

Detailed Contents

Preface	<i>page</i> xvii
To the Student	xxiii
Notices and Disclaimers	xxvi
Acknowledgments	xxix
Part I Getting Basic Tasks Done	1
1 Prologue: Preparing to Program	3
1.1 What Is a Program and Why Learn to Program?	3
1.2 What Is Python and Why Learn This Language?	5
1.3 Software We Will Need	6
2 Python as a Basic Calculator	8
2.1 Example of Python as a Basic Calculator	8
2.2 Python Programming Essentials	10
2.2.1 Expressions and Operators	10
2.2.2 Variables	13
2.2.3 The Python Interpreter	15
2.3 Try This!	18
2.4 More Discipline-Specific Practice	24
2.5 Chapter Review	24
2.5.1 Self-Test Questions	24
2.5.2 Chapter Summary	25
2.5.3 Self-Test Answers	26
3 Python as a Scientific Calculator	27
3.1 Example of Python as a Scientific Calculator	27
3.2 Python Programming Essentials	28
3.2.1 Using Prewritten Functions	29
3.2.2 Importing Modules and Using Module Items	30
3.2.3 Writing and Using Our Own Functions	32
3.2.4 A Programmable Calculator	35
3.2.5 Python Interpreter and Code-Writing Environments for More Complex Programs	38
3.3 Try This!	41

3.4	More Discipline-Specific Practice	47
3.5	Chapter Review	47
3.5.1	Self-Test Questions	47
3.5.2	Chapter Summary	48
3.5.3	Self-Test Answers	49
4	Basic Line and Scatter Plots	52
4.1	Example of Making Basic Line and Scatter Plots	52
4.2	Python Programming Essentials	54
4.2.1	Positional Input Parameters for Required Input	55
4.2.2	Introduction to Lists and Tuples	58
4.2.3	Introduction to Strings	62
4.2.4	Introduction to Commenting and Jupyter Markdown	66
4.3	Try This!	69
4.4	More Discipline-Specific Practice	81
4.5	Chapter Review	81
4.5.1	Self-Test Questions	81
4.5.2	Chapter Summary	83
4.5.3	Self-Test Answers	85
5	Customized Line and Scatter Plots	88
5.1	Example of Customizing Line Plots	88
5.2	Python Programming Essentials	91
5.2.1	Optional Input into Functions Using Keyword Input Parameters	91
5.2.2	Customizing How the Plot Looks	93
5.2.3	Handling Multiple Figures or Curves	96
5.2.4	Adjusting the Plot Size	97
5.2.5	Saving Figures to a File	98
5.2.6	Introduction to Array Calculations	99
5.2.7	The Concept of Typing	103
5.3	Try This!	106
5.4	More Discipline-Specific Practice	117
5.5	Chapter Review	117
5.5.1	Self-Test Questions	117
5.5.2	Chapter Summary	119
5.5.3	Self-Test Answers	121
6	Basic Diagnostic Data Analysis	124
6.1	Example of Basic Diagnostic Data Analysis	124
6.2	Python Programming Essentials	126
6.2.1	More on Creating Arrays and Inquiring about Arrays	128
6.2.2	More on Functions on Arrays	132

