PYTHON *from the very beginning*

In *Python from the Very Beginning* John Whitington takes a no-prerequisites approach to teaching a modern general-purpose programming language. Each small, self-contained chapter introduces a new topic, building until the reader can write quite substantial programs. There are plenty of questions and, crucially, worked answers and hints.

Python from the Very Beginning will appeal both to new programmers, and to experienced programmers eager to explore a new language. It is suitable both for formal use within an undergraduate or graduate curriculum, and for the interested amateur.

JOHN WHITINGTON founded a company which sells software for document processing. He taught programming to students of Computer Science at the University of Cambridge. His other books include the textbooks *"PDF Explained"* (O'Reilly, 2012), *"OCaml from the Very Beginning"* (Coherent, 2013), and *"Haskell from the Very Beginning"* (Coherent, 2019) and the Popular Science book *"A Machine Made this Book: Ten Sketches of Computer Science"* (Coherent, 2016).

Python

from the very beginning

John Whitington

COHERENT PRESS

COHERENT PRESS
Cambridge

Published in the United Kingdom by Coherent Press, Cambridge

First published October 2020

A catalogue record for this book is available from the British Library

ISBN 978-0-9576711-5-7 Paperback

by the same author

PDF Explained (O'Reilly, 2012)
OCaml from the Very Beginning (Coherent, 2013)
More OCaml: Methods, Algorithms & Diversions (Coherent, 2014)
A Machine Made this Book: Ten Sketches of Computer Science (Coherent, 2016)
Haskell from the Very Beginning (Coherent, 2019)

Contents

Preface

I have tried to write a book which has no prerequisites – and with which any intelligent person ought to be able to cope, whilst trying to be concise enough that a programmer experienced in another language might not be too annoyed by the pace or tone.

This may well not be the last book you read on Python, but one of the joys of Python is that substantial, useful programs can be constructed quickly from a relatively small set of constructs. There is enough in this book to build such useful programs, as we see in the four extended projects.

Answers and Hints are at the back of the book.

Chapters

In chapter 1 we begin our exploration of Python with a series of preliminaries, introducing ways to calculate with the whole numbers, compare them with one another, and print them out. We learn about *truth values* and the other types of simple data which Python supports.

In chapter 2 we build little Python programs of our own, using *functions* to perform calculations based on changing inputs. We make decisions using conditional constructs to choose differing courses of action.

In chapter 3 we learn about Python constructs which perform actions repeatedly, for a fixed number of times or until a certain condition is met. We start to build larger, more useful programs, including interactive ones which depend upon input from the keyboard.

In chapter 4 we begin to build and manipulate larger pieces of data by combining things into lists, and querying and processing them. This increases considerably the scope of programs we can write.

In chapter 5 we expand our work with lists to manipulate strings, splitting them into words, processing them, and putting them back together. We learn how to sort lists into order, and how to build lists from scratch using *list comprehensions*.

In chapter 6 we learn more about printing messages and data to the screen, and use this knowledge to print nicely-formatted tables of data. We learn how to write such data to a file on the computer, instead of to the screen.

In chapter 7 we learn another way of storing data – in *dictionaries* which allow us to build little databases, looking up data by searching for it by name. We also work with *sets*, which allow us to store collections of data without repetition, in the same way as mathematical sets do.

In chapter 8 we deal with the thorny topic of errors: what do we do when an input is unexpected? When we find a number when we were expecting a list? When an item is not found in a dictionary? We learn how to report, handle, and recover from these errors.

In chapter 9 we return to the subject of files, learning how to read from them as well as write to them, and illustrate with a word counting program. We deal with errors, such as the unexpected absence of a file.

In chapter 10 we talk about real numbers, which we have avoided thus far. We show how to calculate with the trigonometric functions and how to convert between whole and real numbers by rounding.

In chapter 11 we introduce the Python Standard Library, greatly expanding the pre-built components at our disposal. We learn how to look up functions in Python's official documentation.

In chapter 12 we build stand-alone programs which can be run from the command line, as if they were built in to the computer. We are now ready to begin on larger projects.

Projects

In project 1 we draw all sorts of pretty pictures by giving the computer instructions on how to draw them line by line. We make a graph plotter and a visual clock program.

In project 2 we write a calorie-counting program which stores its data across several files, and which allows multiple users. We build an interface for it, with several different commands. We learn how to use a standard data format, so that spreadsheet programs can load our calorie data.

In project 3 we investigate the childhood game of Noughts and Crosses, writing human and computer players, and working out some statistical properties of the game by building a structure containing all possible games.

In project 4 we learn how to manipulate photographs, turning them into greyscale, blurring them, and making animations from them.

Online resources

To save typing, all the examples and exercises for this book can be found in electronic form at `https://pythonfromtheverybeginning.com`. The book's errata lives there too.

Acknowledgements

The technical reviewer provided valuable corrections and suggestions, but all mistakes remain the author's. The image on page 147 is from "Flexible Strategy Use in Young Children's Tic-Tac-Toe" by Kevin Crowley and Robert S. Siegler, and is reproduced courtesy of Elsevier.

Getting Ready

This book is about teaching the computer to do new things by writing computer programs. Just as there are different languages for humans to speak to one another, there are different *programming languages* for humans to speak to computers.

We are going to be using a programming language called **Python**. A Python system might already be on your computer, or you may have to find it on the internet and install it yourself. You will know that you have it working when you see something like this:

```
Python 3.8.2 (default, Feb 24 2020, 18:27:02)
[Clang 10.0.1 (clang-1001.0.46.4)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Make sure the Python version number in the first line, here Python 3.8.2, is at least `3`. You might need to type `python3` instead of `python` to achieve this. Python is waiting for us to type something. Try typing 1 space + space 2 followed by the Enter key. You should see this:

```
Python
>>> 1 + 2
3
>>>
```

(We have abbreviated Python's welcome message). Python tells us the result of the calculation. You may use the left and right arrow keys on the keyboard to correct mistakes and the up and down arrow keys to look through a history of previous inputs. You can also use your computer's usual copy and paste functions, instead of typing directly into Python, if you like.

To abandon typing, and ask Python to forget what you have already typed, enter Ctrl-C (hold down the Ctrl key and tap the c key). This will allow you to start again. To leave Python altogether, give the `exit()` command, again followed by Enter:

```
Python
>>> exit()
```

You should find yourself back where you were before. We are ready to begin.