win.setCoords (- 1.0 , -1.0 , 3.0 , 3.0)

Interactive Graphics

event-driven programming:

The flow of the program is determined by user actions (ex. mouse clicks, keyboard entries, sensor inputs).



Mouse Click Inputs

Clicking the mouse, or pressing a key generates an **event** object.

For example: clicking a button generates a button event

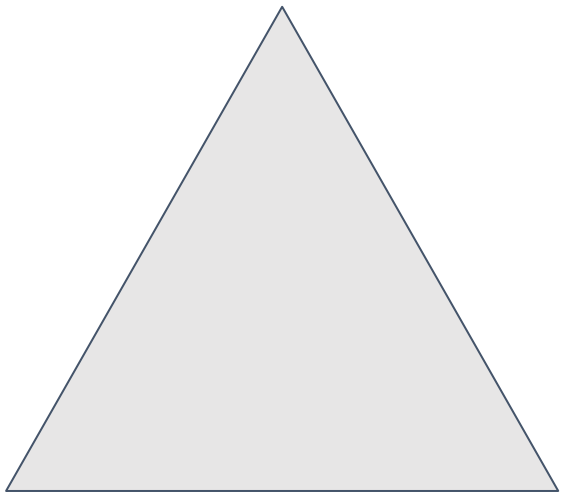
The **getMouse()** method tells the window object to wait until the user clicks on the graphics window. The spot where the user clicked on the window is returned as a Point object.

Mouse Click Example

```
win = graphics.GraphWin ("Click Me!",250, 250)
for i in range (5):
    p = win.getMouse ( )
    print (" You clicked at:", p.getX(), p.getY())
```

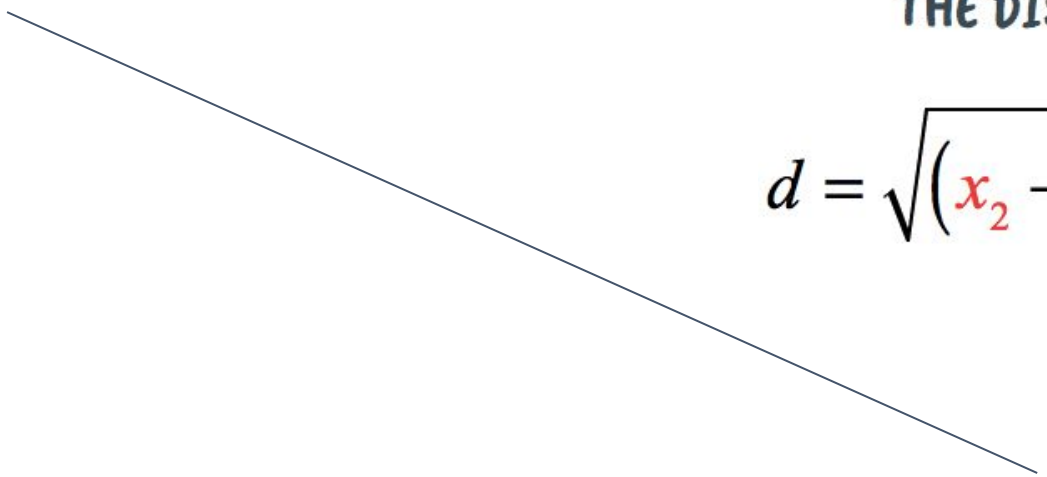
Triangle Problem

Design a program that lets the user draw a triangle by clicking on 3 points on the graphics window.



Line Problem

Design a program that lets the user draw a line by clicking on two points on the graphics window. Once the line is drawn, print the length of the line in pixels on the graphics window.

