VIDEO 9 : FUNCTIONS

Functions allow use to reuse code and make the code easier to understand. To create a function type def (define) the function name and then in parentheses a comma separated list of values that function can accept.

This function adds 2 values and returns the sum.

CODE

```
def add_numbers(num_1, num_2):

    # Return returns a value if needed
    return num_1 + num_2

# You call the function by name followed by passing comma
# separated values if needed and a value may or may not be
# returned

print("5 + 4 =", add_numbers(5, 4))
```

Function Local Variables

Any variable defined inside a function is not available outside of that function. For example

CODE

```
def assign_name():
    name = "Doug"

assign_name()

# Throws a NameError
# print(name)
```

Global Variables

You can't change a global variable even if it is passed into a function. That is because a value and not the actual variable is passed to the function.

CODE

```
def change_name(name):

    # Trying to change the global
    name = "Mark"

# A variable defined outside of a function can't be changed
# in the function using the above way
name = "Tom"

# Try to change the value
change_name(name)
```

```
# Prints Tom even though the function tries to change it
print(name)
```

If you want to change the value pass it back

CODE

```
def change_name_2():
    return "Mark"

name = change_name_2()
print(name)
```

You can also use the global keyword to change it.

CODE

```
gbl_name = "Sally"

def change_name_3():
    global gbl_name
    gbl_name = "Sammy"

change_name_3()
print(gbl_name)
```

If you don’t return a value a function will return none.

CODE

```
def get_sum(num1, num2):
    sum = num1 + num2

print(get_sum(5, 4))
```

MAKE A is_float FUNCTION

There is no way to check if a string contains a float so let's make one by defining our own function.

```
def is_float(str_val):
    try:

        # If the string isn't a float float() will throw a
        # ValueError
        float(str_val)

        # If there is a value you want to return use return
        return True
    except ValueError:
        return False
```

```
pi = 3.14
print("Is Pi a Float :", is_float(pi))
```

Python Problem for you to Solve

In this problem you'll receive an algebraic equation and solve for x. Know that x will always be the 1st value received and you only will deal with addition. Here is a sample of calling the function and then the output.

```
print(solve_eq("x + 4 = 9"))
x = 5
```

Solution

```
def solve_eq(equation):
    x, add, num1, equal, num2 = equation.split()

    # Convert the strings into ints
    num1, num2 = int(num1), int(num2)

    # Convert the result into a string and join (concatenate)
    # it to the string "x = "
    return "x = " + str(num2 - num1)

print(solve_eq("x + 4 = 9"))
```

That's it for this video. In the next video I'll show you how to return and receive multiple values. We'll also calculate primes, calculate areas for different shapes and talk about main().