6.2.3	Going Through Array Elements and an Introduction to Loops	134
6.2.4	Introduction to Asking Questions of Data and Branching	139
6.2.5	Examples of One-Dimensional Loops and Branching	148
6.2.6	Docstrings	153
6.2.7	Three Tips on Writing Code	155
6.3	Try This!	158
6.4	More Discipline-Specific Practice	170
6.5	Chapter Review	170
6.5.1	Self-Test Questions	170
6.5.2	Chapter Summary	171
6.5.3	Self-Test Answers	173
7	Two-Dimensional Diagnostic Data Analysis	176
7.1	Example of Two-Dimensional Diagnostic Data Analysis	176
7.2	Python Programming Essentials	182
7.2.1	The Shape of Two-Dimensional Arrays	183
7.2.2	Creating Two-Dimensional Arrays	184
7.2.3	Accessing, Setting, and Slicing in a Two-Dimensional Array	186
7.2.4	Array Syntax and Functions in Two-Dimensional Arrays	190
7.2.5	Nested <code>for</code> Loops	191
7.3	Try This!	194
7.4	More Discipline-Specific Practice	203
7.5	Chapter Review	203
7.5.1	Self-Test Questions	203
7.5.2	Chapter Summary	205
7.5.3	Self-Test Answers	207
8	Basic Prognostic Modeling	209
8.1	Example of a Basic Prognostic Model	209
8.2	Python Programming Essentials	217
8.2.1	Random Numbers in Computers	217
8.2.2	Scalar Boolean Type and Expressions	221
8.2.3	Nested Branching	230
8.2.4	Looping an Indefinite Number of Times Using <code>while</code>	232
8.2.5	Making Multiple Subplots	236
8.2.6	More on Nested Loops	237
8.2.7	Conditionals Using Floating-Point Numbers	239
8.3	Try This!	241
8.4	More Discipline-Specific Practice	252
8.5	Chapter Review	252
8.5.1	Self-Test Questions	252
8.5.2	Chapter Summary	254
8.5.3	Self-Test Answers	257

9	Reading In and Writing Out Text Data	261
9.1	Example of Reading In and Writing Out Text Data	262
9.2	Python Programming Essentials	267
9.2.1	Introduction to Objects	268
9.2.2	Arrays as Objects	269
9.2.3	Lists as Objects	277
9.2.4	Strings as Objects	280
9.2.5	Copying Variables, Data, and Objects	286
9.2.6	Reading and Writing Files	290
9.2.7	Catching File Opening and Other Errors	298
9.3	Try This!	300
9.4	More Discipline-Specific Practice	317
9.5	Chapter Review	317
9.5.1	Self-Test Questions	317
9.5.2	Chapter Summary	319
9.5.3	Self-Test Answers	321
10	Managing Files, Directories, and Programs	327
10.1	Example of Managing Files, Directories, and Programs	328
10.2	Python Programming Essentials	331
10.2.1	Filenames, Paths, and the Working Directory	332
10.2.2	Making and Removing Empty Directories	335
10.2.3	Moving and Renaming Files and Directories	337
10.2.4	Copying and Deleting Files and Directories	338
10.2.5	Listing the Contents of a Directory	340
10.2.6	Testing to See What Kind of “File” Something Is	341
10.2.7	Running Non-Python Programs in Python	342
10.3	Try This!	343
10.4	More Discipline-Specific Practice	347
10.5	Chapter Review	347
10.5.1	Self-Test Questions	347
10.5.2	Chapter Summary	348
10.5.3	Self-Test Answers	350
Part II	Doing More Complex Tasks	353
11	Segue: How to Write Programs	355
11.1	From Blank Screen to Program: A Process to Follow	355
11.2	The Importance of Testing	360
11.3	The Importance of Style Conventions	363

12 <i>n</i>-Dimensional Diagnostic Data Analysis	365
12.1 Example of <i>n</i> -Dimensional Diagnostic Data Analysis	365
12.2 Python Programming Essentials	367
12.2.1 The Shape of and Indexing <i>n</i> -Dimensional Arrays	368
12.2.2 Selecting Subarrays from <i>n</i> -Dimensional Arrays	370
12.2.3 Array Syntax and Functions in <i>n</i> -Dimensional Arrays	372
12.2.4 Reshaping <i>n</i> -Dimensional Arrays and Memory Locations of Array Elements	374
12.2.5 Subarrays and Index Offset Operations	376
12.2.6 Triple Nested Loops and Mixing Array Syntax/Selection and Looping	378
12.2.7 Summary Table of Some Array Functions	380
12.3 Try This!	382
12.4 More Discipline-Specific Practice	386
12.5 Chapter Review	387
12.5.1 Self-Test Questions	387
12.5.2 Chapter Summary	389
12.5.3 Self-Test Answers	391
13 Basic Image Processing	394
13.1 Example of Image Processing	394
13.2 Python Programming Essentials	400
13.2.1 Reading, Displaying, and Writing Images in Matplotlib	401
13.2.2 Boolean Arrays	404
13.2.3 Array Syntax and Functions and Asking Questions of Data in Arrays	408
13.2.4 Performance of Looping and Array Syntax and Functions	414
13.2.5 The NumPy <code>reduce</code> Method	416
13.2.6 Looping Through Lists of Objects	417
13.3 Try This!	419
13.4 More Discipline-Specific Practice	430
13.5 Chapter Review	431
13.5.1 Self-Test Questions	431
13.5.2 Chapter Summary	432
13.5.3 Self-Test Answers	434
14 Contour Plots and Animation	439
14.1 Example of Making Contour Plots and Animations	440
14.2 Python Programming Essentials	445
14.2.1 An Introduction to Matplotlib's Object API	446
14.2.2 Line and Shaded Contour Plots	451
14.2.3 Using <code>cartopy</code> to Overlay Maps	453
14.2.4 Basic Animation Using Matplotlib	456
14.2.5 Flexible Functions and Dictionaries	459

