Universidad Autónoma de San Luis Potosí

Python Basics

What is and Why Python?

It is used for:

- web development (server-side)
- software development
- mathematics
- system scripting

- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
- Python has a simple syntax similar to the English language.
- Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
- Python can be treated in a procedural way, an object-oriented way or a functional way.

Getting Started with Python

- Check Python Version with: python --version
- Python CLI
- Creation of files with CLI
- Hello World!

Python Comments

Comments can be used to explain Python code.
Comments can be used to make the code more readable.
Comments can be used to prevent execution when testing code.
Multiline comments.

Comments starts with a #, and Python will ignore them.

```
# This is a comment
```

Python Variables

- Variables are containers for storing data values.

- Python has no command for declaring a variable.
A variable is created the moment you first assign a value to it.

- Variables do not need to be declared with any particular *type*, and can even change type after they have been set.

- You can get the data type of a variable with the type() function.

Python Variables

A variable can have a short name (like x and y) or a more descriptive name (age, carname, total_volume). Rules for Python variables:

1. A variable name must start with a letter or the underscore character
2. A variable name cannot start with a number
3. A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
4. Variable names are case-sensitive (age, Age and AGE are three different variables)
5. A variable name cannot be any of the Python keywords.

Multi words variable names: myCamelCase, MyPascalCase, my_snake_case

Python Basic Data Types

```
# String
greeting = 'Hello, friend’ # “Hello, friend”

# Integer
age = 35

# Floats
speed1 = 4.5
speed2 = 4.0
speed3 = 4.

# Boolean
am_i_short = True
am_i_tall = False
```

You can get the data type of any object by using the type() function

Assign multiple values

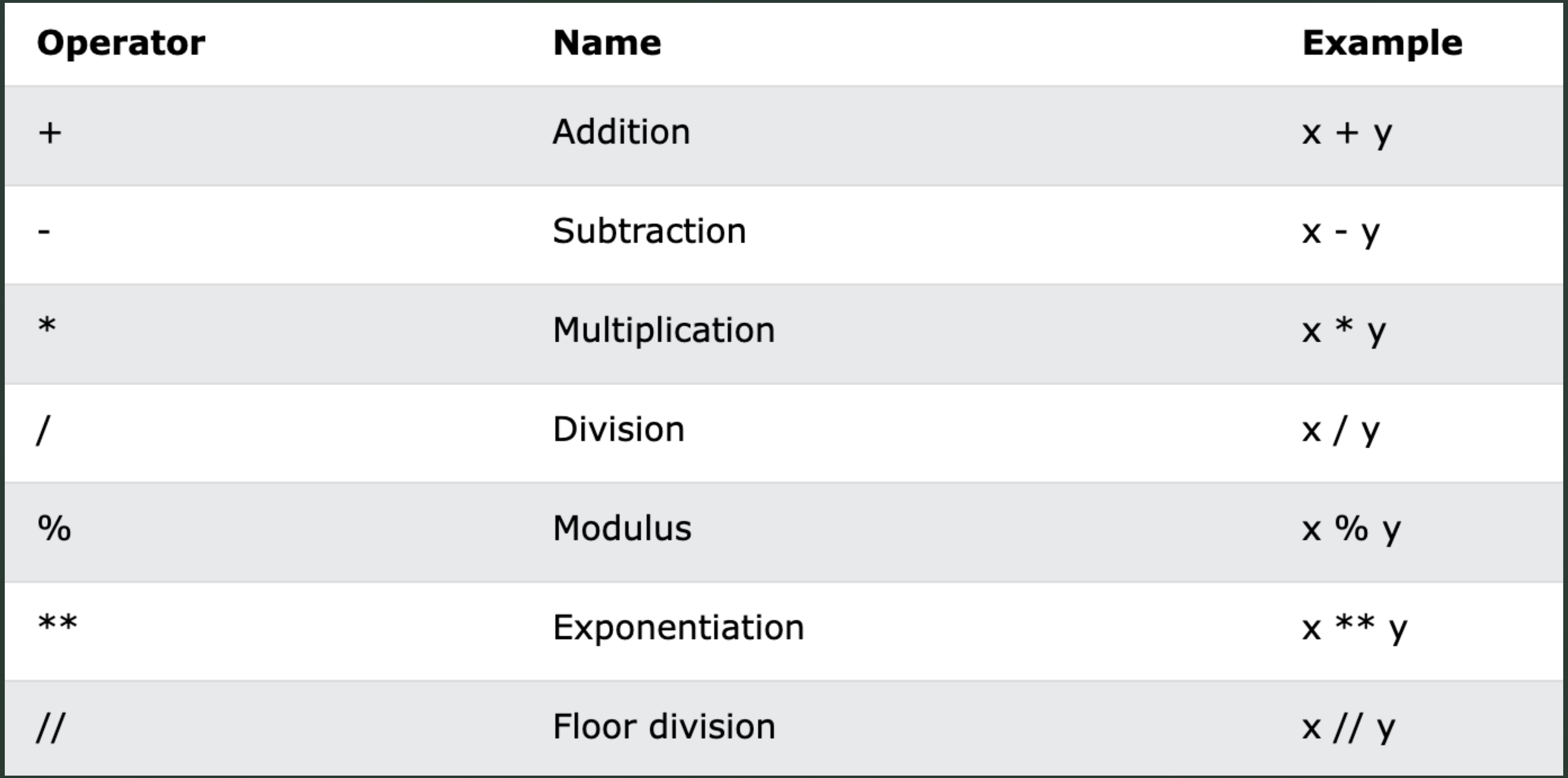
Python allows you to assign values to multiple variables in one line:

```
x, y, z = "Orange", "Banana", "Cherry"
print(x)
print(y)
print(z)
```

