

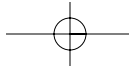
# Fundamentals of Python:

From First Programs  
Through Data Structures

**Kenneth A. Lambert**  
**Martin Osborne, Contributing Author**



Australia • Brazil • Japan • Korea • Mexico • Singapore • Spain • United Kingdom • United States



**Fundamentals of Python: From First Programs Through Data Structures**  
**Kenneth A. Lambert**

Executive Editor: Marie Lee  
Acquisitions Editor: Amy Jollymore  
Senior Product Manager: Alyssa Pratt  
Development Editor: Ann Shaffer  
Editorial Assistant: Julia Leroux-Lindsey  
Marketing Manager: Bryant Chrzan  
Content Project Manager: Matt Hutchinson  
Art Director: Marissa Falco  
Compositor: Gex Publishing Services

© 2010 Course Technology, Cengage Learning  
ALL RIGHTS RESERVED. No part of this work covered by the copyright herein may be reproduced, transmitted, stored or used in any form or by any means graphic, electronic, or mechanical, including but not limited to photocopying, recording, scanning, digitizing, taping, Web distribution, information networks, or information storage and retrieval systems, except as permitted under Section 107 or 108 of the 1976 United States Copyright Act, without the prior written permission of the publisher.

For product information and technology assistance, contact us at **Cengage Learning Customer & Sales Support, 1-800-354-9706**  
For permission to use material from this text or product, submit all requests online at [www.cengage.com/permissions](http://www.cengage.com/permissions)  
Further permissions questions can be emailed to [permissionrequest@cengage.com](mailto:permissionrequest@cengage.com)

ISBN-13: 978-1-4239-0218-8  
ISBN-10: 1-4239-0218-1

**Course Technology**  
25 Thomson Place  
Boston, Massachusetts 02210  
USA

Cengage Learning is a leading provider of customized learning solutions with office locations around the globe, including Singapore, the United Kingdom, Australia, Mexico, Brazil, and Japan. Locate your local office at: [international.cengage.com/region](http://international.cengage.com/region)

Cengage Learning products are represented in Canada by Nelson Education, Ltd.

For your lifelong learning solutions, visit [course.cengage.com](http://course.cengage.com).

Purchase any of our products at your local college store or at our preferred online store [www.ichapters.com](http://www.ichapters.com).

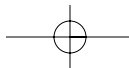
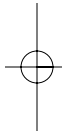
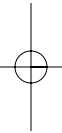
Some of the product names and company names used in this book have been used for identification purposes only and may be trademarks or registered trademarks of their respective manufacturers and sellers.

Any fictional data related to persons or companies or URLs used throughout this book is intended for instructional purposes only. At the time this book was printed, any such data was fictional and not belonging to any real persons or companies.

Course Technology, a part of Cengage Learning, reserves the right to revise this publication and make changes from time to time in its content without notice.

The programs in this book are for instructional purposes only. They have been tested with care, but are not guaranteed for any particular intent beyond educational purposes. The author and the publisher do not offer any warranties or representations, nor do they accept any liabilities with respect to the programs.

Printed in Canada  
1 2 3 4 5 6 7 12 11 10 09 08



# Table of Contents

**[CHAPTER] 1**

**INTRODUCTION**

<b>1.1</b>	Two Fundamental Ideas of Computer Science: Algorithms and Information Processing .....	2
1.1.1	Algorithms .....	2
1.1.2	Information Processing .....	4
<b>1.1</b>	Exercises .....	5
<b>1.2</b>	The Structure of a Modern Computer System .....	6
1.2.1	Computer Hardware .....	6
1.2.2	Computer Software .....	8
<b>1.2</b>	Exercises .....	10
<b>1.3</b>	A Not-So-Brief History of Computing Systems .....	10
1.3.1	Before Electronic Digital Computers .....	11
1.3.2	The First Electronic Digital Computers (1940–1950) .....	15
1.3.3	The First Programming Languages (1950–1965) .....	16
1.3.4	Integrated Circuits, Interaction, and Timesharing (1965–1975) .....	18
1.3.5	Personal Computing and Networks (1975–1990) .....	19
1.3.6	Consultation, Communication, and Ubiquitous Computing (1990–Present) .....	21
<b>1.4</b>	Getting Started with Python Programming .....	23
1.4.1	Running Code in the Interactive Shell .....	23
1.4.2	Input, Processing, and Output .....	25
1.4.3	Editing, Saving, and Running a Script .....	27
1.4.4	Behind the Scenes: How Python Works .....	29
<b>1.4</b>	Exercises .....	30
<b>1.5</b>	Detecting and Correcting Syntax Errors .....	30
<b>1.5</b>	Exercises .....	32
	Suggestions for Further Reading .....	32
	Summary .....	32
	Review Questions .....	35
	Projects .....	37

