Name:______________________________ **Partner:** ________________________

Python Activity 10: Lists

Holding and accessing collections of objects helps code scale.

Learning Objectives
Students will be able to:
Content:

- Define a **list**
- Identify **elements** of a list
- Explain the purpose of positive and negative **index**es in a list**.**
- Explain how to access individual elements of a list as well as subsequences of the list
- Explain how to find if an item is contained within a list

Process:

- Write code that prints a list, finds the **len**gth of a list, **slice**s a list
- Write code that determines if an item is or is not contained **in** a sequence
- Write code that adds items to a list through **concatenation**

Prior Knowledge

- Variables, string literals, types, conditionals

Concept Model:

Examine the following partially completed code:

Concept Model

```python
def print_month(num_month):
    # num_month is a number between 0 & 11, representing Jan - Dec
    str_month = '??'
    # What code needs to go here?
    print("The month is", str_month)
```

CM1. If we wanted the function `print_month` to display a string representation of the numerical month stored in `num_month` (e.g., `print_month(0)` displays `January`, `print_month(3)` displays `April`), summarize what code we would have to write to make this possible, using only concepts we've already learned:

____

____

CM2. Will this approach *scale* for larger problems (say, if we wanted a similar mapping between the numerical year 1999 and the string representation, `nineteen ninety-nine`, and *all* other years up to now)?

____

____

Critical Thinking Questions:

FYI: A ***sequence*** is an object that stores multiple data items in a contiguous/ordered manner. Two types of sequences are **strings** and **lists.** Each value stored in a list is called an **element.**

1. Examine the sample lists below.

Sample Lists in Python

```
digits = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
 fruits = ["apple", "banana", "cantelope", "pear", "orange"]
 studentData = ["Jones", 10234, 3.5, "Brown", 23145, 2.8]
```

a. How many **elements** does the list named **digits** contain? ______________________

b. What type of data is stored in each list (String, numeric)?

- **digits** list:

  ____

- **fruits** list:

  ____

- **studentData** list:

  ____

c. How would you define a **list?**

____

____

d. Why might a **list** be useful?

____

____

2. The second line of code in the following program prints the first **element** in the **fruits** list (i.e., 'apple').

```
fruits = ["apple", "banana", "cantelope", "pear", "orange"]
 print(fruits[0])
```

a. What value in the list does **fruits[3]** represent? ______________________________

b. Write a line of code that prints the last element.

____

c. `fruits[-1]` points to `'orange'`. What might `fruits[-2]` point to?

____

FYI: The number used to locate an element in a sequence, including lists and strings, is called an **index**.

e. Explain how the positive and negative indexes locate specific elements.

____

____

f. `print(fruits)` produces the following output, is this what you expected? Why/not? `['apple', 'banana', 'cantelope', 'pear', 'orange']`

____

____

3. Examine the following lines of code:

```
fruits = ["apple", "banana", "cantelope", "pear", "orange"]
 legumes = ["beans", "peas"]
 vegs = ["asparagus", "broccoli", "carrot"]
```

a. How many elements are in each of the above Lists?

fruits: ____________ legumes: ___________ vegs:_____________

b. The command `len(fruits)` returns `5`. What might be the output for the following statements: `len(legumes):` ________ `len(vegs):`________

c. What might the built-in function **len()** do when its argument is a List?

____

d. What is the [positive] index of the last item in each of the above Lists?

fruits: ____________ legumes: ___________ vegs:_____________

e. What is the relationship between a List's last index and the number of elements in the List?

____

4. Examine the following lines of code:

```
fruits = ["apple", "banana", "cantelope", "pear", "orange"]
some1 = fruits[2:]
some2 = fruits[2:4]
others = fruits[::2]
```

a. What is at index 2 of `fruits`? ___________Index 4? ___________

b. After running this code, `some1` is assigned the value `['cantelope', 'pear', 'orange']`, why might that be?

____

c. After running this code, `some2` is assigned the value `['cantelope', 'pear']`, why might that be?

____

c. What does the **[:]** slicing operator do to sequences, such as lists?

____

d. At the end of this code `others` contains `['apple', 'cantelope', 'orange']`, what might the **[::2]** slicing operator do to sequences, such as lists?

____

f. What might `fruits[::3]` return? ____________________

FYI: The **Slicing Operator** allows you to access parts of sequences such as strings and lists. You can select multiple elements of a sequence.
Syntax: <sequenceName>[startInclusive : endExclusive : stepIncrement].