And you can assign the *same* value to multiple variables in one line:

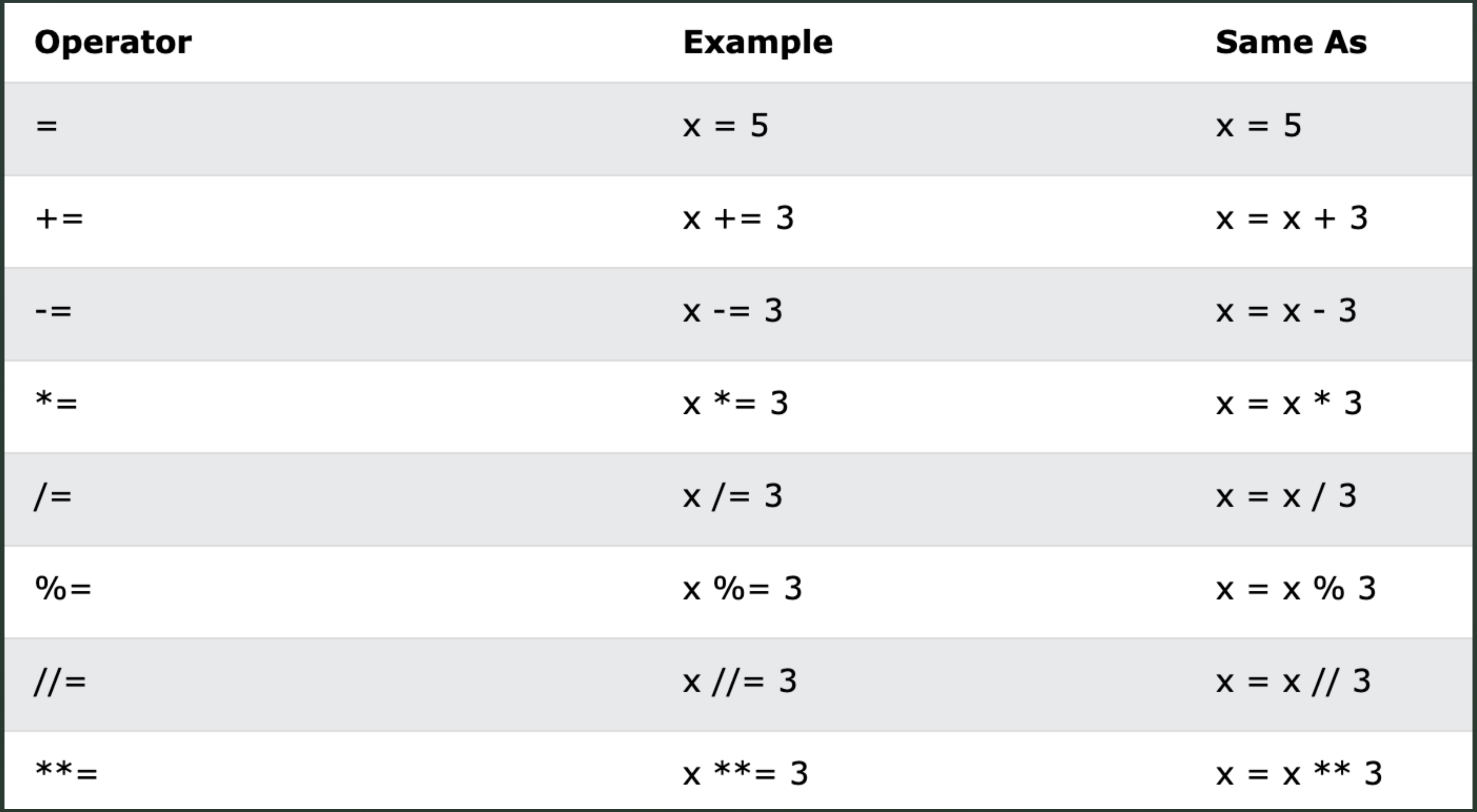
```
x = y = z = "Orange"
print(x)
print(y)
print(z)
```

