**Learning Objectives**

Students will be able to:

* **Content:***
  - Explain how *slots* differ from attributes
  - Explain why the slots attribute is useful

* **Process:**
  - Write code that creates a new user-defined class with slots

**Prior Knowledge**

- Python concepts from Activities 1-24.

---

**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.

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

2 >>> iris = Flower()
3 >>> iris.petals = 3
4 >>> iris.petals
5 3
6 >>> iris.bloomTime
7 AttributeError: 'Flower' object has no attribute 'bloomTime'
```

a. What type of object is `iris`? How do you know?

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

c. On which line is `iris.bloomTime` on the lefthand side of an assignment operator?

d. Why might `iris.bloomTime` on line 7 throw an error?

e. Write a line of python to enter before line 6, to fix the error:
2. Examine the following code below, which continues from the previous example:

```python
8 >>> daisy = Flower()
9 >>> daisy.nonsense = 'wut WUT'
10 >>> daisy.nonsense
11 'wut WUT'
```

a. What differs between our assignment of daisy in this example, and iris in the earlier example? ____________________________________________

b. Where do we assign a value to daisy.petals in this example? __________________

c. Where do we assign a value to daisy.nonsense in this example? What’s its value?
   _________________________________________________________________

d. Is nonsense a meaningful attribute for objects of type Flower?
   _________________________________________________________________

3. Examine the following code below, that overwrites previous versions of Flower:

```
Interactive Python
0 >>> class Flower:
1 ... __slots__ = ['petals']

2 >>> rose = Flower()
3 >>> rose.petals = 5
4 >>> rose.nonsense = 'May'
5 AttributeError: Flower object has no attribute 'nonsense'
```

a. How does the assignment of rose.petals differ from the assignment of iris.petals in question 1? ______________________________________________________________________

b. How does the assignment of rose.nonsense differ from the assignment of
daisy.nonsense in the previous question? ______________________________________________________________________

b. What happens with line 5 in this example that didn’t occur in the previous question? ______________________________________________________________________

c. How does the definition of the Flower class differ in this example, from the definition of Flower used in questions 1-2?  
   ______________________________________________________________________

   ______________________________________________________________________

FYI: The __slots__ keyword defines a list of attributes for a class object. No additional attributes can be added to an instance, unless their name appears in the __slots__ list.

   d. What might happen if we modify line 1 to be __slots__ = ['petals', 'nonsense'] and then ran the code?
4. Examine the following code below, which continues from the previous example:

```python
>>> violet = Flower()
>>> violet.petals = 5
>>> violet.petals
5
>>> rose.petals + violet.petals
10
```

a. What is stored in `violet.petals`?

b. What is happening on line 10?

5. Examine the following code below, which continues from the previous example:

```python
>>> def avgPetals(flwrList):
...     total = 0
...     for flwr in flwrList:
...         total += flwr.petals
...     return total / len(flwrList)
```

a. What is an example value for `flwrList`?

b. What would the output for your example value in (a) result in?

c. What does `avgPetals` do?

d. Write a function, `droughtPetals`, that accepts a `Flower` object as a parameter and an integer `days`, and removes one petal from the flower for each `days` of drought:

Application Questions: Use Python to check your work
1a. Create a class, `Student`. Create an instance of `Student` which has a `name` and a `hobby` as instance slots.

1b. Create a function, `findAllHobbies` that takes a list of `Student` objects and returns a list of every student hobby.

1c. Create a function, `flipHobbies` that takes a list of `Student` objects and swaps the hobbies of the first & last person in the list, the second and second-to-last person in the list, the third and third-to-last person in the list, etc.

1d. Write a few lines of code for interactive python that uses all of the above functions you wrote: