# PROGRAMMING WITH PYTHON CRASH COURSE

Fabio Pacifici
Full Stack Developer
www.fabiopacifici.com

# Programming with Python 3

## THE E-BOOK

Version 1.0.1 | 22-01-2022

Fabio Pacifici



## Free Video Serie

This e-Book comes with a free video serie for those who prefer to watch instead of reading. You can find a link to the video under each chapter. The entire vide course is available here

# Preface

I turned 40 in January 2020. It was the prelude of the biggest global crisis that the world has ever seen in the past century. I'll never forget it. I am Italian but, at the time of writing this preface, I am living in the UK. This e-book comes from a side project that I started back in March 2020 when Italy was the first county in the world to go under full lockdown.

In the UK the government wasn't taking the covid19 seriously yet and so, no restrictions were in place. During such tough times, I thought it was nice to show some support to my home country and do something to help folks to stay at home and spend some quality time.

It was back then that I started recording my very first video course on Python in Italian, which I released for free on YouTube. Alongside it, I wrote a blog post for every lesson I recorder to give to the user also something to read as reference for the future.

Not much later, also the UK went into lockdown. I kept recording lessons to fill up my time to until at the end of the course I realised that it was time to start recording also for the British people so, I recorded an in-depth course on Python 3 in English too and wrote a blog post for it too.

This book comes from that experience and, it's meant to be a portable reference to the Python 3 Crash Course 2020 that I recorded. You can refer to it even without an internet connection.

I wrote this book but first place I recorder the video courses because I noticed that often programming languages have documentation, books and blog posts that are difficult to understand for newcomers. Lot's of none sense function names, like foo and bar, a lack of real-world examples and many hard to understand concepts that would make even a seasoned developer sweat cold.

This e-book aims to explain in simple English concepts that span from a basic to an advanced level using real-world examples.

I hope you enjoy it and find it easy to understand.

## Who am I

I am Fabio, a senior Fullstack developer and teacher, an entrepreneur with over a 15 years of experience in programming using different languages and technologies. Formerly a martial arts teacher and a passionate BMX rider.

## Thank you

I dedicate this e-book to my lovely wify Serena, she supports me in everything I do. She is my moon and sky, my best friend, fan and supporter. She also cooks for me delicious food while I am focused 100% on coding, teaching and writing technical blog posts and e-books. Serena, you are the best!

## Why Python

Python is a general-purpose programming language, it's easy to read and worth learning. It can be used to build different types of programs from web-based applications to software with Graphical user interfaces, command-line applications, Machine learning and IT Automation.

During the course we will cover the basics but also dig into more advenced concepts like files manipulation, object oriented programming and testing.

# Topics

0. Course Introduction A quick introduction to the Programming with Python Crash Course
1. Environment Setup Setup our development environment, Install Python (Windows/Linux/macOS), Install VSCode and the code runner extension.
2. Variables Learn about variables in Python. How to define a valid variable and invalid variables errors
3. Strings In this module, we will study strings, one of the Python basic data types.
4. Numbers and Booleans In this module, we will study two more basic data types, Numbers and Booleans in Python.
5. Lists In this module, we will cover the basic data type called List, and how to use them.
6. Dictionaries In this module, we will cover the basic data type called Dictionary, and how to use them.
7. Tuples In this module, we will cover the basic data type called Tuple, and how to use them.
8. Conditionals In this module we will learn a powerful concept in python programming, and how to make our code behave differently based on a given condition.
9. Loops In the loops module we will cover how to write programs to repeat certain tasks avoiding code duplication.
10. List Comprehension A shot and practical alternative of writing lists with the resulting values of loops and conditionals
11. Functions In this module we will learn how to write clean and reusable code by grouping it into reusable function.
12. Object-Oriented Programming (Classes) This module covers the fundamentals of OOP in Python, from classes definition to modules and class attributes.
13. Object-Oriented Programming (Inheritance) In this module we will keep working with classes and learn how to define child classes and use special methods
14. Files and CSV Manipulation In this module we will cover techniques to manipulate files and have a quick look at the CSV module

15. Testing The final Module of the course covers Testing in Python, including practical examples and a general overview around the different types of tests.

# Environment Setup

Watch the video lesson on YouTube

Let’s see how to install Python and a couple of other tools that we will use during the course.

## Windows Installation

To install python on windows, visit the website python.org, click the download button to get the installer then open it and follow the instructions. Make sure to check the box during "Add python to your path" during the installation.

Once the installation is completed open the windows powershell and type

```
python -V
```

## Install on Linux/MacOS

Linux distributions comes with python 2 and 3 preinstalled. to verify that run the following in the terminal

```
python -v
python3 -v
```

The same applies to Mac OS.