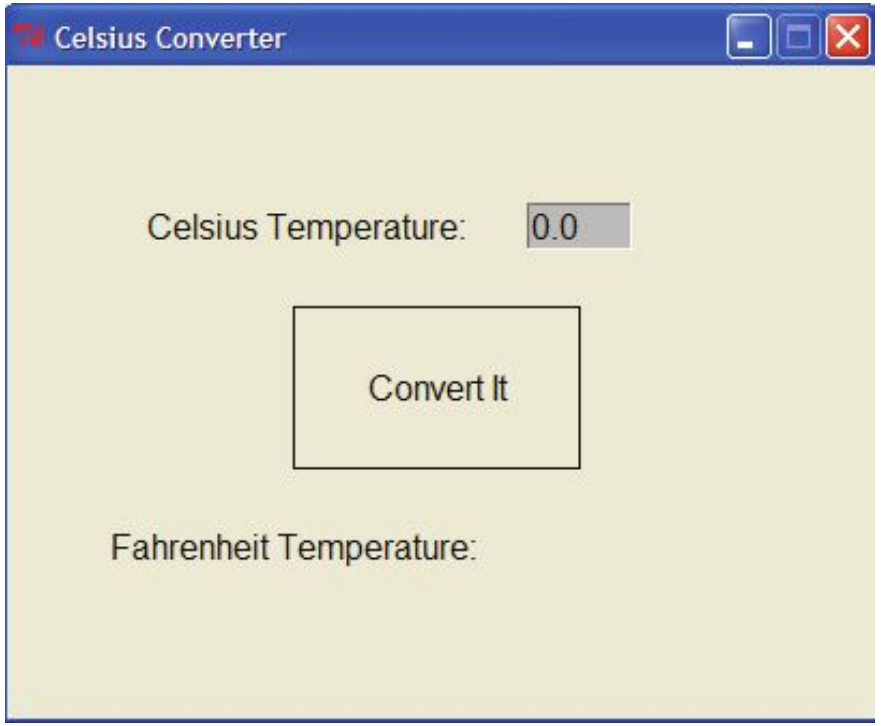


THE DISTANCE FORMULA

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

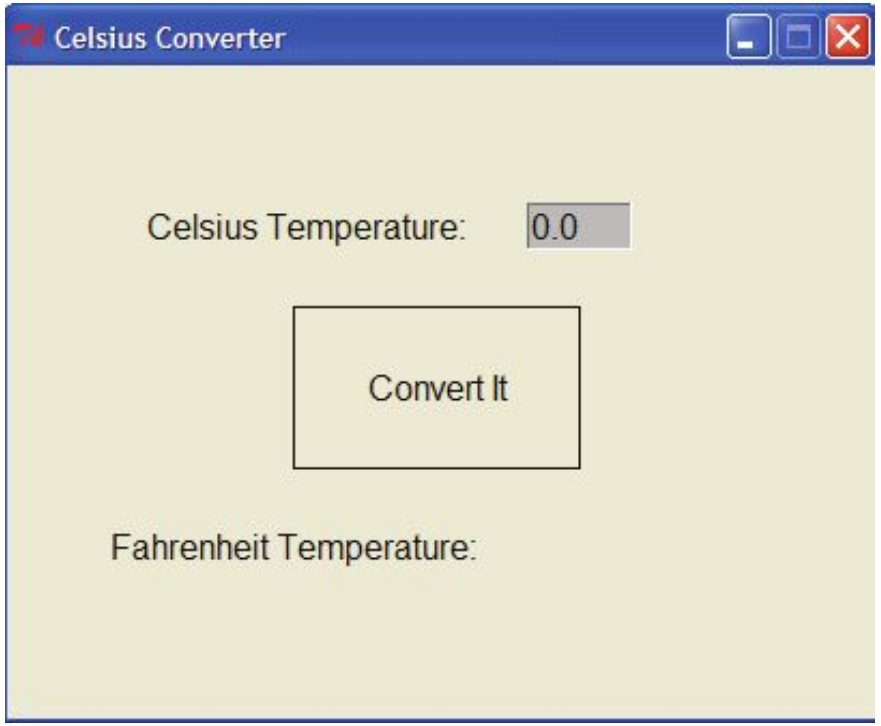
Graphical User Interface (GUI)

Developing a graphical user interface (GUI) to convert a celsius value into fahrenheit.



Draw the graphics window

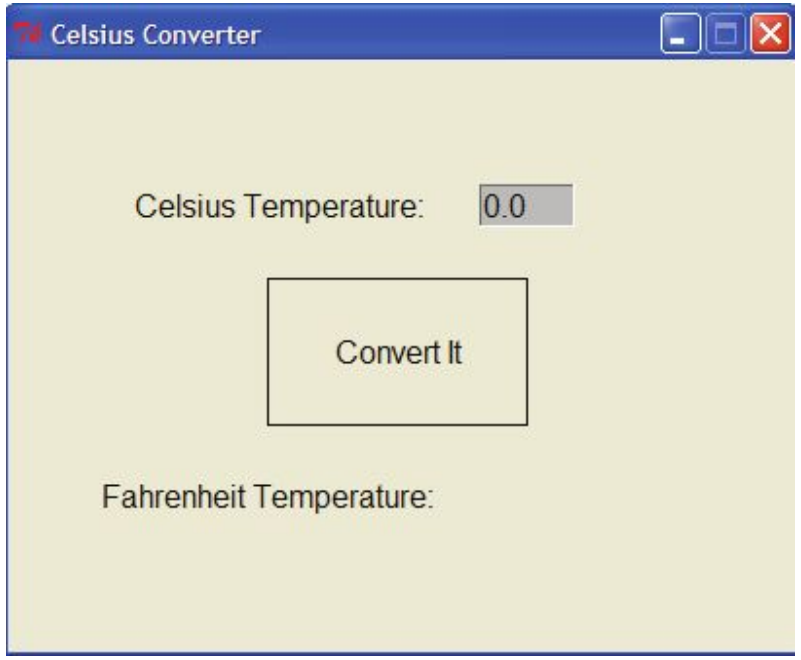
```
import graphics
win = graphics.GraphWin ("Celsius
Converter", 400,300)
win.setCoords(0.0, 0.0, 3.0, 4.0)
```



