Print Formatting

In this lecture we will briefly cover the various ways to format your print statements. As you code more and more, you will probably want to have print statements that can take in a variable into a printed string statement.

The most basic example of a print statement is:

In [17]:
```
print 'This is a string'
```

```
This is a string
```

Strings

You can use the %s to format strings into your print statements.

In [4]:
```
s = 'STRING'
print 'Place another string with a mod and s: %s' %(s)
```

```
Place another string with a mod and s: STRING
```

Floating Point Numbers

Floating point numbers use the format %n1.n2f where the n1 is the total minimum number of digits the string should contain (these may be filled with whitespace if the entire number does not have this many digits. The n2 placeholder stands for how many numbers to show past the decimal point. Lets see some examples:

```
print 'Floating point numbers: %1.2f' %(13.144)
```

In [13]:
```
print 'Floating point numbers: %1.0f' %(13.144)
```

```
Floating point numbers: 13
```

In [14]:
```
print 'Floating point numbers: %1.5f' %(13.144)
```

```
Floating point numbers: 13.14400
```

In [15]:
```
print 'Floating point numbers: %10.2f' %(13.144)
```

```
Floating point numbers:      13.14
```

In [19]:
```
print 'Floating point numbers: %25.2f' %(13.144)
```

```
Floating point numbers:                     13.14
```

Conversion Format methods.

It should be noted that two methods %s and %r actually convert any python object to a string using two separate methods: str() and repr(). We will learn more about these functions later on in the course, but you should note you can actually pass almost any Python object with these two methods and it will work:

In [23]:
```
print 'Here is a number: %s. Here is a string: %s' %(123.1,'hi')
```

```
Here is a number: 123.1. Here is a string: hi
```

In [24]:
```
print 'Here is a number: %r. Here is a string: %r' %(123.1,'hi')
```

```
Here is a number: 123.1. Here is a string: 'hi'
```

Multiple Formatting

Pass a tuple to the modulo symbol to place multiple formats in your print statements:

In [22]:
```
print 'First: %s, Second: %1.2f, Third: %r' %('hi!',3.14,22)
```

```
First: hi!, Second: 3.14, Third: 22
```

Using the string .format() method

The best way to format objects into your strings for print statements is using the format method. The syntax is:

```
'String here {var1} then also {var2}'.format(var1='something1',var2='some
thing2')
```

Lets see some examples:

In [27]:
```
print 'This is a string with an {p}'.format(p='insert')
```

```
This is a string with an insert
```

In [28]:
```
# Multiple times:
print 'One: {p}, Two: {p}, Three: {p}'.format(p='Hi!')
```

```
One: Hi!, Two: Hi!, Three: Hi!
```

In [29]:
```
# Several Objects:
print 'Object 1: {a}, Object 2: {b}, Object 3: {c}'.format(a=1,b='two',c=12.3)
```

```
Object 1: 1, Object 2: two, Object 3: 12.3
```

In [2]:
```
from __future__ import print_function
```

In [3]:
```
print('I love {x}'.format(x='my wife'))
```

```
I love my wife
```

That is the basics of string formatting! Remember that Python 3 uses a print() function, not the print statement!