Run Python scripts from the interpreter: run in the terminal

```
python
```

or python3 on linux to start the interactive interpreter.

```
python3
```

Run python on a file:

To do this I assume that you already use some sort of text editor or IDE like sublime, Athom or VisualStudio code. If not I reccomend you to use VisualStudio code becase it's what I will be using during the course.

You can download it at https://code.visualstudio.com/, it's free.

Once downloaded and installed, start the program and click the extensions tab and install the extention "code runner"

Now that your environment is ready, let's get started with Python programming.

# Variables

Watch the video lesson on YouTube The first building block of most programming languages. Variables.

The first concept that we will cover is Variables. If you already master other programming languages this is nothing new to you. A Variable is a word that we define and use to store information in our program for later use. It can only start with letters and underscore, It can contain numbers and underscore. It cannot start with several other characters and cannot contain characters that are not numbers, letters or underscore.

```
# Valid variables:
_name = "Fabio"
name = "Fabio"
name_1 = "Fabio"
# Invalid:
-name = "Fabio"
9_name = "Fabio"
name! = "Fabio"
```

# Strings

Watch the video lesson on My YouTube channel:
https://youtu.be/TFeVqVBVN9Y

Strings are sequences of characters. We can write strings in single, double or triple quotes.

```
'This is a python crash course'
"This is a python crash course"
"""This is a python crash course"""
```

We can assign stings to a variable and concatenate them using the plus sign `+` or the `.format()` method.

```
name = "Fabio"
profession = "Developer"
# String contatenation with the plus operator
sentence = name + " is a " + profession
# String concatenation with the format() method
sentence = "{} is a {}".format(name, profession)
```

We can assign a string to a variable and access single elements or part of the string using square brackets and the slice method.

**Access a single character:**

```
name = "Fabio"
name[0]
Using the Slice method:
name[2:]
name[:2]
```

**String Indexing:**

We can get the position of an element of the string (character) using the index method

```
name.index("b")
```

**Strings are Immutable:**

If we try to access a character of a string and assign to it a new value we will get an error. We cannot mutate a string.

```
name[0] = "B"
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'str' object does not support item
assignment
```

However we can create a new string and replaca the character like so to obtain the same result we wanted in the code above.

```
new_name = "B" + name[1:]
print(new_name)
'Babio' # <-- Result
```

# Numbers and Booleans

Watch the video lesson on my YouTube channel https://youtu.be/pIkldpqOtBM Numbers are another basic data type in Python. There are two types of numbers. Integers and float.

```
15 # integer
15.5 # float
We can use numbers to perform calculations.
# Sum
5+2
# Subtraction
5-2
# Moltiplication
5*2
# Division
10/2

# Exponenet
5**2
# Modulo
16%4
# Floor division
15//4
```

Like in math operations inside parentheses take precedence. Then the exponent, multiplication and division, modulo and floor and finally additions and subtractions.

## Booleans

A boolean value is either True or False. We use it to make decisions in our code based on a condition.

```
is_risky = True
is_safe = False
```

# Lists

Watch the video lesson: https://youtu.be/DwUxxEiYtmY

A list is a sequence of elements. usually of the same type but it can contain different elements. To define a list we use square brackets and separate each element with a comma.

> ***Attention**: when you see the following symbol `>>>` in a code block I am executing the code inside a Python interpreter. Instead, when there is no symbol before the line of code, the code is written in a Python file. If you follow along make sure not to copy the `>>>` in your code otherwise it won't work.*

```
[1, 3, 5 , 6 , 7]
["Developer", "Writer", "Student", "Teacher"]
```

## Access a list element

We can access elements in a list by assigning it to a variable as we did with strings. Then we can use square brackets and the index method in the same way.

```
>>> jobs = ["Developer", "Writer", "teacher"]
>>> jobs[0]
'Developer'
>>> jobs[1:]
['Writer', 'teacher']
>>> jobs[:2]
['Developer', 'Writer']
>>> jobs[0:2]
['Developer', 'Writer']
```

```
>>> jobs.index("Writer")
1
```

## Add elements

We can do a lot more with lists, like add an element to it using the append method. It adds it at the end of the list.

```
>>> jobs.append("Front-end engineer")
>>> jobs
['Developer', 'Writer', 'teacher', 'Front-end
engineer']
>>>
```

## Insert an element

we can also insert an element in a specific position. The first argument is the index where we want to insert the element, it’s an integer. if the integer is a number greater than the length of items of the list the new element will be inserted at the end.

```
>>> jobs.insert(0, "SEO")
>>> jobs.insert(10, "Back-end developer")
>>> jobs
['SEO', 'Developer', 'Writer', 'teacher', 'Front-
end engineer', 'Back-end developer']
>>>
```