**[CHAPTER] 2**

**SOFTWARE DEVELOPMENT, DATA TYPES, AND EXPRESSIONS 39**

<b>2.1</b>	The Software Development Process .....	40
<b>2.1</b>	Exercises .....	43
<b>2.2</b>	Case Study: Income Tax Calculator .....	43
2.2.1	Request .....	43
2.2.2	Analysis .....	44
2.2.3	Design .....	44
2.2.4	Implementation (Coding) .....	45
2.2.5	Testing .....	46

2.3	Strings, Assignment, and Comments.....	47
2.3.1	Data Types.....	47
2.3.2	String Literals.....	48
2.3.3	Escape Sequences.....	50
2.3.4	String Concatenation.....	50
2.3.5	Variables and the Assignment Statement.....	51
2.3.6	Program Comments and Docstrings.....	52
2.3	Exercises.....	53
2.4	Numeric Data Types and Character Sets.....	54
2.4.1	Integers and Long Integers.....	54
2.4.2	Floating-Point Numbers.....	55
2.4.3	Character Sets.....	56
2.4	Exercises.....	57
2.5	Expressions.....	58
2.5.1	Arithmetic Expressions.....	58
2.5.2	Mixed-Mode Arithmetic and Type Conversions.....	60
2.5	Exercises.....	63
2.6	Using Functions and Modules.....	63
2.6.1	Calling Functions: Arguments and Return Values.....	64
2.6.2	The math Module.....	65
2.6.3	The Main Module.....	66
2.6.4	Program Format and Structure.....	67
2.6.5	Running a Script from a Terminal Command Prompt.....	68
2.6	Exercises.....	70
	Summary.....	70
	Review Questions.....	72
	Projects.....	73
<b>[CHAPTER] 3</b>	<b>CONTROL STATEMENTS</b>	<b>75</b>
3.1	Definite Iteration: The for Loop.....	76
3.1.1	Executing a Statement a Given Number of Times.....	76
3.1.2	Count-Controlled Loops.....	77
3.1.3	Augmented Assignment.....	79
3.1.4	Loop Errors: Off-by-One Error.....	80
3.1.5	Traversing the Contents of a Data Sequence.....	80
3.1.6	Specifying the Steps in the Range.....	81
3.1.7	Loops That Count Down.....	82
3.1	Exercises.....	83
3.2	Formatting Text for Output.....	83
3.2	Exercises.....	86
3.3	Case Study: An Investment Report.....	87
3.3.1	Request.....	87
3.3.2	Analysis.....	87
3.3.3	Design.....	88
3.3.4	Implementation (Coding).....	88
3.3.5	Testing.....	90
3.4	Selection: if and if-else Statements.....	91
3.4.1	The Boolean Type, Comparisons, and Boolean Expressions.....	91
3.4.2	if-else Statements.....	92