5. Examine the following lines of code:

```
fruits = ["apple", "banana", "cantelope", "pear", "orange"]
mystery1 = fruits[:] # contains ['apple', 'banana', 'cantelope', 'pear', 'orange']
mystery2 = fruits[::-1] # contains ['orange', 'pear', 'cantelope', 'banana', 'apple']
```

a. In the command `fruits[:]` above what is the start index and end index? ___:____

b. What is another way to describe what the `fruits[:]` command is doing?

____

c. In the command `fruits[::-1]` above what is the start index and end index?

___::____

d. What is another way to describe what the `fruits[::-1]` command is doing?

____

6. Examine the following lines of code:

```
0 fruits = ["apple", "banana", "cantelope", "pear", "orange"]
1
2 print("pear" in fruits)
3 print("guava" in fruits)
```

a. What might be displayed at line 2? ______________________

b. What *type* of value does `"pear" in fruits` return? ____________

c. What might be displayed at line 3? ______________________

d. What might the **`in`** operator do when used on a List?

____

e. If we added a 4th line, `print("guava" **not in** fruits)`, what would you expect would be displayed? ______________________

f. What might the **not in** operator do when used on a List?

____

6. Examine the following program and its output:

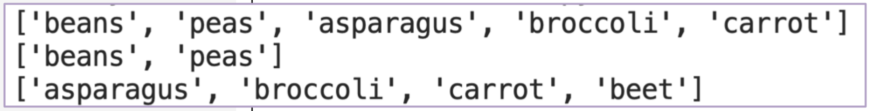
Program :

```
legumes = ["beans", "peas"]
vegs = ["asparagus", "broccoli", "carrot"]

combine = legumes + vegs
print(combine)
print(legumes)

vegs = vegs + ["beet"]
print(vegs)
```

Output:



```
['beans', 'peas', 'asparagus', 'broccoli', 'carrot']
['beans', 'peas']
['asparagus', 'broccoli', 'carrot', 'beet']
```

a. Draw lines between the `print()` statements in the Program and their associated output.

b. What is stored in `combine`?

c. At the end of the code, what is stored in `legumes`? How has it changed from the beginning?

d. At the end of the code, what is stored in `vegs`? How has it changed from the beginning?

e. Why do we have to write `vegs = vegs + ["beet"]` rather than just `vegs + ["beet"]` to update the value stored in `vegs`?

f. Write a *single* line of code that adds the strings "lentil" and "chickpea" to `legumes`.

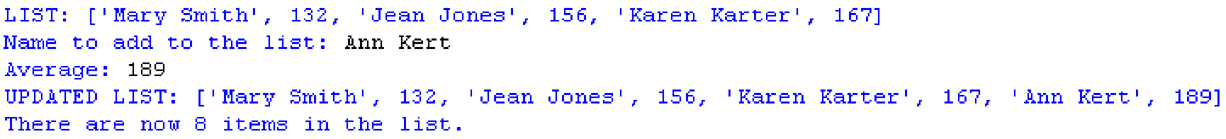
FYI: The **Concatenation Operator +** allows you to append one sequence, such as Lists or strings, to the end of another sequence of the same type. It returns the *new*, appended sequence.

7. Draw lines between the left column and the right, matching the sequence operations we have learned that work for **lists**, to the result of those operations:

Operation	**Result**
`seq[i]`	True if `x` is contained within `seq`
`seq[startIncl : endExcl]`	slice of `seq`: `startIncl` to `endExcl` with step `step`
`seq[startIncl : endExcl : step]`	the i'th item of `seq`, when starting with 0
`len(seq)`	slice of `seq` from `startIncl` to `endExcl`
`seq1 + seq2`	False if `x` is contained within `seq`
`x in seq`	length of `seq`
`x not in seq`	The concatenation of `seq1` and `seq2`

Application Questions: Use the Python Interpreter to check your work

1. Create a program that prints a given list, prompts the user for a name and average, adds the new information to the list and prints the new list. It should produce output similar to the following:



```
LIST: ['Mary Smith', 132, 'Jean Jones', 156, 'Karen Karter', 167]
Name to add to the list: Ann Kert
Average: 189
UPDATED LIST: ['Mary Smith', 132, 'Jean Jones', 156, 'Karen Karter', 167, 'Ann Kert', 189]
There are now 8 items in the list.
```

2. Revise the previous program so that it allows the user to enter the name of a person and an average, but only if that person does not already exist in the list.