Name:_____ **Partner:** _____

Python Activity 34: List & String Methods

Built-in functions work for many kinds of objects, but what about when we need more specific functions?

Learning Objectives

Students will be able to:

Content:

- Use string methods on strings and list methods on lists with ***dot notation***
- Define methods, compare/contrast to functions
- Find documentation on string and list methods

Process:

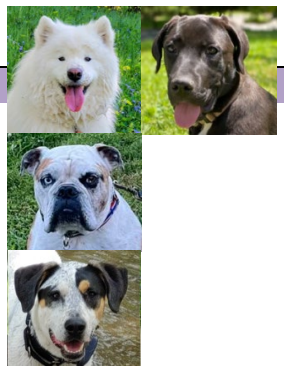
- Write code that uses list methods to modify lists
- Write code that uses string methods to manipulate string values

Prior Knowledge

- Python concepts: lists, strings, functions

Critical Thinking Questions:

1. Observe the following session in interactive python below:



Interactive Python

```
0 >>> dogs = ['pixel', 'chelsea']
1 >>> dogs += ['linus']
2 >>> dogs
3 ['pixel', 'chelsea', 'linus']
4 >>> dogs.append('artie')
5 >>> dogs
6 ['pixel', 'chelsea', 'linus', 'artie']
```

a. Circle the *syntax* that is new to us.

b. What does line 1 do? _____

c. What *object* does the += operator append the string to? _____

d. What does line 4 appear to do? _____

e. Does the `.append()` method appear to have different behavior than the += append operator?

f. What *object* does the append *method* append the string to? _____

FYI: ***Methods*** are functions that work on a specific object and are accessed using ***dot notation***.

2. Lists have other methods that work on them using dot notation. Observe the following lines of code below their questions:

a. What might the `.append(..)` method do? _____

```
>>> word_lst = ["Computer"]
>>> word_lst.append("Science")
>>> word_lst
['Computer', 'Science']
```

b. What might the `.extend(..)` method do? _____

```
>>> course_lst = ["Computer", "Science "]
>>> course_lst.extend([1, 3, 4])
>>> course_lst
['Computer', 'Science', 1, 3, 4]
```

c. Why might the value stored in `course_lst` change?

_____

h. What might the `.index(..)` method do? _____

```
>>> course_lst = ["Computer", "Science", 134]
>>> course_lst.index(134)
2
>>> course_lst.index(136)
ValueError: 136 is not in list
```

i. What might the `.count(..)` method do? _____

```
>>> course_lst = ["Computer", "computer", "Computer "]
>>> course_lst.count("Computer")
1
```

3. Strings also have methods accessible through dot notation! You can see them all with `pydoc3 str` from the terminal (or for lists, with the command `pydoc3 list`).

a. Draw lines between the left column and the right, matching the code on the left with what you expect the output/result to be on the right:

Code	**Result**
`"CsCI134".lower()`	`0`
`"CsCI134"[0].isupper()`	`'C s CI 134'`
`' '.join(['C','s',"CI","134"])`	`'CSCI 134'`
`"Cs CI 134".split()`	`False`
`"Cs, CI,134".split(',')`	`4`
`"  CSCI 134   ".strip()`	`2`
`"CSCI 134".find(' ')`	`True`
`'\n'.isspace()`	`"['Cs', ' CI', '134']"`
`'134'.isalpha()`	`"['Cs', 'CI', '134']"`
`"CSCI134".count('C')`	`True`
`"CSCI134".index('C')`	`'CSCI136'`
`"CSCI134".replace('4', '6')`	`'csci134'`

c. If you had to guess, what do you think each of these string methods do?

Method / Function	What the parameter represents	What it does
`.lower()`		
`.isupper()`		
`.join(something)`		
`.split(something)`		
`.strip()`		
`.find(something)`		
`.isspace(something)`		
`.isalpha(something)`		
`.count(something)`		
`.index(something)`		
`.replace(some1, some2)`		

d . If we wanted to write code that takes a string, `dog_name`, assigns it the value "`pixel`" and then uses string methods to change the character "`x`" to "`zz`", what would that line of code be:

_____

e. What is stored in `dog_name` after these methods are executed?

_____

FYI: There are no string methods that *change* a string! This is because strings are **immutable**, or unchangeable. String methods return a *new* string, and that new string can be re-assigned to the original string's variable.

Application Questions: Use the Python Interpreter to check your work

TBD.