Draw the graphics objects on the window

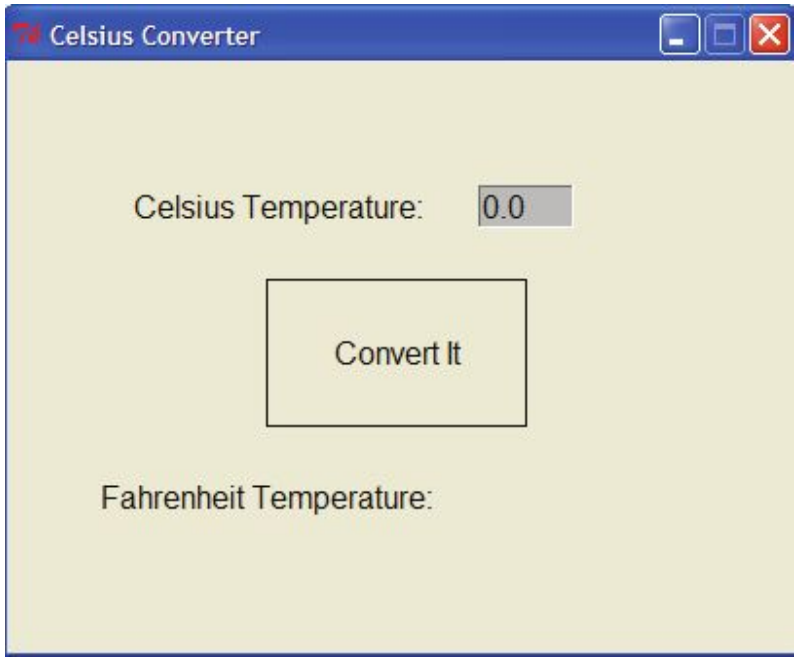
```
c_label = graphics.Text(graphics.Point(1,3),"Celsius Temperature:")
f_label = graphics.Text(graphics.Point(1,1),"Fahrenheit Temperature:")
button_label = graphics.Text(graphics.Point(1.5,2),"Convert It")
rect = graphics.Rectangle(graphics.Point(1,1.5),graphics.Point(2,2.5))
```

```
c_label.draw(win)
f_label.draw(win)
button_label.draw(win)
rect.draw(win)
```



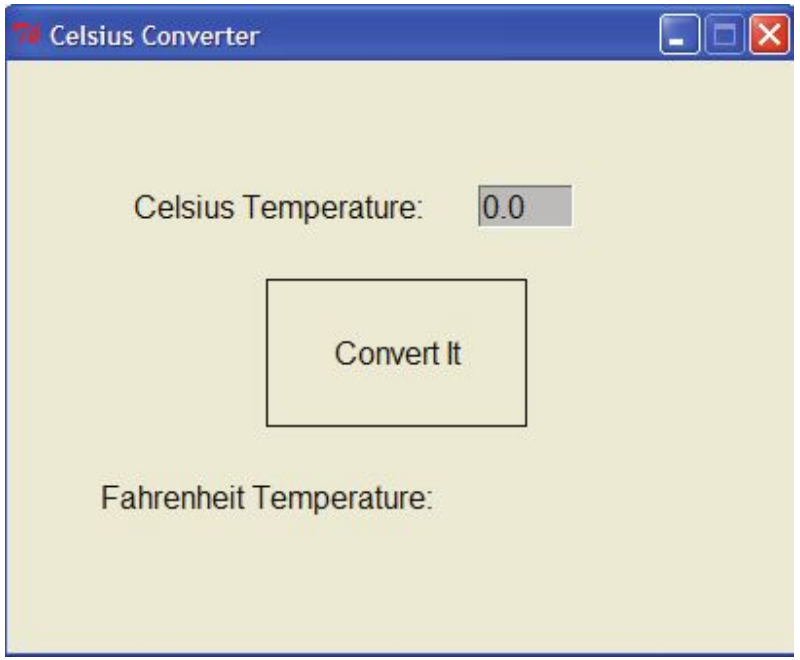
Make entry object to get text

```
inputText = graphics.Entry(graphics.Point(2,3),5)
inputText.setText("0.0")
inputText.draw(win)
```



