Name:_____   **Partner:** _____

Python Activity 4: None-Returning Functions

Functions are great for code you want to reuse several times – no need to copy & paste!

Learning Objectives

Students will be able to:

Content:

- Explain the meaning and purpose of a function
- Recognize a function definition, function header, and function call in a program
- Explain programs that use the same function multiple times
- Use good test data for programs that include functions

Process:

- Write code that includes function definitions and function calls

Prior Knowledge

- Print, strings, expressions

Critical Thinking Questions:

Python Program

```python
# Description: This program uses a function to print a message

# Function definition
def print_message():
    print("Please think in silence for 10 s.")
    print("Shout your favorite number!")

# Main program + Function call
print("Welcome to the program!")
#Function call
print_message()
```

> **FYI:** A **function** is a segment of code that performs a single task.
> A **function definition** is the segment of code that tells the program what to do when the function is executed. The first line of a function definition (plus any docstrings) is the **function header**

1.   Closely examine the Python program above.

   a.   What Python **keyword** is used to indicate that a code segment is a **function definition**?

   _____

   b.   What is the **function header** in the Python code?

   _____
   _____

   c.   The name of the function is in the function header. What is the name of the function above?

   _____

   d.   The **function call** is when we tell python to run the function. Circle the function call.

_____

e.   What will the output be? (Confirm by entering & executing the code – as a class)

_____

_____

_____

f.   What line of code would you add to the program to print the last two lines twice? Where would you add the code?

_____

_____

2.   Examine the following program:

Python Program

```python
# Description: This program uses function to
# calculate the area of a circle, given the radius
def calculate_area(radius):
    area = 3.141592 * radius**2
    print("Area of a circle with a radius of",radius,"is",area)


radius = int(input("Enter the radius: "))
calculate_area(radius)
```

a.   Label the **function definitions** and the **function calls**.

b.   The function call and the function definition for `calculate_area` each include a variable within the parentheses. The variable in the function call is known as an **argument**. The variable in the function definition is called a **parameter.** What is the parameter in the function definition? What is its purpose?

_____

_____

c.   In this example the parameter in the _function definition_ and the argument in the _function call_ have the same name. Is this required?   _____

d.   Enter and execute the program. Verify your answer to question (c) by changing the variable name in the main function from **radius** to **number**. Do not change the parameter variable name in the function definition. Does the program still work? _____

e.   Write a line of code that calls the **calculate_area** function and sends the value "6" as the argument. Add the line of code to the main program and execute it to be sure it works properly. _____

f.   Add another function to the program that calculates and prints the _diameter_ of a circle, given the radius as the parameter. Place the function definition above call to the main function of the program. Write the function below.

_____

_____

_____

g.   Add another line of code to the main function of the program to call the function that was created in part 'f'. Send the radius entered by the user as the argument to the function.

_____

```
    @..@
   (----)
  ( >__< )
  ^^  ~~  ^^
```

Application Questions: Use the Python Interpreter to check your work
1.   Write a function that draws a frog. Call the function to be sure it works. Sample frog:

_____

_____

_____

_____

_____

_____

_____

2.   Expand the program in #1 to produce the following output (without copying & pasting tons of lines of code!)

```
One frog...
      @..@
     (----)
    ( >__< )
    ^^  ~~  ^^
Two frog...
      @..@
     (----)
    ( >__< )
    ^^  ~~  ^^
Three frog...
      @..@
     (----)
    ( >__< )
    ^^  ~~  ^^
Four...
      @..@
     (----)
    ( >__< )
    ^^  ~~  ^^
```

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____