Arithmetic Operators



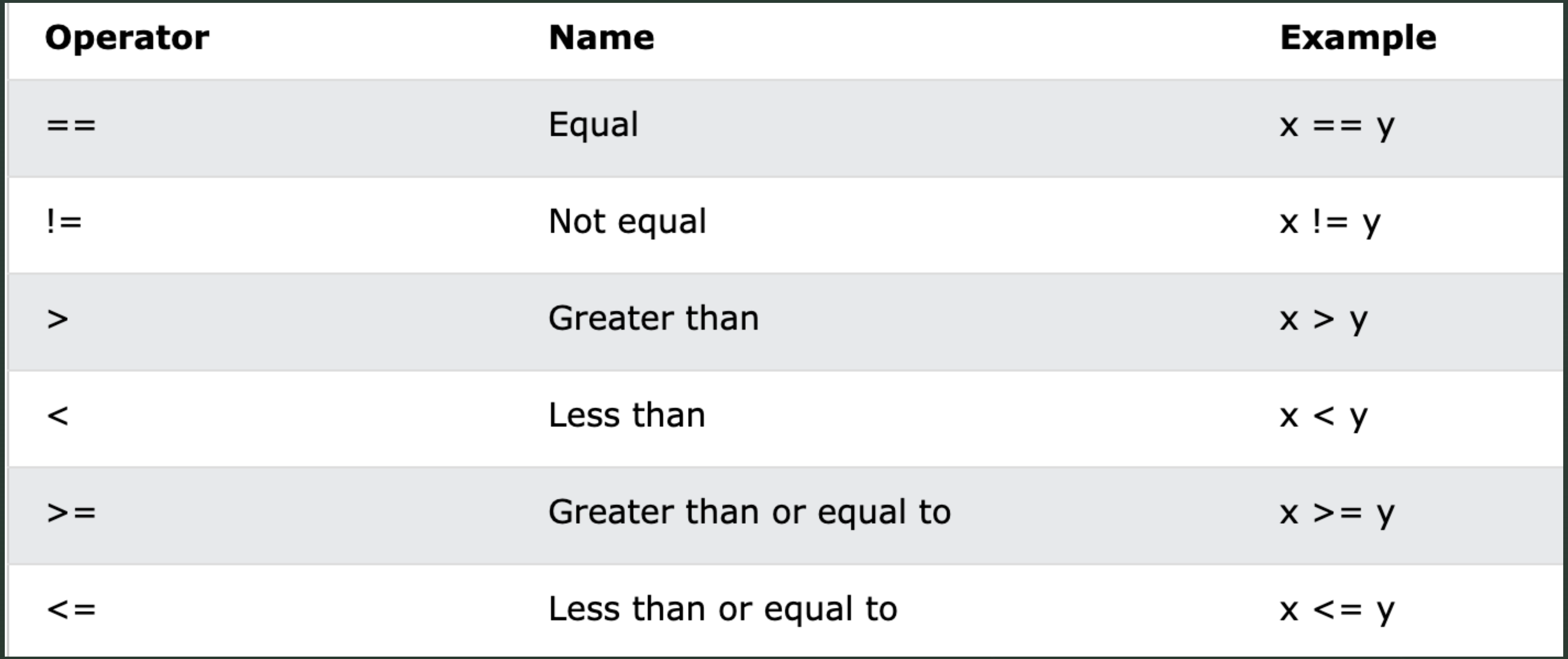
Operator	Name	Example
+	Addition	x + y
-	Subtraction	x - y
*	Multiplication	x * y
/	Division	x / y
%	Modulus	x % y
**	Exponentiation	x ** y
//	Floor division	x // y

