Name:_____ **Partner:** _____

Python Activity 22: List Comprehensions

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.

Sample Code

```
0 monies = [1.22, 5.50, 3]
1 gbp = []
2 for usd in monies:
3    gbp.append(usd*0.77)
```

   a. What is the purpose of line 1?

   _____

   b. What line of code iterates through each element of the `monies` list?

   _____

   c. What part of the code convert the values of `monies` from USD to GBP?

   _____

   d. What line adds these new elements to `gbp`? _____

   e. What are the elements of the list, `gbp`, at the end of this code?

   _____

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

```
0 monies = [1.22, 5.50, 3]
1 gbp = [usd*0.77 for usd in monies]
```

   a. What part of code initializes the list `gbp`?

   _____

   b. What part of the code iterates through each element of the `monies` list?

   _____

   c. What part of the code converts the values of `monies` from USD to GBP?

   _____

d. What part of the code adds these converted values to `gbp`?

_____

e. What are the elements of the list, `gbp`, at the end of this code?

_____

f. How do we know that `gbp` is a list? (What punctuation typically indicates lists?)

_____

FYI: ***List Comprehensions*** provide a concise way to create lists.

3. Examine the sample code below which also uses a list comprehension:

Sample Code

```
0 # Assume each element of the list words is a line from
0 # /usr/share/dict/words (the unix dictionary)
1 longer = [ wd for wd in words if len(wd) > 5 ]
```

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:

resultList = [<transform> <iteration> <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:

```
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:

```
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. What might `substring in word`, do?

_____

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

_____

_____

6. Examine the following list comprehension:

```
combined = [ 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!

```
def capitalize(stringList):
```

_____

_____

_____

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:

`words =` _____

_____

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`!

```
def squared(numList):
```

_____

_____

_____

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!

```
def squarePrimes(numList):
```

_____

_____

_____

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