The `format()` function

It is desirable to print program output in a nicely formatted manner. The `format()` function enables us to do that. It also enables us to combine string and numerical values together. Suppose we want to print a string that contains two variables that we wish to format and print: The syntax is as follows: `'-----{0}---- {1}-----'.format(value1, value2)` The dashes are any string text that we wish to display. `value1` corresponds to the index value of 0 and `value2` corresponds to the index value of 1. Therefore the first curly brace is replaced by `value1` and the second curly brace is replaced by `value2`.

If we switch the curly braces with 0 and 1 as shown below `'-----{1}---- {0}-----'.format(value1, value2)` then, the first curly brace will be replaced by `value2` and the second curly brace will be replaced by `value1`

In [2]:

```
'''
We can refer to the arguments of the format() function in any order we choose as long as we use th
e correct integer values
inside the curly braces.  These values correspond to the order in which the arguments appear insid
e the format() function.
'''
x = 5
y = 3
print('{2} is the square of {1} and {3} is the square of {0}'.format(y, x, x**2 , y**2))
print('{} is the square '.format(y, x))
```

```
25 is the square of 5 and 9 is the square of 3
3 is the square
```

In [6]:

```
'''
You can use the format() function to format float values with precision.
For example, use {0:0.3f} to format the value to 3 decimal places.
'''
'''
print the value of 33 divided by 7 formatted to four decimal places
'''
x = 33
y = 33/7
print('The quotient of {0} is {1:0.4f}'.format(x, y))
```

```
The quotient of 33 is 4.7143
```

In [3]:

```
'''
The `,` can be used as a separator together with the `format()` function to display numerical valu
es
'''
x = 1000005
y = x/1000
print('The square root of {0} is {1:,.3f}'.format(x, y)) # Note the use of a comma
```

```
The square root of 1000005 is 1,000.005
```