
Python Programming
Basics

Python

- In this course you will learn:
  - Python basics
  - Functions and Modules
  - Math
  - Python Strings
  - Iterables, Sequences, Dictionaries and Sets
  - Flow Control

Python Basics

- How Python works.
- Python’s place in the world of programming languages.
- Python literals.
- Python comments.
- Variables and Python data types.
- Simple modules.
- Outputting data with print().
- Collecting user input.

Introduction

- Guido van Rossum, 1991
- webdevelopment (server-side)
- software development
- artificial intelligence
- system scripting
- prototyping

Python2 vs Python3

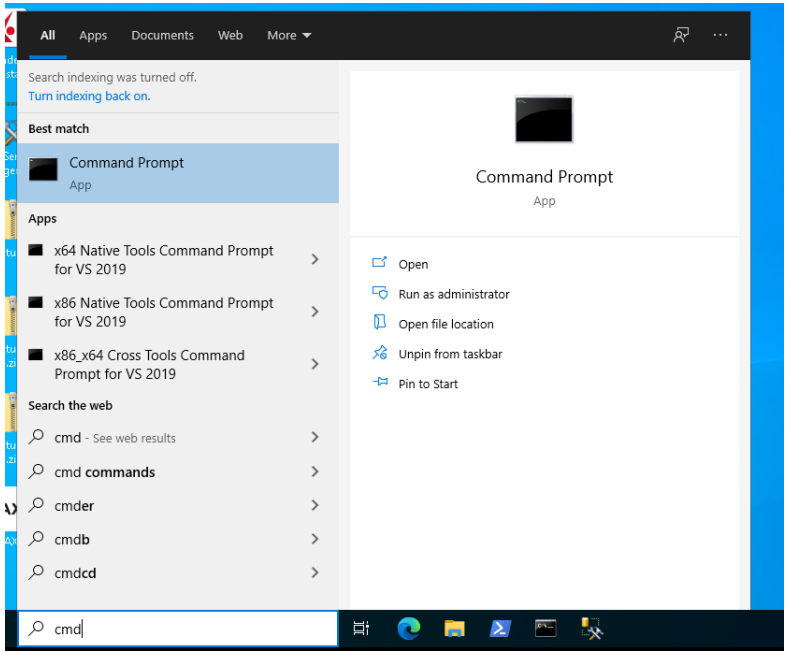
- The most recent major version of Python is Python 3, which we will use in this course. However, Python 2 is still quite popular, although it won't be updated with anything other than security updates.

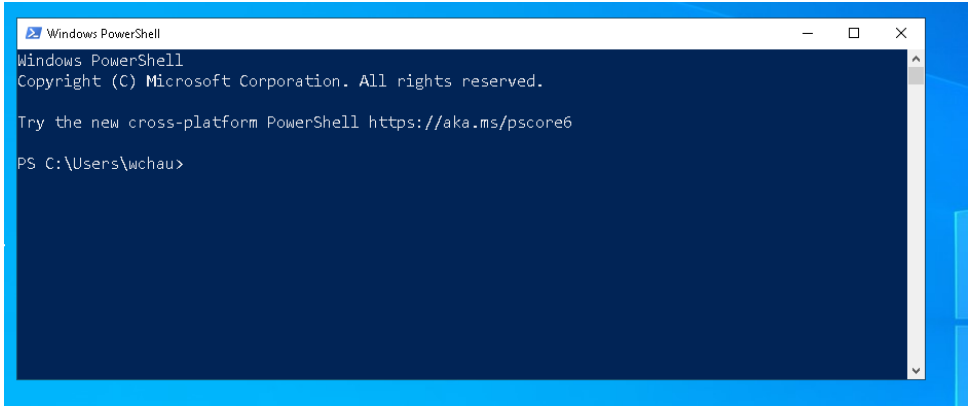
The Python Way

- Beautiful is better than ugly.
- Explicit is better than implicit.
- Simple is better than complex.
- Readability counts.

