Learning Objectives
Students will be able to:

Content:
- Define objects, attributes, and classes in python
- Identify differences between attributes/variables
- Explain why classes are useful

Process:
- Write code that creates a new user-defined class with attributes
- Write code that uses user-defined types as function arguments, embedded objects, return values

Prior Knowledge
- Python concepts from Activities 1-23.

Folks, this is a brand new activity. If you encounter any issues/typos, please let Iris know!

Critical Thinking Questions:

1. Examine the following code from interactive python below using a Flower data structure.

<table>
<thead>
<tr>
<th>Interactive Python</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 &gt;&gt;&gt; iris = Flower()</td>
</tr>
<tr>
<td>4 &gt;&gt;&gt; iris.petals = 3</td>
</tr>
<tr>
<td>5 &gt;&gt;&gt; iris.petals</td>
</tr>
<tr>
<td>6 3</td>
</tr>
<tr>
<td>7 &gt;&gt;&gt; iris.color = ‘purple’</td>
</tr>
<tr>
<td>8 &gt;&gt;&gt; iris.color</td>
</tr>
<tr>
<td>10 flwrList = list()</td>
</tr>
<tr>
<td>11 flwrList = [iris]</td>
</tr>
</tbody>
</table>

   a. What type of object is flwrList? How do you know?  
      ____________________________________________________

   b. What type of object is iris? How do you know?  
      ____________________________________________________

   c. On which line do we place flwrList on the lefthand side of an assignment operator?  
      What value is assigned?  
      ____________________________________________________

   d. On which line is iris.petals on the lefthand side of an assignment operator?  
      What value is assigned?  
      ____________________________________________________

   e. What is displayed when we call iris.petals?  
      ____________________________

   f. What will be displayed when we call iris.color?  
      ____________________________

FYI: Creating a new object, such as iris or flwrList, is called instantiation and flwrList is an instance of a List class object.
2. Examine the following code below, that creates a new class in interactive python:

```python
>>> class Flower:
    ...     """ A new class representing flowers """

>>> iris = Flower()
>>> iris.petals = 3
>>> iris.sepals = 3
>>> print(iris.petals + iris.sepals)
```

a. What additional attribute are we giving to `iris` in this example?

b. What is likely to be the output after line 5?

c. What attributes does `iris` have in this example?

e. If we add `print(iris.bloomTime)` as our 7th line above, this code will generate the following error, “AttributeError: ‘Flower’ object has no attribute ‘bloomTime’” why do you think that is?

f. Write a line of python to place before `print(iris.bloomTime)` so that the AttributeError won’t occur:

3. Observe what happens when we enter the following lines, continuing from those above:

```python
>>> def countPetals(flwr):
    ...     return flwr.petals + flwr.sepals

>>> countPetals(iris)
```

a. What argument is being passed to `countPetals` on line 9? What is `countPetals`’s parameter named? arg:_________________________ param:_________________________

b. Does `iris` or `flwr` appear on the lefthand side of an assignment operator in lines 7-10?

c. Is the `iris` object modified/changed in any way in lines 7-10?

FYI: We can assign values to named elements of objects. These named elements are called attributes.

FYI: User-defined object instances can be passed to functions just like built-in object instances.
4. Examine the following code below, that creates a new class in interactive python:

```
>>> class Garden:
...    """ Represents a flower garden """

>>> myGarden = Garden()
>>> myGarden.flower = Flower()
>>> myGarden.flower.petals = 21
>>> myGarden.flower.petals
21
```

a. What type of object is myGarden? How do you know?
_______________________________________________________________________

b. What type of object is myGarden.flower? How do you know?
_______________________________________________________________________

c. What type of object is myGarden.flower.petals? How do you know?
_______________________________________________________________________

d. What is new about the assignment of a value to petals in this example?
_______________________________________________________________________

**FYI:** Embedded objects are used within other objects and can be referred to through **dot notation**.

5. The following code below continues from the previous example:

```
>>> iris.petals = 3
>>> myGarden.flower = iris
>>> myGarden.flower.petals = 6
>>> iris.petals
6
```

a. What value is assigned to iris.petals on line 18?
_______________________________________________________________________

b. What value is assigned to myGarden.flower on line 19?
_______________________________________________________________________

c. What value is assigned to myGarden.flower.petals on line 20?
_______________________________________________________________________

d. What value is stored in iris.petals, according to line 22?
_______________________________________________________________________

e. On what line might iris.petals’ value have been changed to this value?
_______________________________________________________________________

**FYI:** Objects are mutable. Their attributes can be changed inside of functions or even when embedded in other objects.
6. The following code below continues from the previous example:

```python
23 >>> def makeHybrid(flwr1, flwr2):
24     hybrid = Flower()
25     hybrid.petals = (flwr1.petals + flwr2.petals)/2
26     return hybrid
27 >>> daisy = Flower()
28 >>> daisy.petals = 21
29 >>> iraisy = makeHybrid(daisy, iris)
30 >>> iraisy.petals
31 13.5
```

a. What is the value stored in `iris.petals`? ____________________________
b. What is the value stored in `daisy.petals`? ____________________________
c. What is the value stored in `flwr1.petals` in this example? ______________
d. What is the value stored in `flwr2.petals` in this example? ______________
e. When line 25 is executed, what value is assigned to `hybrid.petals`? __________
f. What type of object is `iraisy`? How do you know?

Application Questions: Use Python to check your work

1a. Create a class, `Dog`. Create an instance of `Dog` which has a `name` and an `age` as instance attributes.

```python
def makeHybrid(flwr1, flwr2):
    hybrid = Flower()
    hybrid.petals = (flwr1.petals + flwr2.petals)/2
    return hybrid
```

1b. Write a function, `dogYears`, that takes a `Dog` object as a parameter and returns the dog’s age in dog years (multiply `age` in years by 7).
```python
def dogYears(aDog):
```

1c. Create a function, `addNickname` that accepts a `Dog` object as a parameter, and modifies that object by adding a nickname attribute to it. The nickname is ‘schmoo’ appended to the dog’s name.
```python
def makeHybrid(flwr1, flwr2):
    hybrid = Flower()
    hybrid.petals = (flwr1.petals + flwr2.petals)/2
    return hybrid
```
1d. Write a few lines of code for interactive python that uses all of the above functions you wrote:

```python
>>> 
>>> 
>>> 
>>> 
>>> 
>>> 
```