Common Issues with Functions

Function Name

When we are calling a function, it's really important to get the name of the function right, this might seem obvious, but it's amazing how often people get the name wrong, and it's a big problem because the computer won't know which function we are talking about unless we get the name right (it can't guess). This is particularly a problem if the name of the function is made up of several words, so, for example, we remember the function `IsDivisibleBy3`, it could go wrong like this:

WRONG CODE	REASON
`print(DivisibleBy3(15))`	Needs the correct name: `print(IsDivisibleBy3(15))`

Input Parameters

When we are calling a function it's really important to know how many parameters (input values) we need to pass into a function, because if we pass in too many parameters, or too few parameters, it will give us an error. So, for example, we remember the function `IsDivisibleByN` takes in two parameters that are numbers, so `print(IsDivisibleByN(15, 2))` returns `False`, because 2 doesn't divide evenly into 15, but `print(IsDivisibleByN(15, 3))` returns `True`. The function takes in two parameters, and it could go wrong like this:

WRONG CODE	REASON
`IsDivisibleByN(15)`	Too few parameters
`IsDivisibleByN(15, 2, 4)`	Too many parameters

Another common issue is when we pass in the wrong <u>type</u> of parameters, so for example, the `IsDivisibleByN` takes in two parameters that are numbers, and it could go wrong if something other than numbers are input as parameters:

WRONG CODE	REASON
`IsDivisibleByN(@, &)`	The parameters are characters
`IsDivisibleByN(False, True)`	The parameters are Boolean

The Return Value

The two functions `IsDivisibleBy3` and `IsDivisibleByN` both return a Boolean value (either `True` or `False`), so when we call those functions we need to make sure that we are checking for the right return type:

WRONG CODE	REASON
`if (IsDivisibleBy3(15) > 7):`	The return value is a Boolean so you can't compare it to a number (7).

So it's important to understand what values are going in and out of a function.