14.3	Try This!	465
14.4	More Discipline-Specific Practice	477
14.5	Chapter Review	477
14.5.1	Self-Test Questions	477
14.5.2	Chapter Summary	479
14.5.3	Self-Test Answers	481
15	Handling Missing Data	483
15.1	Example of Handling Missing Data	483
15.2	Python Programming Essentials	487
15.2.1	Approach 1: Define a Data Value as Missing and Process with Boolean Arrays or Expressions	488
15.2.2	Approach 2: Use <i>Series</i> and IEEE NaN Values	490
15.2.3	Approach 3: Use Masked Arrays	492
15.2.4	Which Approach Is Better?	493
15.3	Try This!	494
15.4	More Discipline-Specific Practice	498
15.5	Chapter Review	498
15.5.1	Self-Test Questions	498
15.5.2	Chapter Summary	499
15.5.3	Self-Test Answers	501
Part III	Advanced Programming Concepts	503
16	More Data and Execution Structures	505
16.1	Example of Using More Advanced Data and Execution Structures	505
16.1.1	Solution 1: Explicitly Call Functions and Store Results in Variables	506
16.1.2	Solution 2: Explicitly Call Functions and Store Results in Arrays	507
16.1.3	Solution 3: Explicitly Call Functions and Store Results in Dictionaries	508
16.1.4	Solution 4: Store Results and Functions in Dictionaries	509
16.2	Python Programming Essentials	511
16.2.1	More Data Structures	511
16.2.2	More Execution Structures	519
16.2.3	When to Use Different Data and Execution Structures	521
16.3	Try This!	523
16.4	More Discipline-Specific Practice	529
16.5	Chapter Review	530
16.5.1	Self-Test Questions	530
16.5.2	Chapter Summary	531
16.5.3	Self-Test Answers	533

17 Classes and Inheritance	536
17.1 Examples of Classes and Inheritance	536
17.1.1 Scientific Modeling Example	537
17.1.2 Scientific Bibliography Example	544
17.2 Python Programming Essentials	546
17.2.1 Defining and Using a Class	546
17.2.2 Inheritance	550
17.2.3 More Sophisticated Sorting Using <code>sorted</code>	553
17.2.4 Why Create Our Own Classes?	554
17.2.5 Automating Handling of Objects and Modules	557
17.3 Try This!	560
17.4 More Discipline-Specific Practice	564
17.5 Chapter Review	564
17.5.1 Self-Test Questions	564
17.5.2 Chapter Summary	566
17.5.3 Self-Test Answers	568
18 More Ways of Storing Information in Files	570
18.1 Examples of Using Other File Formats	570
18.2 Python Programming Essentials	576
18.2.1 Excel Files	576
18.2.2 pickle Files	578
18.2.3 netCDF files	579
18.3 Try This!	583
18.4 More Discipline-Specific Practice	589
18.5 Chapter Review	589
18.5.1 Self-Test Questions	589
18.5.2 Chapter Summary	590
18.5.3 Self-Test Answers	592
19 Basic Searching and Sorting	595
19.1 Examples of Searching and Sorting	595
19.2 Python Programming Essentials	598
19.2.1 Summary of Some Ways to Search and Sort	598
19.2.2 Searching and Sorting Algorithms	601
19.2.3 Basic Searching and Sorting Using pandas	611
19.3 Try This!	622
19.4 More Discipline-Specific Practice	628
19.5 Chapter Review	628
19.5.1 Self-Test Questions	628
19.5.2 Chapter Summary	629
19.5.3 Self-Test Answers	631

