String Formatting

- String formatting provides a more powerful way to embed non-strings within strings
- String formatting uses a string's **format** method to substitute a number of arguments in the string
- **Example:**

```
# string formatting
nums = [4, 5, 6]
msg = "Numbers: {0} {1} {2}". format(nums[0], nums[1], nums[2])
print(msg)
```

String Formatting

- **Result:**

```
>>>
Numbers: 4 5 6
>>>
```

- Each argument of the format function is placed in the string at the corresponding position, determined using the curly braces { }
- String formatting can also be done with named arguments

String Formatting

- **Example:**

```
a = "{x}, {y}".format(x=5, y=12)
print(a)
```

- **Result:**

```
>>>
5, 12
>>>
```

String Functions

- Python contains many useful built-in functions and methods to accomplish common tasks
- **join** - joins a list of strings with another string as a separator
- **replace** - replaces one substring in a string with another
- **startswith** and **endswith -** determine if there is a substring at the start and end of a string, respectively
- To change the case of a string, you can use **lower** and **upper**

String Functions

- The method **split** is the opposite of **join**, turning a string with a certain separator into a list
- **Some examples:**

```
print(", ".join(["spam", "eggs", "ham"]))
#prints "spam, eggs, ham"

print("Hello ME".replace("ME", "world"))
#prints "Hello world"
```

String Functions

```
print("This is a sentence.".startswith("This"))
# prints "True"

print("This is a sentence.".endswith("sentence."))
# prints "True"

print("This is a sentence.".upper())
# prints "THIS IS A SENTENCE."
```

String Functions

```
print("AN ALL CAPS SENTENCE".lower())
#prints "an all caps sentence"

print("spam, eggs, ham".split(", "))
#prints "['spam', 'eggs', 'ham']"
```

Numeric Functions

- To find the maximum or minimum of some numbers or a list, use **max** or **min**
- To find the distance of a number from zero (its absolute value), use **abs**
- To round a number to a certain number of decimal places, use **round**
- To find the total of a list, use **sum**

Numeric Functions

- **Some examples:**

```
print(min(1, 2, 3, 4, 0, 2, 1))
print(max([1, 4, 9, 2, 5, 6, 8]))
print(abs(-99))
print(abs(42))
print(sum([1, 2, 3, 4, 5]))
```

Numeric Functions

- **Result:**

```
>>>
0
9
99
42
15
>>>
```

List Functions

- Often used in conditional statements
- **all** and **any** take a list as an argument
- return **True** if all or any (respectively) of their arguments evaluate to **True** (and **False** otherwise)
- Function **enumerate** can be used to iterate through the values and indices of a list simultaneously

List Functions

- **Example:**

```
nums = [55, 44, 33, 22, 11]

if all([i > 5 for i in nums]):
print("All larger than 5")

if any([i % 2 == 0 for i in nums]):
print("At least one is even")

for v in enumerate(nums):
print(v)
```

List Functions

- **Result:**

```
>>>
All larger than 5
At least one is even
(0, 55)
(1, 44)
(2, 33)
(3, 22)
(4, 11)
>>>
```