	3.4.3	One-Way Selection Statements.....	94
	3.4.4	Multi-way if Statements .....	95
	3.4.5	Logical Operators and Compound Boolean Expressions.....	97
	3.4.6	Short-Circuit Evaluation .....	99
	3.4.7	Testing Selection Statements .....	100
3.4		Exercises.....	101
3.5		Conditional Iteration: The while Loop.....	102
	3.5.1	The Structure and Behavior of a while Loop .....	102
	3.5.2	Count Control with a while Loop.....	104
	3.5.3	The while True Loop and the break Statement .....	105
	3.5.4	Random Numbers.....	107
	3.5.5	Loop Logic, Errors, and Testing.....	109
3.5		Exercises.....	109
3.6		Case Study: Approximating Square Roots.....	110
	3.6.1	Request .....	110
	3.6.2	Analysis .....	110
	3.6.3	Design.....	110
	3.6.4	Implementation (Coding) .....	112
	3.6.5	Testing .....	113
		Summary .....	113
		Review Questions .....	116
		Projects.....	118
		<b>STRINGS AND TEXT FILES</b>	<b>121</b>
		Accessing Characters and Substrings in Strings.....	122
	4.1.1	The Structure of Strings.....	122
	4.1.2	The Subscript Operator.....	123
	4.1.3	Slicing for Substrings.....	124
	4.1.4	Testing for a Substring with the in Operator .....	125
4.1		Exercises.....	126
4.2		Data Encryption .....	126
4.2		Exercises.....	129
4.3		Strings and Number Systems.....	129
	4.3.1	The Positional System for Representing Numbers.....	130
	4.3.2	Converting Binary to Decimal .....	131
	4.3.3	Converting Decimal to Binary .....	132
	4.3.4	Conversion Shortcuts.....	133
	4.3.5	Octal and Hexadecimal Numbers .....	134
4.3		Exercises.....	136
4.4		String Methods .....	136
4.4		Exercises.....	140
4.5		Text Files.....	141
	4.5.1	Text Files and Their Format.....	141
	4.5.2	Writing Text to a File .....	142
	4.5.3	Writing Numbers to a File .....	142
	4.5.4	Reading Text from a File .....	143
	4.5.5	Reading Numbers from a File .....	145
	4.5.6	Accessing and Manipulating Files and Directories on Disk.....	146

[CHAPTER] **4**

4.5	Exercises.....	148
4.6	Case Study: Text Analysis.....	148
4.6.1	Request .....	149
4.6.2	Analysis .....	149
4.6.3	Design.....	150
4.6.4	Implementation (Coding) .....	151
4.6.5	Testing .....	152
	Summary .....	153
	Review Questions .....	154
	Projects.....	156
<b>[CHAPTER] 5</b>	<b>LISTS AND DICTIONARIES</b>	<b>159</b>
5.1	Lists .....	160
5.1.1	List Literals and Basic Operators .....	160
5.1.2	Replacing an Element in a List .....	163
5.1.3	List Methods for Inserting and Removing Elements .....	165
5.1.4	Searching a List.....	167
5.1.5	Sorting a List.....	168
5.1.6	Mutator Methods and the Value None .....	168
5.1.7	Aliasing and Side Effects.....	169
5.1.8	Equality: Object Identity and Structural Equivalence .....	171
5.1.9	Example: Using a List to Find the Median of a Set of Numbers .....	172
5.1.10	Tuples .....	173
5.1	Exercises.....	174
5.2	Defining Simple Functions .....	175
5.2.1	The Syntax of Simple Function Definitions .....	175
5.2.2	Parameters and Arguments.....	176
5.2.3	The <code>return</code> Statement.....	177
5.2.4	Boolean Functions.....	177
5.2.5	Defining a main Function.....	178
5.2	Exercises.....	179
5.3	Case Study: Generating Sentences .....	179
5.3.1	Request .....	179
5.3.2	Analysis .....	179
5.3.3	Design.....	180
5.3.4	Implementation (Coding) .....	182
5.3.5	Testing .....	183
5.4	Dictionaries.....	183
5.4.1	Dictionary Literals.....	183
5.4.2	Adding Keys and Replacing Values .....	184
5.4.3	Accessing Values.....	185
5.4.4	Removing Keys .....	186
5.4.5	Traversing a Dictionary .....	186
5.4.6	Example: The Hexadecimal System Revisited.....	188
5.4.7	Example: Finding the Mode of a List of Values .....	189
5.4	Exercises.....	190

