String "Interpolation"

In [2]:
```
'My favorite number is %d' % 42
```

Out[2]: 'My favorite number is 42'

In [3]:
```
'My favorite number is %(num)d' % { 'num': 42 }
```

Out[3]: 'My favorite number is 42'

In [4]:
```
'The coordinates are (%d,%d)' % (5,6)
```

Out[4]: 'The coordinates are (5,6)'

In [5]:
```
'Hello, %s' % 'World'
```

Out[5]: 'Hello, World'

In [6]:
```
'Hello, %s' % 42 # calls str() on arguments
```

Out[6]: 'Hello, 42'

In [7]:
```
'Hello, %r' % 'World' # calls repr() on arguments
```

Out[7]: "Hello, 'World'"

In [8]:
```
import math
'Pi, to 3 decimals, is %.3f' % math.pi
```

Out[8]: 'Pi, to 3 decimals, is 3.142'

In [15]:
```
'Padding with spaces: "%20d"' % 42
```

Out[15]: 'Padding with spaces: "   42"'

In [14]:
```
'Padding with spaces: "%-20d"' % 42
```

Out[14]: 'Padding with spaces: "42   "'

In [13]:
```
'Padding with zeros: "%.20d"' % 42
```

Out[13]: 'Padding with zeros: "00000000000000000042"'

In [16]:
```
'Float padding: "%10.4f"' % math.pi
```

Out[16]: 'Float padding: 3.1416'

In [18]:
```
'Int padding: "%10.4d"' % 42
```

Out[18]: 'Int padding: "   0042"'

String methods

In [19]:
```
text = '''
The quick brown
 fox jumped
over the
    lazy
dog
'''
```

In [67]:
```
words = text.split()
words
```

Out[67]:
```
['The', 'quick', 'brown', 'fox', 'jumped', 'over', 'the', 'lazy', 'dog']
```

In [68]:
```
' - '.join(words)
```

Out[68]:
```
'The - quick - brown - fox - jumped - over - the - lazy - dog'
```

In [21]:
```
text.splitlines()
```

Out[21]:
```
['', 'The quick brown', ' fox jumped', 'over the', '    lazy    ', 'dog']
```

In [22]:
```
text
```

Out[22]:
```
'\nThe quick brown\n fox jumped\nover the\n    lazy    \ndog\n'
```

In [23]:
```
text.strip()
```

Out[23]:
```
'The quick brown\n fox jumped\nover the\n    lazy    \ndog'
```

In [24]:
```
text.lstrip()
```

Out[24]:
```
'The quick brown\n fox jumped\nover the\n    lazy    \ndog\n'
```

In [25]:
```
text.rstrip()
```

Out[25]:
```
'\nThe quick brown\n fox jumped\nover the\n    lazy    \ndog'
```

In [30]:
```
text.title()
```

Out[30]:
```
'\nThe Quick Brown\n Fox Jumped\nOver The\n    Lazy    \nDog\n'
```

In [31]:
```
'55'.isdigit()
```

Out[31]:
```
True
```

In [32]:
```
'55a'.isdigit()
```

Out[32]:
```
False
```

In [35]:
```
'55a'.isalnum()
```

Out[35]: True

In [36]:
```
'33'.isalpha()
```

Out[36]: False

In [37]:
```
text.startswith('\n')
```

Out[37]: True

In [38]:
```
text.endswith('\n')
```

Out[38]: True

In [39]:
```
text.index('fox')
```

Out[39]: 18

In [40]:
```
text[18:]
```

Out[40]: 'fox jumped\nover the\n   lazy   \ndog\n'

In [42]:
```
text.find('gopher')
```

Out[42]: -1

In [43]:
```
text.index('gopher')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
/vagrant/<ipython-input-43-bbb5e3c56822> in <module>()
----> 1 text.index('gopher')

ValueError: substring not found
```

In [44]:
```
'foo bar baz foo'.find('foo')
```

Out[44]: 0

In [46]:
```
'foo bar baz foo'.find('foo', 1)
```

Out[46]: 12

In [29]:
```
print 'Hello'.center(80)
```

```
                                      Hello
```

In [47]:
```
'foo bar baz foo'.rfind('foo')
```

Out[47]: 12

In [52]:
```
'foo, bar, baz, foo'.split(',')
```

Out[52]:
```
['foo', ' bar', ' baz', ' foo']
```

In [54]:
```
first, rest = 'foo, bar, baz, foo'.split(',', 1)
rest
```

Out[54]:
```
' bar, baz, foo'
```

In [55]:
```
rest, last = 'foo, bar, baz, foo'.rsplit(',', 1)
rest
```

Out[55]:
```
'foo, bar, baz'
```

In [56]:
```
text.lower()
```

Out[56]:
```
'\nthe quick brown\n fox jumped\nover the\n    lazy    \ndog\n'
```

In [57]:
```
text.upper()
```

Out[57]:
```
'\nTHE QUICK BROWN\n FOX JUMPED\nOVER THE\n    LAZY    \nDOG\n'
```

In [60]:
```
text.title().swapcase()
```

Out[60]:
```
'\ntHE qUICK bROWN\n fOX jUMPED\noVER tHE\n    lAZY    \ndOG\n'
```

String templates

In [61]:
```
import string
```

In [62]:
```
tmpl = string.Template('''
Dear $to,

I am intrigued by your ideas and wish to
subscribe to your newsletter.

Best regards,
$from
''')
```

In [64]:
```
dct = {'to': 'Rick', 'from': 'Stuart'}
print tmpl.substitute(dct)
```

```
Dear Rick,

I am intrigued by your ideas and wish to
subscribe to your newsletter.

Best regards,
Stuart
```

In [66]:
```
dct = { 'to': 'Rick' }
print tmpl.safe_substitute(dct)
```

```
Dear Rick,

I am intrigued by your ideas and wish to
subscribe to your newsletter.

Best regards,
$from
```

Exercises

- Write a function that will read a file and return a list of all the words in the file.
- Enhance this function to return a `dict` containing each distinct word as a key and the number of occurrences as the value.
- Write a function that takes a list of strings as a parameter and prints them, in title case, centered for an 80-column display.