Basic Python Operators

Shouke Wei, Ph.D. Professor

Email: shouke.wei@gmail.com

Objective

This section offers some basic operators used to perform operations on variables and values, which covers:

- Arithmetic Operators
- Comparison Operators
- Assignment Operators
- Logical Operators

1. Arithmetic Operators

- Python is an advanced caculator

1.1 Numberics

In [25]:
```
%%html
<style>
table {float:left}
</style>
```

Table 1: Arithmetic operators for common mathematical operations

Operator	Name	Description	Example
+	Addition	Returns the sum of two expressions	a + b
-	Subtraction	Returns the difference of two expressions	a - b
*	Multiplication	Returns the product of two expressions	a * b
/	Division	Returns the quotient of two expressions	a / b
%	Modulus	Returns the decimal part (remainder) of the quotient	a % b
**	Power	Returns the value of a numeric expression raised to a specified power	a**b
//	Floor division	Returns the integer part of the quotient	a//b

Addition

```
In [11]: a = 8
         b= 3
         c = a + b
         print(c)

         11
```

Subtraction

```
In [12]: d = a - b
         print(d)

         5
```

Multiplication

```
In [13]: e = a * b
         print(e)

         24
```

Division

```
In [14]: f = a / b
         print(f)

         2.6666666666666665
```

Modulus

- Remainder when a is divided by b

```
In [15]: g = a % b
         print(g)

         2
```

Exponentiation

- a raised to the power of b

```
In [16]: h = a**b
         print(h)

         512
```

Floor division

- also called Integer Division,
- Quotient when a is divided by b, rounded to the next smallest whole number

```
In [17]: i = a//b
         print(i)

         2
```

- Python puts no limit on the size of an integer

```
In [20]: len(str(9999999999**100000))

Out[20]: 1000000
```

1.2 Operators with Strings

String addition

- Concatenate strings using the operator

In [1]:
```
hello = "hello"
name = "world"
helloworld = hello + " " + name + '!'
print(helloworld)
```

```
hello world!
```

- But mixing operators between numbers and strings is not supported

In [2]:
```
x = 5
y = 8
z = 'Hello'

print(z + x + y)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_5632/3380406417.py in <module>
      3 z = 'Hello'
      4 
----> 5 print(z + x + y)

TypeError: can only concatenate str (not "int") to str
```

String multiplication

- form a string with a repeating sequence

In [ ]:
```
multihellos = "hello" * 20
print(multihellos)
```

1.3 Operators with Lists and Tuples

Addition

- Join lists or tuples together

In [3]:
```
even_numbers = [2,4,6,8]
odd_numbers = [1,3,5,7]
all_numbers = odd_numbers + even_numbers
print(all_numbers)
```

```
[1, 3, 5, 7, 2, 4, 6, 8]
```

In [4]:
```
fruitList1 = ['Apple','Banana','Orange']
fruitList2 = ['Melon','Grape']

fruitList = fruitList1 + fruitList2
print(fruitList)
```

```
['Apple', 'Banana', 'Orange', 'Melon', 'Grape']
```

In [5]:
```
fruitTuple1 = ('Apple','Banana','Orange')
fruitTuple2 = ('Melon','Grape')

fruitTuple = fruitTuple1 + fruitTuple2
print(fruitTuple)
```

```
('Apple', 'Banana', 'Orange', 'Melon', 'Grape')
```

Multiplication

In [1]:
```
a = [2,4,6]
b = a*5
print(b)
```

```
[2, 4, 6, 2, 4, 6, 2, 4, 6, 2, 4, 6, 2, 4, 6]
```

In [6]:
```
fruitList = ['Apple','Banana','Orange']
fruitList*2
```

Out[6]:
```
['Apple', 'Banana', 'Orange', 'Apple', 'Banana', 'Orange']
```

In [3]:
```
fruitTuple = ('Apple','Banana','Orange')
fruitTuple*2
```

Out[3]:
```
('Apple', 'Banana', 'Orange', 'Apple', 'Banana', 'Orange')
```

2. Comparison Operators

- also called relational operators

Table 2: Comparison Operators for comparing two values

Operator	Name	Description	Example
==	Equal	Returns a Boolean stating whether two expressions are equal.	a == b
!=	Not equal	Returns a Boolean stating whether two expressions are not equal	a != b
>	Greater than	Returns a Boolean stating whether one expression is greater than the other	a > b
<	Less than	Returns a Boolean stating whether one expression is less than the other	a < b
>=	Greater than or equal to	Returns a Boolean stating whether one expression is greater than or equal the other	a >= b
<=	Less than or equal to	Returns a Boolean stating whether one expression is less than or equal the other	a <= b

In [4]:
```
a = 8
b = 10

print(a==b)
print(a!=b)
print(a>b)
print(a<b)
print(a>=b)
print(a<=b)
```

```
False
True
False
True
False
True
```

3. Assignment Operators

Table 3: Assignment operators for assigning values to variables

Operator	Description	Example	Equal to
=	simple assignment: Assigns a value to a variable(s)	a = 8	a = 8

Operator	Description	Example	Equal to
+=	increment assignment: Adds a value to the variable and assigns the result to that variable	a += 2	a = a + 2
-=	decrement assignment: Subtracts a value from the variable and assigns the result to that variable	a -= 2	a = a - 2
*=	multiplication assignment: Multiplies the variable by a value and assigns the result to that variable	a *= 2	a = a * 2
/=	division assignment: Divides the variable by a value and assigns the result to that variable	a /= 2	a = a / 2
%=	modulus assignment: Computes the modulus of the variable and assigns the result to that variable	a %= 2	a = a % 2
//=	floor division assignment: Floor divides the variable by a value and assigns the result to that variable	a //= 2	a = a // 2
**=	power assignment: Raises the variable to a specified power and assigns the result to the variable	a ** = 2	a = a **2

In [1]:
```
a = 5
a +=1
print(a)
```

```
6
```

In [2]:
```
b = 7
b -= 3
print(b)
```

```
4
```

4. Logical Operators

Table 4. Logical operators to evaluate true or false

Operator	Description	Example
and	Returns True if both statements are true	x<2 and x<8
or	Returns True if one of the statements is true	x<2 or x>8
not	Reverse the result, returns True if the result is False	not(x<2 and x<8)

In [2]:
```
c = 1
print(c < 2 and c<8)
print(c>2 and c<3)
print(not(c>2 and c<3))
```

```
True
False
True
```

In [3]:
```
x = 9
print(x<5 or x>8)
```

```
True
```

In [9]:
```
print(not(x<0 or x>10))
```

```
True
```

In [10]:
```
print(not(x>0 and x<10))
```

```
False
```