Assignment Operators



Operator	Example	Same As
=	x = 5	x = 5
+=	x += 3	x = x + 3
-=	x -= 3	x = x - 3
*=	x *= 3	x = x * 3
/=	x /= 3	x = x / 3
%=	x %= 3	x = x % 3
//=	x //= 3	x = x // 3
**=	x **= 3	x = x ** 3

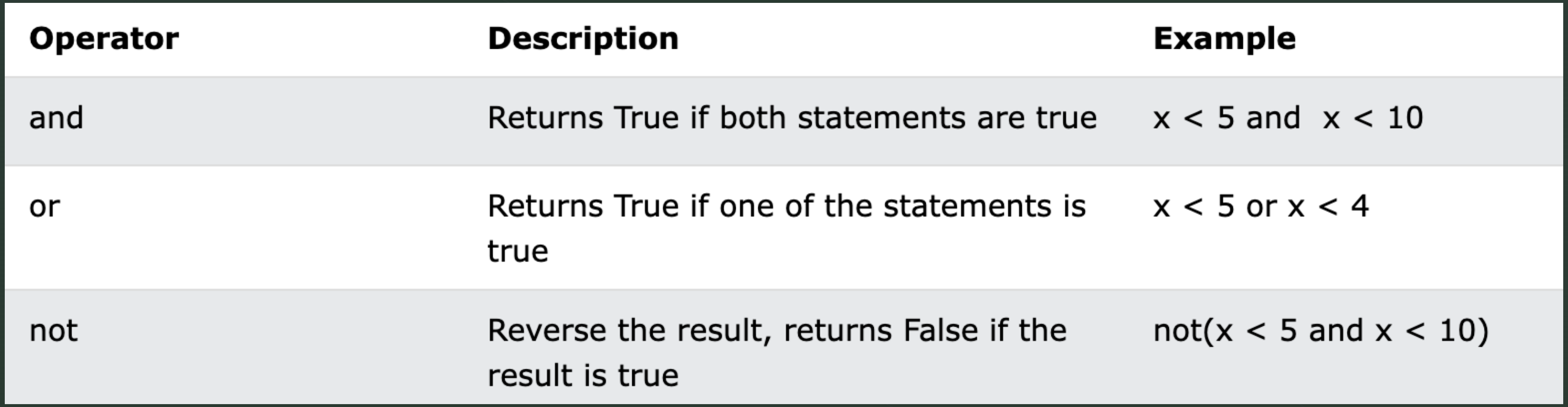
Comparison Operators



Operator	Name	Example
==	Equal	x == y
!=	Not equal	x != y
>	Greater than	x > y
<	Less than	x < y
>=	Greater than or equal to	x >= y
<=	Less than or equal to	x <= y

