

Practical Programming^Second Edition

An Introduction to Computer Science Using python 3

Paul Gries
Jennifer Campbell
Jason Montojo

The Pragmatic Bookshelf

Dallas, Texas • Raleigh, North Carolina

Contents

Acknowledgments	xi
Preface	xiii
1. What's Programming?	1
1.1 Programs and Programming	2
1.2 What's a Programming Language?	3
1.3 What's a Bug?	4
1.4 The Difference Between Brackets, Braces, and Parentheses	5
1.5 Installing Python	5
2. Hello, Python	7
2.1 How Does a Computer Run a Python Program?	7
2.2 Expressions and Values: Arithmetic in Python	9
2.3 What Is a Type?	12
2.4 Variables and Computer Memory: Remembering Values	15
2.5 How Python Tells You Something Went Wrong	22
2.6 A Single Statement That Spans Multiple Lines	23
2.7 Describing Code	25
2.8 Making Code Readable	26
2.9 The Object of This Chapter	27
2.10 Exercises	27
3. Designing and Using Functions	31
3.1 Functions That Python Provides	31
3.2 Memory Addresses: How Python Keeps Track of Values	34
3.3 Defining Our Own Functions	35
3.4 Using Local Variables for Temporary Storage	39
3.5 Tracing Function Calls in the Memory Model	40
3.6 Designing New Functions: A Recipe	47
3.7 Writing and Running a Program	59

3.8	Omitting a Return Statement: None	60
3.9	Dealing with Situations That Your Code Doesn't Handle	61
3.10	What Did You Call That?	62
3.11	Exercises	63
4.	Working with Text	65
4.1	Creating Strings of Characters	65
4.2	Using Special Characters in Strings	68
4.3	Creating a Multiline String	70
4.4	Printing Information	70
4.5	Getting Information from the Keyboard	73
4.6	Quotes About Strings in This Text	74
4.7	Exercises	75
5.	Making Choices	77
5.1	A Boolean Type	77
5.2	Choosing Which Statements to Execute	86
5.3	Nested If Statements	92
5.4	Remembering the Results of a Boolean Expression Evaluation	92
5.5	You Learned About Booleans: True or False?	94
5.6	Exercises	94
6.	A Modular Approach to Program Organization	99
6.1	Importing Modules	100
6.2	Defining Your Own Modules	104
6.3	Testing Your Code Semiautomatically	109
6.4	Tips for Grouping Your Functions	112
6.5	Organizing Our Thoughts	113
6.6	Exercises	114
7.	Using Methods	115
7.1	Modules, Classes, and Methods	115
7.2	Calling Methods the Object-Oriented Way	117
7.3	Exploring String Methods	119
7.4	What Are Those Underscores?	123
7.5	A Methodical Review	125
7.6	Exercises	125
8.	Storing Collections of Data Using Lists	129
8.1	Storing and Accessing Data in Lists	129
8.2	Modifying Lists	133

8.3	Operations on Lists	134
8.4	Slicing Lists	137
8.5	Aliasing: What's in a Name?	138
8.6	List Methods	140
8.7	Working with a List of Lists	142
8.8	A Summary List	144
8.9	Exercises	144
9.	Repeating Code Using Loops	147
9.1	Processing Items in a List	147
9.2	Processing Characters in Strings	149
9.3	Looping Over a Range of Numbers	150
9.4	Processing Lists Using Indices	152
9.5	Nesting Loops in Loops	154
9.6	Looping Until a Condition Is Reached	158
9.7	Repetition Based on User Input	161
9.8	Controlling Loops Using Break and Continue	161
9.9	Repeating What You've Learned	165
9.10	Exercises	166
10.	Reading and Writing Files	171
10.1	What Kinds of Files Are There?	171
10.2	Opening a File	173
10.3	Techniques for Reading Files	176
10.4	Files over the Internet	181
10.5	Writing Files	182
10.6	Writing Algorithms That Use the File-Reading Techniques	183
10.7	Multiline Records	191
10.8	Looking Ahead	194
10.9	Notes to File Away	196
10.10	Exercises	197
11.	Storing Data Using Other Collection Types	199
11.1	Storing Data Using Sets	199
11.2	Storing Data Using Tuples	204
11.3	Storing Data Using Dictionaries	209
11.4	Inverting a Dictionary	216
11.5	Using the In Operator on Tuples, Sets, and Dictionaries	217
11.6	Comparing Collections	218

11.7	A Collection of New Information	218
11.8	Exercises	* 219
12.	Designing Algorithms	223
12.1	Searching for the Smallest Values	224
12.2	Timing the Functions	232
12.3	At a Minimum, You Saw This	234
12.4	Exercises	234
13.	Searching and Sorting	237
13.1	Searching a List	237
13.2	Binary Search	245
13.3	Sorting	249
13.4	More Efficient Sorting Algorithms	259
13.5	Mergesort: A Faster Sorting Algorithm	261
13.6	Sorting Out What You Learned	265
13.7	Exercises	266
14.	Object-Oriented Programming	269
14.1	Understanding a Problem Domain	270
14.2	Function "Isinstance," Class Object, and Class Book	271
14.3	Writing a Method in Class Book	274
14.4	Plugging into Python Syntax: More Special Methods	280
14.5	A Little Bit of OO Theory	282
14.6	A Case Study: Molecules, Atoms, and PDB Files	288
14.7	Classifying What You've Learned	292
14.8	Exercises	293
15.	Testing and Debugging	297
15.1	Why Do You Need to Test?	297
15.2	Case Study: Testing above_freezing	298
15.3	Case Study: Testing running_sum	304
15.4	Choosing Test Cases	310
15.5	Hunting Bugs	311
15.6	Bugs We've Put in Your Ear	312
15.7	Exercises	312
16.	Creating Graphical User Interfaces	317
16.1	Using Module Tkinter	317
16.2	Building a Basic GUI	319
16.3	Models, Views, and Controllers, Oh My!	323
16.4	Customizing the Visual Style	327

16.5	Introducing a Few More Widgets	332
16.6	Object-Oriented GUIs	335
16.7	Keeping the Concepts from Being a GUI Mess	336
16.8	Exercises	336
17.	Databases	339
17.1	Overview	339
17.2	Creating and Populating	340
17.3	Retrieving Data	344
17.4	Updating and Deleting	347
17.5	Using NULL for Missing Data	348
17.6	Using Joins to Combine Tables	349
17.7	Keys and Constraints	353
17.8	Advanced Features	354
17.9	Some Data Based On What You Learned	360
17.10	Exercises	361
	Bibliography	365
	Index	367