Learning Objectives
Students will be able to:

Content:
- Define a list comprehension
- Describe the key pieces of constructing a list comprehension

Process:
- Write code to construct lists using list comprehensions.
- Convert multi-line list construction loops into one-line list comprehensions.

Prior Knowledge
- Python concepts from Activities 1-20.

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

Critical Thinking Questions:

1. Examine the sample code that converts a list of US Dollar amounts to British pound.

<table>
<thead>
<tr>
<th>Sample Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 monies = [1.22, 5.50, 3]</td>
</tr>
<tr>
<td>1 gbp = []</td>
</tr>
<tr>
<td>2 for usd in monies:</td>
</tr>
<tr>
<td>3 gbp.append(usd*0.77)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Critical Thinking Questions</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. What is the purpose of line 1?</td>
</tr>
<tr>
<td>b. What line of code iterates through each element of the monies list?</td>
</tr>
<tr>
<td>c. What part of the code converts the values of monies from USD to GBP?</td>
</tr>
<tr>
<td>d. What line adds these new elements to gbp?</td>
</tr>
<tr>
<td>e. What are the elements of the list, gbp, at the end of this code?</td>
</tr>
</tbody>
</table>

2. The following code below results in identical outcomes as the above Sample Code:

<table>
<thead>
<tr>
<th>Sample Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 monies = [1.22, 5.50, 3]</td>
</tr>
<tr>
<td>1 gbp = [usd*0.77 for usd in monies]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Critical Thinking Questions</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. What part of code initializes the list gbp?</td>
</tr>
<tr>
<td>b. What part of the code iterates through each element of the monies list?</td>
</tr>
<tr>
<td>c. What part of the code converts the values of monies from USD to GBP?</td>
</tr>
</tbody>
</table>
3. Examine the sample code below which also uses a list comprehension:

<table>
<thead>
<tr>
<th>Sample Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 # Assume each element of the list words is a line from</td>
</tr>
<tr>
<td>0 # /usr/share/dict/words (the unix dictionary)</td>
</tr>
<tr>
<td>1 longer = [ wd for wd in words if len(wd) &gt; 5 ]</td>
</tr>
</tbody>
</table>

a. What differs in this list comprehension that we did not have in the previous USD/GBP example?

b. What does the variable wd represent in this code?

c. What does the code if len(wd) > 5 do?

d. Why is this line of code enclosed in square brackets?

e. When this code completes execution, describe what is stored in the longer variable:

f. Write code to create a list that contains only words that begin with the letter ‘w’. Use a list comprehension:

FYI: You can imagine visually breaking down the syntax of a list comprehension as follows:

\[
\text{resultList} = [ \text{<transform>} \text{<iteration>} \text{<boolean conditional>} ]
\]

The Boolean conditional works as a filter and may be omitted. Likewise, the transformation may not actually change the value.
4. Examine the following code:

```python
0 testStr = "Hello 12345 World"
1 newList = []
2 for x in testStr:
3     if x.isdigit():
4         newList.append(x)
```

a. What does the code on line 3 do?

b. What will `newList` contain when this code completes execution?

c. Construct a list comprehension that accomplishes the same tasks as this example code:

5. Examine the following code from an interactive Python session:

```python
0 >>> def hasSub(word, substring):
1     ...    return substring in word
2 >>> names = ['pixel','tally','wally','linus','annie']
3 >>> similar = [ dog for dog in names if hasSub(dog,'lly') ]
4 >>> similar
5 ['tally', 'wally']
```

a. If we call `hasSub(dog, 'lly')`, what does the function return?

b. Construct a list comprehension that accomplishes the same tasks as this example code, but without the function `hasSub(..):

6. Examine the following list comprehension:

```python
split6 = [ x+y for x in wds for y in wds if x+y in words ]
```

a. Rewrite the above list comprehension as a multi-line statement:

b. What does this list comprehension do?
Application Questions: Use the Python Interpreter to check your work

1. Write a list comprehension to make a copy of the list, myList:

2. Write a list comprehension to create a list of all numbers between 0 and 10 (Hint: range(..)):

3. Write a function that capitalizes a list of strings into a new list, using list comprehensions. Return the new list. Do not modify the given list!
   
   ```python
def capitalize(stringList):
    # Your code here
```

4. Write a list comprehension to generate a list, words, where each element is a line from a file, /usr/share/dict/words, stripped of leading and trailing whitespaces:

   ```python
   words = # Your code here
   ```

5. Write a function that returns a list containing the values of numList squared. Use a list comprehension. Do not modify the given list, numList!
   
   ```python
def squared(numList):
    # Your code here
```

6. Using a list comprehension, write a function that returns a list containing the values of numList squared, but only of the prime numbers in numList. You can use the function isPrime(..) to determine if a given number is prime. Return the new list. Do not modify the given list!

   ```python
def squarePrimes(numList):
    # Your code here
```

   ```python
def isPrime(num):
    # returns True if num is a prime number, False if it isn’t.
```

7. Review lab assignments and Homeworks for more applications of list comprehensions.