Video 22 : Functions as Objects

In this video I'll explore how we can treat functions as objects which opens up a world of possibilities. We'll also explore function annotations. Then we'll present you with another problem for you to solve.

CODE

```
# Function multiplies a parameter by 2
def mult_by_2(num):
    return num * 2

# A function can be
# 1. Assigned to another name

times_two = mult_by_2

print("4 * 2 =", times_two(4))

# 2. Passed into other functions

def do_math(func, num):

    return func(num)

print("8 * 2 =", do_math(mult_by_2, 8))

# 3. Returned from a function

def get_func_mult_by_num(num):

    # Create a dynamic function that will receive a value
    # and then return that value times the value passed
    # into get_func_mult_by_num()
    def mult_by(value):
        return num * value

    return mult_by

generated_func = get_func_mult_by_num(5)

print("5 * 10 =", generated_func(10))

# 4. Embedded in a data structure

list_of_funcs = [times_two, generated_func]

print("5 * 9 =", list_of_funcs[1](9))
```

Python Problem for you to Solve

Now that we have explored new ways we can use functions let's try another problem. I want you to create a function that receives a list and a function. The function passed will return True

or False if a list value is odd. And then the surrounding function will return a list of odd numbers.

Solution

```
def is_it_odd(num):
    if num % 2 == 0:
        return False
    else:
        return True

def change_list(list, func):

    odd_list = []

    for i in list:
        if func(i):
            odd_list.append(i)

    return odd_list

a_list = range(1, 21)

print(change_list(a_list, is_it_odd))
```

Function Annotations

It is possible to define the data types of attributes and the returned value with annotations, but they have no impact on how the function operates, but instead are for documentation.

```
def random_func(name: str, age: int, weight: float) -> str:
    print("Name :", name)
    print("Age :", age)
    print("Weight :", weight)

    return "{} is {} years old and weighs {}".format(name, age, weight)

print(random_func("Derek", 41, 165.5))

# You don't get an error if you pass bad data
print(random_func(89, "Derek", "Turtle"))

# You can print the annotations
print(random_func.__annotations__)
```

That's it for this video. In the next video we'll cover Anonymous functions, lambda, map, filter, reduce and 2 new problems.