

Name: \_\_\_\_\_

Partner: \_\_\_\_\_

## Python Activity 11: Looping Structures – FOR Loops

*How do we look at each element in a sequence?*

### Learning Objectives

Students will be able to:

*Content:*

- Motivate the use of loops
- Explain the syntax of a **for-each loop**
- Explain the use of **accumulation variables**.

*Process:*

- **Trace** through for..loops, identifying how local variables are modified
- Write code that includes a **for-each loop**.
- Write code that uses **accumulation variables**.

### Prior Knowledge

- Strings, indexing, conditionals, input, expressions

### Conceptual Model

Closely examine the Python program below that uses only concepts we've learned so far.

#### Python Program

```
word = "Williams"
print("Word is: " + word)
item = input("What letter to count? ")
count = 0
if word[0] == item:
    count = count + 1
if word[1] == item:
    count = count + 1
if word[2] == item:
    count = count + 1
if word[3] == item:
    count = count + 1
if word[4] == item:
    count = count + 1
if word[5] == item:
    count = count + 1
if word[6] == item:
    count = count + 1
if word[7] == item:
    count = count + 1
print("Occurrences: " + str(count))
```

CM1. When this code runs, and the user enters `i`, what might the output of the program be?

CM2. What would we have to do if we wanted to accomplish the same task for a longer string, like `'Williams College 1793'`? \_\_\_\_\_



CM3. Approximately how many lines of code would your solution require? \_\_\_\_\_

## Critical Thinking Questions

1. Closely examine the Python program below.

```


Python Program





```
word = "Williams"
print("Word is: " + word)
item = input("What letter to count? ")

count = 0
for char in word:
    if char == item: # is our current element item?
        count = count + 1


print("Occurrences: " + str(count))
```


```

- a.  Circle the new keyword we haven't seen before. How many lines of code is this example?
- b.  The output from this program will be identical to that of the previous example, except it will work for a string of *any length* as well. Why might that be?

---


---

- c.  What might the `for` keyword do?


---

---

**FYI:** A **looping structure** allows a block of code to be repeated one or more times. A **for** loop is one of the two looping structures available in Python. This particular example of `for..loop` is known as a **for-each loop**, as it **iterates** (or loops over) each of the items in a sequence (e.g., each character in a string).

- d.  How does the Python interpreter know what lines of code belong to the loop body? (or, why doesn't Python print "Occurrences" several times?)

---

- e.  **Every `for..loop` structure requires certain syntax.** Identify the part of code in the Python program that corresponds to each of these syntax requirements.

- A keyword that indicates the beginning of the `for..loop`: \_\_\_\_\_

- A **loop variable** that represents each element in the sequence as python loops: \_\_\_\_\_

---

- A sequence to loop over : \_\_\_\_\_

---

- An operator that indicates the test condition between the loop variable and the sequence: \_\_\_\_\_

---

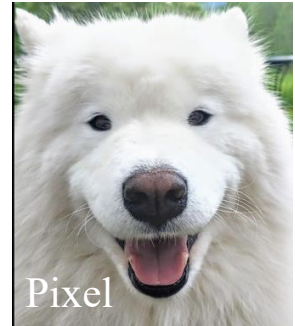
- An operator that indicates the end of the `for..loop` definition: \_\_\_\_\_

**FYI:** An **accumulation variable** "accumulates" values in a loop structure.

**KEY** f. What is the accumulation variable in the above example? \_\_\_\_\_

2. Examine the following interactive python interaction:

```
dog_name = "pixel"  
for ch in dog_name:  
    print(ch)
```



a. What kind of loop is this? \_\_\_\_\_  
b. Identify the part of code in the Python program that corresponds to each of the for..loop syntax requirements.

- A keyword that indicates the beginning of the for..loop: \_\_\_\_\_
- A **loop variable** that represents each element in the sequence as python loops:  
\_\_\_\_\_
- A sequence to loop over :  
\_\_\_\_\_
- An operator that indicates the test condition between the loop variable and the sequence:  
\_\_\_\_\_

c. Is there an accumulation variable in the above example? \_\_\_\_\_

**KEY** d. What does the loop variable `ch` represent?  
\_\_\_\_\_

Where is `ch` assigned its value(s)?  
\_\_\_\_\_

3. Examine the following code:

```
Python Program  
items = "cookie"  
count = 0  
for it in items:  
    count = count + 1  
    print(str(count) + it + " AH AH AH")
```

**KEY** a. What is the output of this program? Trace through the values as they change:

Before the loop:	items	count	it	displayed/print()
Iteration 1:	_____	_____	_____	_____
Iteration 2:	_____	_____	_____	_____
Iteration 3:	_____	_____	_____	_____
Iteration 4:	_____	_____	_____	_____
Iteration 5:	_____	_____	_____	_____
Iteration 6:	_____	_____	_____	_____