5.5	Case Study: Nondirective Psychotherapy .....	191
5.5.1	Request .....	191
5.5.2	Analysis .....	191
5.5.3	Design .....	192
5.5.4	Implementation (Coding) .....	193
5.5.5	Testing .....	195
	Summary .....	195
	Review Questions .....	196
	Projects .....	198
<b>6</b>	<b>DESIGN WITH FUNCTIONS</b>	<b>201</b>
6.1	Functions as Abstraction Mechanisms .....	202
6.1.1	Functions Eliminate Redundancy .....	202
6.1.2	Functions Hide Complexity .....	203
6.1.3	Functions Support General Methods with Systematic Variations .....	204
6.1.4	Functions Support the Division of Labor .....	205
6.1	Exercises .....	205
6.2	Problem Solving with Top-Down Design .....	206
6.2.1	The Design of the Text-Analysis Program .....	206
6.2.2	The Design of the Sentence-Generator Program .....	207
6.2.3	The Design of the Doctor Program .....	209
6.2	Exercises .....	210
6.3	Design with Recursive Functions .....	211
6.3.1	Defining a Recursive Function .....	211
6.3.2	Tracing a Recursive Function .....	213
6.3.3	Using Recursive Definitions to Construct Recursive Functions .....	214
6.3.4	Recursion in Sentence Structure .....	214
6.3.5	Infinite Recursion .....	215
6.3.6	The Costs and Benefits of Recursion .....	216
6.3	Exercises .....	218
6.4	Case Study: Gathering Information from a File System .....	219
6.4.1	Request .....	219
6.4.2	Analysis .....	220
6.4.3	Design .....	222
6.4.4	Implementation (Coding) .....	224
6.5	Managing a Program's Namespace .....	227
6.5.1	Module Variables, Parameters, and Temporary Variables .....	227
6.5.2	Scope .....	228
6.5.3	Lifetime .....	229
6.5.4	Default (Keyword) Arguments .....	230
6.5	Exercises .....	232
6.6	Higher-Order Functions (Advanced Topic) .....	233
6.6.1	Functions as First-Class Data Objects .....	233
6.6.2	Mapping .....	234
6.6.3	Filtering .....	236
6.6.4	Reducing .....	237
6.6.5	Using Lambda to Create Anonymous Functions .....	237
6.6.6	Creating Jump Tables .....	238

[CHAPTER]

**6**

**201**

	6.6	Exercises.....	239
		Summary .....	240
		Review Questions .....	242
		Projects.....	244
<b>[CHAPTER]</b>	<b>7</b>	<b>SIMPLE GRAPHICS AND IMAGE PROCESSING</b>	<b>247</b>
	7.1	Simple Graphics .....	248
	7.1.1	Overview of Turtle Graphics .....	248
	7.1.2	Turtle Operations.....	249
	7.1.3	Object Instantiation and the <code>turtlegraphics</code> Module .....	251
	7.1.4	Drawing Two-Dimensional Shapes.....	254
	7.1.5	Taking a Random Walk.....	255
	7.1.6	Colors and the RGB System.....	256
	7.1.7	Example: Drawing with Random Colors .....	257
	7.1.8	Using the <code>str</code> Function with Objects .....	259
	7.1	Exercises.....	260
	7.2	Case Study: Recursive Patterns in Fractals.....	261
	7.2.1	Request .....	262
	7.2.2	Analysis .....	262
	7.2.3	Design.....	263
	7.2.4	Implementation (Coding) .....	265
	7.3	Image Processing .....	266
	7.3.1	Analog and Digital Information .....	266
	7.3.2	Sampling and Digitizing Images .....	267
	7.3.3	Image File Formats .....	267
	7.3.4	Image-Manipulation Operations .....	268
	7.3.5	The Properties of Images .....	269
	7.3.6	The <code>images</code> Module .....	269
	7.3.7	A Loop Pattern for Traversing a Grid .....	273
	7.3.8	A Word on Tuples.....	274
	7.3.9	Converting an Image to Black and White .....	275
	7.3.10	Converting an Image to Grayscale.....	277
	7.3.11	Copying an Image .....	278
	7.3.12	Blurring an Image .....	279
	7.3.13	Edge Detection .....	280
	7.3.14	Reducing the Image Size .....	281
	7.3	Exercises.....	283
		Summary .....	284
		Review Questions .....	285
		Projects.....	287
<b>[CHAPTER]</b>	<b>8</b>	<b>DESIGN WITH CLASSES</b>	<b>291</b>
	8.1	Getting Inside Objects and Classes .....	292
	8.1.1	A First Example: The <code>Student</code> Class.....	293
	8.1.2	Docstrings .....	296
	8.1.3	Method Definitions.....	296
	8.1.4	The <code>__init__</code> Method and Instance Variables.....	297
	8.1.5	The <code>__str__</code> Method.....	298



