Creating a Testing Function

Creating a Testing Function
Let's remember the program we saw before that is supposed to double a number, but that it doesn't do it correctly. If we were to present it as a function, we could have it taking in an input value, multiply that value by 2 (but we've accidently multiplied it by 3), and returning the answer back, as follows:

Double Number FUNCTION

```
def DoubleNumber(InputValue):

    TheResult = InputValue * 3    #This is wrong
    return TheResult

# END DoubleNumber.
```

And we could call the program as follows (by adding the following to the bottom of the program in the same file, or by typing it directly into the command prompt):

```
DoubleNumber(3)
```

And unfortunately, the answer we would get is:

```
9
```

To help automatically test if a function is working, sometimes it is easier to create a new function (usually with the same name as the original function, but proceeded with `test_`) to check if the function is working. In the simple case of doubling a number, it might not be necessary, but if the function had several inputs, or you wanted to run a lot of tests, it can be useful. Our example would look as follows:

Test Double Number FUNCTION

```
def test_DoubleNumber():

    OurCheck = DoubleNumber(3)
    if (OurCheck == 6):
        print("This test was passed.")
    else:
        print("We better check DoubleNumber!!!")
    #EndIf;

# END test_DoubleNumber.
```

And we could call the program as follows:

```
test_DoubleNumber()
```

Note that if the result we get is correct (so if `DoubleNumber(3)` did give 6) , we just say "This test is passed.", we don't say "The function is working" because we don't know, we'd have to do a lot more testing on the function to be sure. So, we'd have to test it on zero, negative numbers, really big numbers, decimals, blank inputs, characters, etc. before we could start to have confidence in the function.