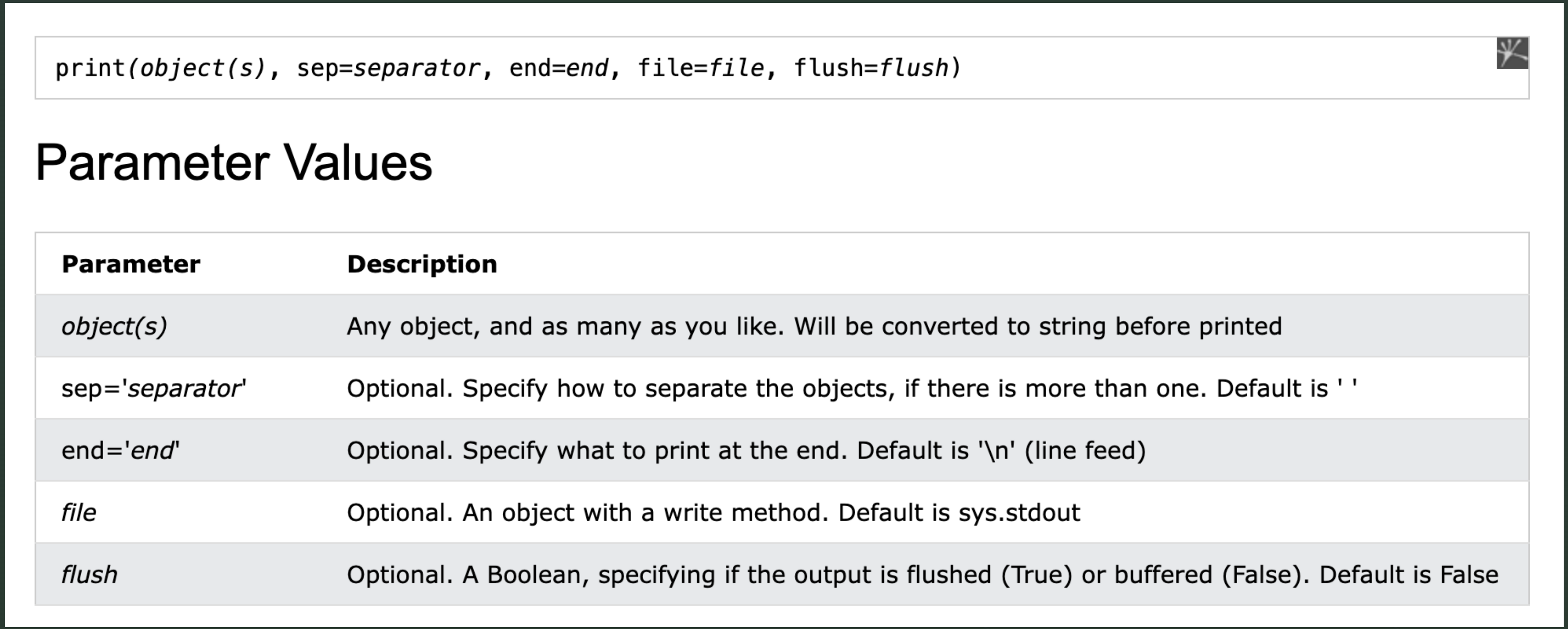
Logical Operators

Logical operators are used to combine conditional statements:



Operator	**Description**	**Example**
and	Returns True if both statements are true	x < 5 and x < 10
or	Returns True if one of the statements is true	x < 5 or x < 4
not	Reverse the result, returns False if the result is true	not(x < 5 and x < 10)

print()



```
print(object(s), sep=separator, end=end, file=file, flush=flush)
```

Parameter Values

Parameter	Description
object(s)	Any object, and as many as you like. Will be converted to string before printed
sep='*separator*'	Optional. Specify how to separate the objects, if there is more than one. Default is ' '
end='*end*'	Optional. Specify what to print at the end. Default is '\n' (line feed)
file	Optional. An object with a write method. Default is sys.stdout
flush	Optional. A Boolean, specifying if the output is flushed (True) or buffered (False). Default is False

Input()

Python allows for user input. That means we are able to ask the user for input.

```
#The input() always return a STRING Value even when the
user inputs numbers

          username = input("Enter username:")
          print("Username is: " + username)
```