	<b>8.1.6</b>	Accessors and Mutators .....	298
	<b>8.1.7</b>	The Lifetime of Objects .....	299
	<b>8.1.8</b>	Rules of Thumb for Defining a Simple Class.....	300
<b>8.1</b>		Exercises.....	301
<b>8.2</b>		Case Study: Playing the Game of Craps .....	301
	<b>8.2.1</b>	Request .....	301
	<b>8.2.2</b>	Analysis .....	301
	<b>8.2.3</b>	Design.....	302
	<b>8.2.4</b>	Implementation (Coding).....	304
<b>8.3</b>		Data-Modeling Examples.....	307
	<b>8.3.1</b>	Rational Numbers.....	307
	<b>8.3.2</b>	Rational Number Arithmetic and Operator Overloading.....	309
	<b>8.3.3</b>	Comparisons and the <code>__cmp__</code> Method.....	310
	<b>8.3.4</b>	Equality and the <code>__eq__</code> Method.....	311
	<b>8.3.5</b>	Savings Accounts and Class Variables .....	312
	<b>8.3.6</b>	Putting the Accounts into a Bank.....	315
	<b>8.3.7</b>	Using <code>cPickle</code> for Permanent Storage of Objects .....	317
	<b>8.3.8</b>	Input of Objects and the <code>try-except</code> Statement.....	318
	<b>8.3.9</b>	Playing Cards .....	319
<b>8.3</b>		Exercises.....	323
<b>8.4</b>		Case Study: An ATM.....	323
	<b>8.4.1</b>	Request .....	323
	<b>8.4.2</b>	Analysis.....	323
	<b>8.4.3</b>	Design.....	325
	<b>8.4.4</b>	Implementation (Coding).....	327
<b>8.5</b>		Structuring Classes with Inheritance and Polymorphism.....	329
	<b>8.5.1</b>	Inheritance Hierarchies and Modeling .....	330
	<b>8.5.2</b>	Example: A Restricted Savings Account.....	331
	<b>8.5.3</b>	Example: The Dealer and a Player in the Game of Blackjack .....	333
	<b>8.5.4</b>	Polymorphic Methods.....	338
	<b>8.5.5</b>	Abstract Classes .....	338
	<b>8.5.6</b>	The Costs and Benefits of Object-Oriented Programming.....	339
<b>8.5</b>		Exercises.....	341
		Summary .....	341
		Review Questions .....	343
		Projects.....	344

**[CHAPTER] 9 GRAPHICAL USER INTERFACES 347**

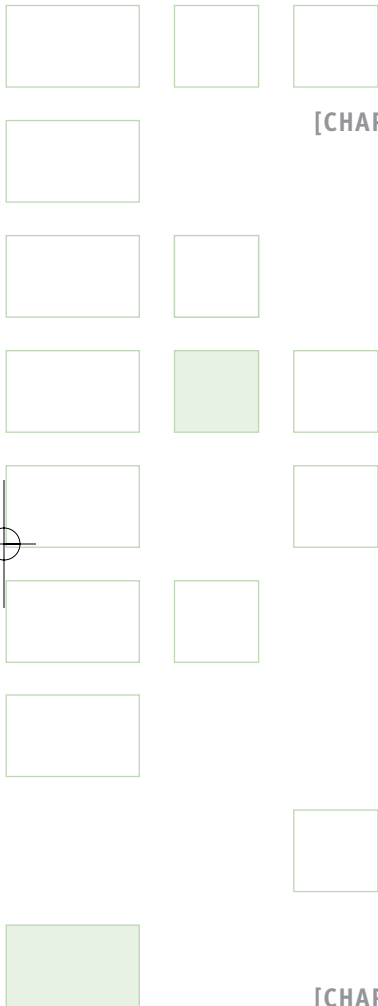
<b>9.1</b>		The Behavior of Terminal-Based Programs and GUI-Based Programs.....	348
	<b>9.1.1</b>	The Terminal-Based Version .....	348
	<b>9.1.2</b>	The GUI-Based Version.....	349
	<b>9.1.3</b>	Event-Driven Programming.....	351
<b>9.1</b>		Exercises.....	353
<b>9.2</b>		Coding Simple GUI-Based Programs.....	353
	<b>9.2.1</b>	Windows and Labels.....	354
	<b>9.2.2</b>	Displaying Images.....	355
	<b>9.2.3</b>	Command Buttons and Responding to Events.....	356
	<b>9.2.4</b>	Viewing the Images of Playing Cards .....	358

