Functions

Content Learning Objectives

After completing this activity, students should be able to:

- Explain the meaning and purpose of a function that does not return a value
- Explain the meaning and purpose of a value returning function
- Recognize a function definition, function header, and function call in a program
- Explain programs that use the same function multiple times

Process Skill Goals

- Write code that includes function definitions and functions calls
- Write programs that incorporate functions and if/else statements

Part 1: Functions

- A ***function*** is a segment of code that performs a single task
- ***Functions*** allow for code reusability while saving space in your program

Parts of a Function:

- A ***function definition*** is the segment of code that creates the function and tells the program what to do when the function is called
- The line that has the function definition is known as the ***function header***
- A ***function call*** uses the function name outside of the function to direct Python to the function

In [ ]:
```
def printRain(): #Function Definition
#A function definition creates the function
#by specifying a function name

    #Whatever the function does is indented just like if statements
    #Function Body
    print('  (           )')
    print(' (             )')
    print('(_______________)')
    print(' ///////////////')
    print(' /It is Raining/')
    print(' ///////////////')



def main(): #Function Definition

    printRain() #Function Call
    #This directs Python to the function to do whatever
    #the function specifies

main() #Function Call
```

Look at the code above and answer the questions below.

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

def.

2. What are the two ***function headers*** in the Python code?

def printRain():

and

def main():

3. The name of the functions are in the ***function headers***. What are the ***function names***?

printRain

and

main

4. Add the code to a new Python file and run it. What is the output?

```
  (                    )
 (                      )
(_____________________)
 ////////////////
 /It is Raining/
 ////////////////
```

5. What line of code would you add to print another cloud?

main()

6. Where would you add the line?

after the ending main then it would print twice.

Part 2: Functions with Arguments

So far we have three parts of a function: function definition, function header, and function call. Now we'll be looking at ***function arguments/parameters***.

Function arguments/parameters allow for variables outside the scope of a function to be used in the function.

Look at the functions below then answer the questions.

In [ ]:
```
def FahrenheitToCelsius(tempF): #Function Definition
    #This function calculates the temperature in Celsius
    #given a Fahrenheit temperature
    tempC = (tempF - 32) * 5/9
    print(tempC, "C")

def CelsiusToFahrenheit(tempC): #Function Definition
    #This function calculates the temperature in Fahrenheit
    #given a Celsius temperature
    tempF = tempC*9/5 + 32
    print(tempF, "F")


def main():
    tempF = int(input("Enter a Fahrenheit temperature: "))
    FahrenheitToCelsius(tempF) #Function Call
    tempC = int(input("Enter a Celsius temperature: "))

main()
```

The function call and the function definition for ***FahrenheitToCelsius*** 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***.

7. What is the parameter in the ***FahrenheitToCelsius()*** function definition?

tempF

8. What is the purpose of a parameter?

to hold the place for an argument.

9. The ***parameter*** and the ***argument*** have the same name in this example. Is this required?

No

10. Copy and paste this code into your program. Verify your answer to 9 by changing the variable in the ***main()*** function from ***tempF*** to ***temperature***. Do not change the parameter variable name in the function definition but make sure you change the argument name in the function call. Does the program still work?

yes

11. Write a line of code that calls the ***CelsiusToFahrenheit()*** function with the argument ***tempC***. Paste the line you added below.

CelsiusToFahrenheit(tempC)

12. Where did you add the line of code.

right under the TempC = int(input("Enter a Celcius temerature: ")) line.

or in the main function.

13. Create a new function called ***feetToKilometers()*** which takes ***feet*** as a parameter. The conversion from feet to kilometers is kilometers = feet / 3281.

```
def feetToKilometers(feet):
    Kilometers = feet / 3281
    print (Kilometers, "Kilometers")

feet = int(input("Enter feet"))
feetToKilometers(feet)
```

14. Add another line of code in the ***main()*** function that calls the function created in question 13. Paste the line in the textbox below.

```
feetToKilometers(feet)
```

15. Try calling the ***feetToKilometers()*** function with no argument. Does it work? If not, what do you need to add to make it work?

no it does not work it doesnt work because it requires 1 positional argument. that being feet

Part 3: Value Returning Functions

Some functions do not send back any information to the line that called the function. This is called a ***void function***. Functions that do send back information are called ***value returning functions***.

Now we have four parts of a function: function definitions, function headers, function calls, and function arguments/parameters. We will be adding one last part, ***function returns***.

In [ ]:
```
def kilometersToFeet(km):
    #add conversion
    feet = km * 3281
    return feet

def main():
    feet_distance = kilometersToFeet()

main()
```

16. What is the new keyword used in the function body?

return

17. What does this new keyword tell the program to do?

The new key word tells the program to return a value to main.

18. What line of code calls the function ***kilometersToFeet()***?

feet_distance = kilometersToFeet()

19. What is different about this line compared to previous void function calls?

we have a variable assignment.

20. Why is the function call on the right side of an assignment statement?
because the assignment wouldnt be the the right variable causing problems later.

Part 4: Write Your Own Function

The best way to learn something is through practice. For this you will be creating a function called ***distanceConversion()*** that can convert between distance units.

distanceConversion() shall:

- have three parameters ***dist*** (int), ***start_unit*** (str), and ***end_unit*** (str)
- convert from kilometers to feet
- convert from kilometers to miles
- convert from feet to kilometers
- convert from feet to miles
- convert from miles to kilometers
- convert from miles to feet
- use if/else statements
- return the converted distance, ***end_value***

```
def distanceConversion(dist, start_unit, end_unit):
    if start_unit == "Km" and end_unit == "Ft" :
        end_value = dist * 3281
    elif start_unit == "Ft" and end_unit == "Km" :
        end_value = dist / 3281
    elif start_value == "Km" and end_unit == "Mi" :
        end_value = dist / 1.61
    elif start_value == "Ft" and end_unit == "Mi" :
        end_value = dist / 5280
    elif start_value == "Mi" and end_unit == "Ft" :
        end_value = dist * 5280
    elif start_value == "Mi" and end_unit == "Km" :
        end_value = dist * 1.61

    return end_value

dist = 526
start_value = "Mi"
end_value = "Ft"

def main():
    distance = distanceConversion(dist, start  value, end  value)
```