20 Recursion	633
20.1 Example of Recursion	633
20.2 Python Programming Essentials	635
20.2.1 Using the <code>walk</code> Generator	635
20.2.2 Recursion and Writing Recursive Code	637
20.2.3 More Applications of Recursion	642
20.3 Try This!	645
20.4 More Discipline-Specific Practice	649
20.5 Chapter Review	649
20.5.1 Self-Test Questions	649
20.5.2 Chapter Summary	650
20.5.3 Self-Test Answers	651
Part IV Going from a Program Working to Working Well	655
21 Make It Usable to Others: Documentation and Sphinx	657
21.1 Introduction	657
21.2 Principles of Documenting	657
21.3 General Convention for Docstrings: The NumPy Format	659
21.4 The Sphinx Documentation Generator	660
22 Make It Fast: Performance	666
22.1 Introduction	666
22.2 Preliminaries	666
22.2.1 Describing the Complexity of Code	666
22.2.2 Practices That Can Result in Inefficient Code	668
22.3 Finding the Bottlenecks Using Profilers	670
22.3.1 <code>timeit</code>	671
22.3.2 <code>cProfile</code>	672
22.3.3 <code>line-profiler</code>	674
22.3.4 <code>memory-profiler</code>	676
22.4 Fixing the Bottlenecks	678
22.4.1 Generators	678
22.4.2 Just-in-Time Compilation	680
22.5 Pitfalls When Trying to Improve Performance	682
23 Make It Correct: Linting and Unit Testing	683
23.1 Introduction	683
23.2 Linting	683
23.3 Unit Testing	686
23.3.1 <code>unittest</code>	687
23.3.2 <code>pytest</code>	688
23.4 The “Test-Driven Development” Process	690

24 Make It Manageable: Version Control and Build Management	693
24.1 Introduction	693
24.2 Version Control	693
24.2.1 Using Git as a Single User	694
24.2.2 Using Git as a User Who Is Part of a Collaboration	696
24.2.3 Using Git with Branching	697
24.3 Packaging	698
24.4 Build Management and Continuous Integration	699
25 Make It Talk to Other Languages	702
25.1 Introduction	702
25.2 Talking with Fortran Programs	702
25.3 Talking with C/C++ Programs	704
Appendix A List of Units	706
Appendix B Summary of Data Structures	708
Appendix C Contents by Programming Topic	709
C.1 Introductory Programming Topics	709
C.1.1 What Is a Program and General Elements of Python	709
C.1.2 Variables and Expressions	710
C.1.3 Typing and Some Basic Types	710
C.1.4 Strings	711
C.1.5 Functions	711
C.1.6 Branching, Conditionals, and Booleans	712
C.1.7 Looping	712
C.1.8 Console Input and Output	713
C.1.9 Text File Input and Output	713
C.1.10 Exceptions	713
C.1.11 Arrays	714
C.1.12 Classes	715
C.2 Intermediate Programming Topics	715
C.2.1 Abstract Data Types and Structures	715
C.2.2 Algorithm Analysis	716
C.2.3 Searching and Sorting	716
C.2.4 Recursion	717
C.3 Other Topics	717
C.3.1 How to Program and Programming Style	717
C.3.2 Distributions and Interactive Development Environments (IDEs)	717

C.3.3	Packages and Modules	717
C.3.4	Calculation Functions and Modules	718
C.3.5	Visualization	718
	Glossary	719
	Acronyms and Abbreviations	726
	Bibliography	727
	Index	729