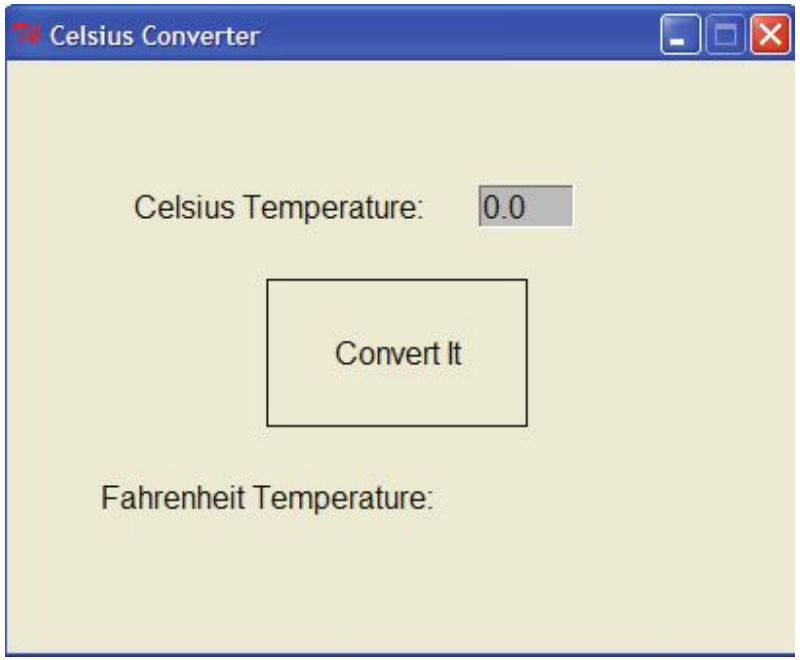
Wait for a mouse click

win.getMouse()



Get the celsius input using getText() method

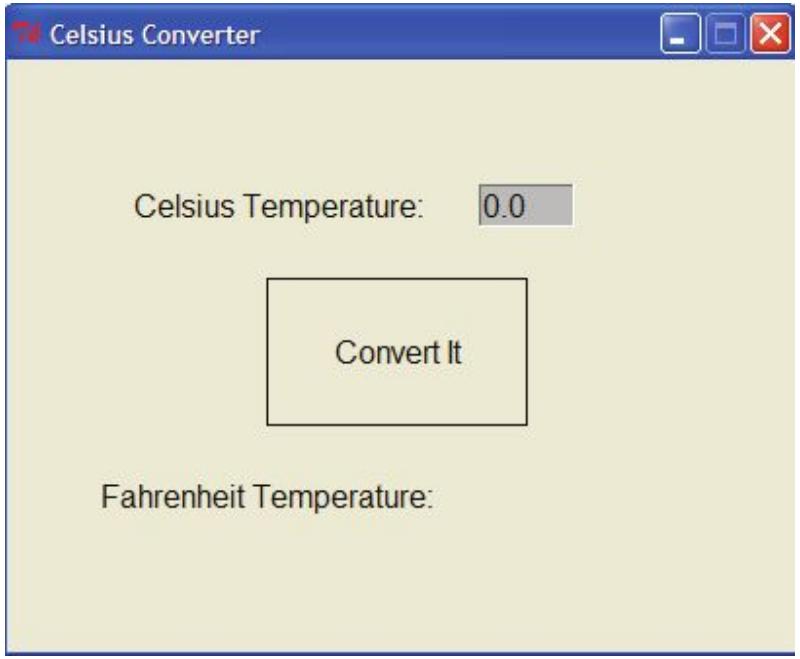
```
c_temp = float(inputText.getText())
```



Perform the conversion, display with setText() ,method

```
outputText = graphics.Text(graphics.Point (2,1),"")
outputText.draw(win)

#after win.getMouse()
f_temp = 9.0/5.0*c_temp+32
outputText.setText(round(f_temp,2))
```



```python
import turtle
turtle.setup(400,500)          # Determine the window size
wn = turtle.Screen()           # Get a reference to the window
wn.bgcolor("lightgreen")       # Set the background color
tess = turtle.Turtle()         # Create our favorite turtle

# The next four functions are our "event handlers".
def h1():
  tess.forward(30)
def h2():
  tess.left(45)
def h3():
  tess.right(45)
def h4():
   wn.bye()                    # Close down the turtle window
# These lines "wire up" keypresses to the handlers we've defined.
wn.onkey(h1, "Up")
wn.onkey(h2, "Left")
wn.onkey(h3, "Right")
wn.onkey(h4, "q")

wn.listen()
wn.mainloop()
```

Turtle library

We will talk more about animation and games once we learn more about programming!

https://www.geeksforgeeks.org/turtle-programming-python/

Archery Target Problem (try at home)

Write a program that draws an archery target. Each ring has the same width, which is the radius of the smallest circle.

Hint: objects drawn later appear on top of objects drawn earlier

Moving Ball Problem (try at home)

Develop a code that moves a purple ball to the location on the window where the mouse was clicked.