Programmer-Defined Types
These are the classes you will create.

The first class will be a class to store a point. Note how the class definition at this stage only contains a docstring.
Attributes and methods will be added later. For now, the client code will add the attributes.

Ex1: point.py

```python
# File: point.py
from math import sqrt

class Point:
    '''Stores a point as x, y

    Attributes: x: float, y: float
    '''

def point_str(p: Point) -> str:
    return '(%g, %g)' % (p.x, p.y)

def distance_between_points(p1: Point, p2: Point) -> float:
    '''Return distance between two Points.''
    return sqrt( (p1.x - p2.x) ** 2 + (p1.y - p2.y) ** 2 )

# Create two Point objects.
p1 = Point()
p2 = Point()

# Initialize each point.
p1.x = p1.y = 0
p2.x = 3.0
```
p2.y = 4.0

dist = distance_between_points(p1, p2)

# Print the distance between two points
print(f'p1: {point_str(p1)}, p2: {point_str(p2)} - distance = {dist}')

The class definition creates a class object. This class object is used to create instances, called objects.

Our next class will store a rectangle.

**Ex2: rectangle.py**

```python
# File: rectangle.py
from math import sqrt
import copy

class Point:
    '''Stores a point as x, y

    Attributes: x: float, y: float
    '''

class Rectangle:
    '''Represents a rectange.

    Attributes:
    width: float,
    height: float,
    origin: Point (lower-left corner)
    '''

def find_center(rect: Rectangle) -> Point:
    pc = Point()
    pc.x = rect.origin.x + rect.width / 2
    pc.y = rect.origin.y + rect.height / 2
    return pc

def resize_rectangle(rect: Rectangle, dw: float, dh: float):
    rect.width += dw
    rect.height += dh
```

Last updated: February 20, 2020 at 6:51 AM
def move_rectangle(rect: Rectangle, dx: float, dy: float):
    rect.origin.x += dx
    rect.origin.y += dy

def moved_rectangle(rect: Rectangle, dx: float, dy: float) -> Rectangle:
    # r = Rectangle()
    # r.origin.x = rect.x + dx
    # r.origin.y = rect.y + dy
    # r.width = rect.width
    # r.height = rect.height

    # Create a deep copy of the original object and
    # update it.
    r = copy.deepcopy(rect)
    r.origin.x += dx
    r.origin.y += dy

    return r

# Create a rectangle
box = Rectangle()
box.width = 100.0
box.height = 200.0
box.origin = Point()
box.origin.x = 0.0
box.origin.y = 0.0

print(f'original-box: origin({box.origin.x}, {box.origin.y}),
      w({box.width}), h({box.height})')

resize_rectangle(box, 100, 100)
print(f' resized-box: origin({box.origin.x}, {box.origin.y}),
       w({box.width}), h({box.height})')

move_rectangle(box, 10, 20)
print(f' moved-box: origin({box.origin.x}, {box.origin.y}),
       w({box.width}), h({box.height})')

r = moved_rectangle(box, 100, 100)
print(f'original: origin({box.origin.x}, {box.origin.y}), moved:
       origin({r.origin.x}, {r.origin.y})')
p1: circle.py