From: The Zen of Python

The Terminal/Command Prompt/Powershell





Running Python in the terminal

```
> python3 --version

Python 3.9.5


> python3 <filename>
```

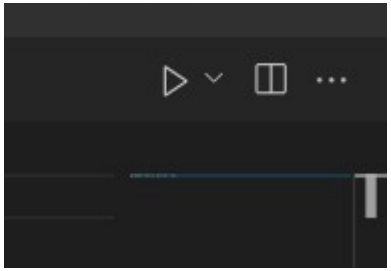
Python Interactive Shell

```
> python3
Python 3.9.5 (default, May 11 2021, 08:20:37)
[GCC 10.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

exit the shell with quit() or ctrl+D

Running Python from Visual Studio Code

- press CTRL+F5 or press the ‘play’ button at the right top
- demo: hello_world.py



Installation + Exercise 1

- If you have not yet installed Visual Studio Code:
  - Install VS Code (see VS Code Installation Document)
- If you have not yet installed Python:
  - Install Python (see Python Installation Document)
- exercise 1: Hello World!
  - 5-10 min

Python Basics

- Literals

```
"hello"
```

```
42
```

- Python Comments (single line and multiline)

```
#
```

```
"""
```

Python Basics

- Data Types
  - boolean (bool)
  - integer (int)
  - float (float)
  - string (str)
  - list (list)
  - tuple (tuple)
  - dictionary (dict)
  - set (set)

Exercise 2

- exploring types
  - 10-15 min

Variables

```
x = 1

woord = "Hallo"
```

- Variable names
  - case sensitive
  - must begin with letter or underscore (_)
  - may only contain letters, digits, underscores

Keywords

False	await	else	import	pass
None	break	except	in	raise
True	class	finally	is	return
and	continue	for	lambda	try
as	def	from	nonlocal	while
assert	del	global	not	with
async	elif	if	or	yield

Variable assignment

- variable name
- assignment operator
- value assigned

```
greeting = "Hello World!"
```

- demo 2: hello_variables.py

Simultaneous assignment

```
var_name1, var_name2 = value1, value2

voornaam, achternaam, leeftijd = "Guido", "van Rossem", 50
```

Exercise 3

- A simple Python Script

Constants

- Python doesn’t have constants but as a convention variables that are meant to act like constants are written in capital letters

```
PI = 3.14
```

Deleting variables

- deleting variables can be done with del

```
x = 1

del x
```

Modules

- a Python module is simply a python file

the main() function

- demo 3: indent_demo.py
- functions are created using def keyword
- function is not executed until called
- visualization demo: http://bit.ly/pythontutor-indentdemo

Functions

1. Functions are created using the def keyword. The content that follows the def keyword on the same line is called the function signature.

2. The convention for naming functions is the same as that for variables: use all lowercase letters and separate words with underscores.

3. In the function definition, the function name is followed by a pair of parentheses, which may contain parameters (more on that soon), and a colon.

4. The contents of the function starts on the next line and must be indented. Either spaces or tabs can be used for indenting, but spaces are preferred.

5. The first line of code after the function definition that is not indented is not part of the function and effectively marks the end of the function definition.

6. Functions are invoked using the function name followed by the parentheses (e.g., indent_demo()).

print() function

- can take multiple arguments

```
print('H', 'e', 'l', 'l', 'o', '!')
```

- output with spaces
- demo 4: variable_and_string_output.py
- named arguments:

```
print('H', 'e', 'l', 'l', 'o', '!', sep=' ', end='\n')
```

- demo 5: variable_and_string_output_fixed_spacing.py
  - get rid of the extra space before the end

Collecting user input

- get user input with input()
- takes 1 parameter
- demo 6: input.py

Exercise 4

- Hello, You!
  - 5-10 min

reading from and writing to files

- reading from a file
  - open a file and assign file handler
  - read content into a variable
  - print variable
  - close file

```
f = open("my-file.txt") # Open my-file.txt and assign result to f.
content = f.read() # Read contents of file into content variable.
print(content) # Print content.
f.close() # Close the file.
```

closing the file

- It is important to close the file to free up the memory space the handler is taking up.
- Python provides a structure that makes explicitly closing the file unnecessary:

```
with open("my-file.txt") as f:
    content = f.read()
    print(content)
```

writing to a file

- open() takes a second parameter:
  - "r" read (default)
  - "w" write
  - "a" append
- to overwrite (or create) a file:

```
with open("my-file2.txt", "w") as f:
    f.write("Hello, world!!!!!")
```

exercise 5

- working with files
  - 10-15 min
- note:
- By default, VS Code will run the file from the Workspace root folder
  - you may get FileNotFoundError errors.
  - to prevent this check the following setting in VS code:
  - Python > Terminal: Execute in File Dirs