Name: ______________________________ **Partner:** ________________________

Python Activity 16: Looping Structures – *WHILE* Loops

We often don't know how many times we want code repeated ahead of time!

Learning Objectives

Students will be able to:

Content:

- Explain the syntax of a **while loop**
- Explain the three parts of **while loop** logic
- Differentiate **sentinel-controlled** and **count-controlled** loops

Process:

- Write code that includes **sentinel-controlled** and **count-controlled** loops

Prior Knowledge

- Python concepts: for..loops, range, input, int(), conditionals, booleans, functions

Concept Model

Suppose we wanted the following behavior in interactive Python (the numbers are user input):

Don't spend more than ~2 minutes on this question!

```
>>> from pogil16_while import conceptmodel
>>> conceptmodel()
Enter a positive number to sum: 10
Enter a positive number to sum: 20
Enter a positive number to sum: 30
Enter a positive number to sum: -5
The sum is: 60
```

CM1. Summarize what this code might be doing:

____

____

CM2. Using *only concepts we've learned so far*, summarize how we might try to implement this behavior:

____

____

CM3. What might be some of the problems with this solution?

____

____

Critical Thinking Questions

1. Closely examine the Python programs below.

FOR LOOP -- Python Program

```
name = input("Your name? ")
times2print = int(input("Num times to print: "))
for i in range(times2print):
    print(name)
```

WHILE LOOP -- Python Program

```
name = input("Your name? ")
times2print = int(input("Num times to print: "))
j = 0
while j < times2print:
    print(name)
    j = j + 1
```

a. **Every for..loop structure requires certain syntax.** Identify the part of code in the Python program that corresponds to each of these syntax requirements for the FOR LOOP.

- *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:*

  ____
- *A colon that indicates the end of the for..loop definition:* ________________________

b. What does the FOR LOOP do? ____

c. This example for loop and WHILE LOOP produce the exact same output. Can you identify the following in the WHILE loop code:

- *Initialize a variable used in the test condition:*

  ____
- *A keyword that indicates the beginning of the while..loop:* ___________________________
- *Include a test condition that causes the loop to end when the condition is false:*

  ____

  *What **type** of value is this test condition?* ____
- *A colon that indicates the end of the while..loop definition:* ________________________
- *Within the loop body, update the variable used in the test condition:*

  ____

2. a. The following code *should* print the numbers beginning with 100 and ending with 0. However it is missing a line of code. Add the missing code (draw an arrow to indicate where the code belongs).

```
countdown = 100
while countdown > 0:
    print(countdown)
print("Done!")
```

b. The following code *should* print the numbers from 1 to 10, but it does not print anything. Correct the problem.

```
number = 12
while number <= 10:
  print(number)
  number = number + 1
```

c. Examine the following code, what might the output be? Will this program end? Why or why not? ____

```
number = 0
while number <= 10:
  print(number)
  number = number - 1
```

🔑 d. Summarize additional things to be aware of when writing a while..loop as suggested in these examples: ____

____

____

3. Examine the following code: And its output:

```
number = 1
while number <= 10:
      if number % 2 == 0:
            print(number)
      number = number + 1
```

```
2
4
6
8
10
```

a. What is the pattern in the output?

____

b. Why isn't this code printing every number between 1 and 10?

____

🔑 c. What does this code tell us about what actions we can take *inside the loop body*?

____

🔑 4. Sometimes a programmer does not know how many times data is to be entered. For example, suppose you want to create a program that adds an unknown amount of positive numbers that will be entered by the user. The program stops adding numbers when the user enters a negative number. Then the program prints the total. Before creating this program, review the three actions required for all loops:

a. *Initialize a variable that will be used in the test condition:* What will be tested to determine if the loop is executed? Write a line of code that initializes a variable to be used in the test condition of the loop for this program. The variable should contain a value entered by the user.

____

For this particular problem, we also need to initialize an *accumulator* variable. What should this accumulator variable do? Write a line of code to initialize it:

____

b. *Include a test condition that causes the loop to end when the condition is false:* What is the test condition for the while loop used in this program?

____

c. *Within the loop body, update the variable used in the test condition:* Write the code for the loop body. Include the code to update the variable in the test condition.

____

____

____

____

For this particular problem, we also need to update our *accumulator* variable. Write a line of code to update it:

____

d. Is this a *count-controlled* loop? Why or why not?

____

e. Complete the program. Enter and execute the code (at home). Does it work properly?

5. Read-through the following code:

```
do_again = 'y'
while do_again == 'y':
     word = input("Enter a word:")
     print("First letter of " + word  + " is " + word[0])
     do_again = input("Type 'y' to enter another word and anything
                             else to quit.")
print("Done!")
```

a. What does the program do? ____________________________________

____

b. What is the variable name used to store the user's input? ____________________

c. What is the variable name used in the test condition? ____________________

d. When does the program end? ____________________________________

FYI: A **sentinel-controlled loop** repeats the loop body until the user enters a pre-specified value. In Python, while..loops are typically the best choice for sentinel-controlled loops, and for..loops best for **count-controlled loops**.

e. Why is the loop in this program an example of a **sentinel-controlled** loop?

____

6. Read-through the following code:

```
def compute_sum():
    sum = 0
    is_pos = True
    while is_pos:
        prompt = "Please enter a positive number: "
        num = int(input(prompt))
        if num < 0:
            is_pos = False
        else:
            sum = sum + num
    return sum
```

a. Where does this code:

- *Initialize a variable used in the test condition:* ______________________
- *Include a test condition that causes the loop to end when the condition is false:*

  ____

  How is this test condition different from the ones we saw earlier?

  ____

- *Within the loop body, update the variable used in the test condition:*

  ____

b. When does the program end? ______________________

c. Why is the loop in this program an example of a **sentinel-controlled** loop?

____

FYI: An **infinite loop** is created when a test condition for looping never becomes False. There are certain cases where an infinite loop is desirable, but we still need to exit the loop eventually. In these cases, the three parts of a loop can be found in *different locations*.

Application Questions: Use the Python Interpreter to check your work

1. Write a code segment that, given a number, prints all the "halves" (i.e., divide by two) of that number, until it reaches 0.

   ____

   ____

   ____

2. Write a code segment that prompts the user for an even number. As long as the number is not even, the user should be given a message and prompted again for an even number.

   ____

   ____

   ____

3. Write code segment that prompts the user for a letter from ‘a-z’. As long as the character is not between ‘a-z’, the user should be given a message and prompted again for a letter between ‘a-z’.

   ____

   ____

   ____

   ____

4. The following directions will create a program that prompts the user to enter a number between 1 and 10. As long as the number is out of range the program re-prompts the user for a valid number. Complete the following steps to write this code.

   a. Write a line of code that prompts the user for a number between 1 and 10.

      ____

   b. Write a **Boolean expression** that tests the number the user entered by the code in step (a) to determine if it is **not** in range.

      ____

   c. Use the Boolean expression created in step (b) to write a **while loop** that executes when the user input is out of range. The body of the loop should tell the user that they entered an invalid number and prompt them for a valid number again.

      ____

      ____

      ____

      ____

   d. Write the code that prints a message telling the user that they entered a valid number.

      ____

   e. Put the segments of code from steps (a)-(d) together. Enter and execute the code (at home). Does it work properly? If not, correct it and test it again.

   f. How many times does the loop execute? ____________________________