2_operators

April 12, 2024

1 Operators and comparisons

Most operators and comparisons in Python work as one would expect:

- Arithmetic operators `+`, `-`, `*`, `/`, `//` (integer division), '**' power

[2]:
```
1 + 2, 1 - 2, 1 * 2, 2 / 2
```

[2]:
```
(3, -1, 2, 1.0)
```

[3]:
```
1.0 + 2.0, 1.0 - 2.0, 1.0 * 2.0, 1.0 / 2.0
```

[3]:
```
(3.0, -1.0, 2.0, 0.5)
```

[4]:
```
# Integer division of float numbers
3.0 // 2.0
```

[4]:
```
1.0
```

[5]:
```
# Note! The power operators in python isn't ^, but **
2 ** 3
```

[5]:
```
8
```

Note: Division

The `/` operator always performs a floating point division in Python 3.x. This is not true in Python 2.x, where the result of `/` is always an integer if the operands are integers. To be more specific, `1/2 = 0.5` (`float`) in Python 3.x, and `1/2 = 0` (`int`) in Python 2.x (but `1.0/2 = 0.5` in Python 2.x).

The boolean operators are spelled out as the words `and`, `not`, `or`.

[6]:
```
True and False
```

[6]:
```
False
```

[7]:
```
not False
```

[7]:
```
True
```

[8]:
```
True or False
```

[8]:
```
True
```

Comparison operators `>`, `<`, `>=` (greater or equal), `<=` (less or equal), `==` equality, `is` identical.

[10]:
```
2 > 1, 2 < 1
```

[10]:
```
(True, False)
```

[11]:
```
2 > 2, 2 < 2
```

[11]:
```
(False, False)
```

[13]:
```
2 >= 2, 2 <= 2
```

[13]:
```
(True, True)
```

[16]:
```
# equality
[1,2] == [1,2]
```

[16]:
```
True
```

[17]:
```
# objects identical?
l1 = l2 = [1,2]

l1 is l2
```

[17]:
```
True
```