

Preface

People, not withstanding caste, creed, gender, ethnic diversities, nationalities, are interacting intensely in the recent decades identifying commonalities, accommodating differences, making common cause. Python stands out as a shining outcome of such distributed but focused co-ordination. It started with an idea—‘Simplicity at lofty heights (*my view*)’—that occurred to Guido van Rossum, who continues to be the accepted *benevolent dictator for life* (BDFL) for Python community. It is not that anyone can join this bandwagon and contribute; as it is not that easy. You can suggest a contribution but its pros and cons are discussed in an open forum through the net and (in the accepted shape) it enters the ‘Holy Book’ as PEP (Python Enhancement Proposal). The (open) Holy Book continues to grow in size shedding better light. It is a thrill to know how well it is evolving and to ‘feel’ or participate in its lustre. Python shines with the layers for its use—simple for the novice, versatile for the programmer, added facilities for the developer, openness for a ‘Python sculptor’. It has a varied and versatile data structure, a vast library, a huge collection of additional resources, and above all OPENNESS. So embrace Python—the language by the people, of the people, for the people.

Definitely this is not justification enough for another book on Python. The variety of data structures and the flexibility and vastness of the modules in the Python library are daunting. The most common features of Python have been dealt with in this book bringing out their subtleties; their potential and suitability for varied use through illustrations. Nothing is glossed over. One can go through the illustrative examples, repeat them in toto, or run their variants at one’s own pace and progress. The matter has been presented in a logical and graded manner. Some of the exercises at the ends of chapters are pedagogical. But many of them call for more efforts—perhaps candidates for minor projects. Concepts associated with constructs like *yield*, *iterator*, *generator*, *decorator*, *super* (inheritance), *format* (Python 3) are often considered to be abstract and difficult to digest. A conscious effort has been made to explain these through apt examples. The associated exercises complement these in different ways. Any feedback by way of corrections, clarifications, or any queries are welcome (blog: nahtap.blogspot.com).

I am grateful to Prof. K. Gangadharan of Amrita University to have opened my eyes to the openness of open systems. This book is an offshoot of this. In many ways, I am indebted to my students and colleagues over the decades; discussions with them, often spurred by a query, have been immensely helpful in honing my understanding and clarifying concepts. Implicitly the same is reflected in the book as well. I thank Suvira Srivastav and Praveen Kumar for steering the book through the Processes in Springer.

Lastly (but not priority wise) my thanks are due to my wife Uma for her unwavering and sustained accommodation of my oddities.

Coimbatore, India

T.R. Padmanabhan



<http://www.springer.com/978-981-10-3276-9>

Programming with Python

Padmanabhan, T.R.

2016, XIII, 344 p. 207 illus., Hardcover

ISBN: 978-981-10-3276-9