	9.2.5	Entry Fields for the Input and Output of Text .....	361
	9.2.6	Using Pop-up Dialog Boxes .....	363
9.2		Exercises.....	364
9.3		Case Study: A GUI-Based ATM.....	365
	9.3.1	Request .....	365
	9.3.2	Analysis .....	365
	9.3.3	Design.....	366
	9.3.4	Implementation (Coding) .....	367
9.4		Other Useful GUI Resources .....	370
	9.4.1	Colors .....	371
	9.4.2	Text Attributes.....	371
	9.4.3	Sizing and Justifying an Entry .....	372
	9.4.4	Sizing the Main Window.....	373
	9.4.5	Grid Attributes .....	374
	9.4.6	Using Nested Frames to Organize Components.....	378
	9.4.7	Multi-Line Text Widgets.....	379
	9.4.8	Scrolling List Boxes .....	382
	9.4.9	Mouse Events .....	385
	9.4.10	Keyboard Events .....	386
9.4		Exercises.....	387
		Summary .....	388
		Review Questions .....	389
		Projects.....	390

**[CHAPTER] 10 MULTITHREADING, NETWORKS, AND CLIENT/SERVER PROGRAMMING**

**393**

10.1		Threads and Processes .....	394
	10.1.1	Threads.....	395
	10.1.2	Sleeping Threads.....	398
	10.1.3	Producer, Consumer, and Synchronization .....	400
10.1		Exercises.....	407
10.2		Networks, Clients, and Servers.....	407
	10.2.1	IP Addresses .....	407
	10.2.2	Ports, Servers, and Clients.....	409
	10.2.3	Sockets and a Day/Time Client Script.....	410
	10.2.4	A Day/Time Server Script.....	412
	10.2.5	A Two-Way Chat Script.....	414
	10.2.6	Handling Multiple Clients Concurrently .....	416
	10.2.7	Setting Up Conversations for Others .....	418
10.2		Exercises.....	420
10.3		Case Study: A Multi-Client Chat Room .....	421
	10.3.1	Request .....	421
	10.3.2	Analysis .....	421
	10.3.3	Design.....	422
	10.3.4	Implementation (Coding) .....	423
		Summary .....	425
		Review Questions .....	426
		Projects.....	428



[CHAPTER] **11**

- 11.1
- 11.2
- 11.2
- 11.3
- 11.3
- 11.4
- 11.4
- 11.5
- 11.6
- 11.7

**SEARCHING, SORTING, AND COMPLEXITY ANALYSIS 431**

Measuring the Efficiency of Algorithms.....432

11.1.1 Measuring the Run Time of an Algorithm .....432

11.1.2 Counting Instructions.....435

11.1.3 Measuring the Memory Used by an Algorithm.....438

Exercises.....439

Complexity Analysis .....439

11.2.1 Orders of Complexity .....439

11.2.2 Big-O Notation .....441

11.2.3 The Role of the Constant of Proportionality .....442

Exercises.....443

Search Algorithms .....443

11.3.1 Search for a Minimum .....444

11.3.2 Linear Search of a List .....444

11.3.3 Best-Case, Worst-Case, and Average-Case Performance.....445

11.3.4 Binary Search of a List.....446

11.3.5 Comparing Data Items and the `cmp` Function .....448

Exercises.....449

Sort Algorithms .....450

11.4.1 Selection Sort .....450

11.4.2 Bubble Sort.....452

11.4.3 Insertion Sort .....453

11.4.4 Best-Case, Worst-Case, and Average-Case Performance Revisited...455

Exercises.....456

An Exponential Algorithm: Recursive Fibonacci .....457

Converting Fibonacci to a Linear Algorithm.....458

Case Study: An Algorithm Profiler.....460

11.7.1 Request .....460

11.7.2 Analysis.....460

11.7.3 Design.....462

11.7.4 Implementation (Coding) .....463

Summary .....466

Review Questions .....467

Projects.....468

[CHAPTER] **12**

- 12.1
- 12.1
- 12.2

**TOOLS FOR DESIGN, DOCUMENTATION, AND TESTING 471**

Software Design with UML.....472

12.1.1 UML and Modeling.....473

12.1.2 Use Case Diagrams .....473

12.1.3 Class Diagrams.....476

12.1.4 Collaboration Diagrams.....479

12.1.5 From Collaboration Diagram to Code .....481

12.1.6 Inheritance.....482

Exercises.....483

Documentation .....484

12.2.1 Writing APIs .....484

12.2.2 Revisiting the `Student` Class .....485

12.2.3 Preconditions and Postconditions .....487

	12.2.4	Enforcing Preconditions with Exceptions.....	488
	12.2.5	Web-Based Documentation with <code>pydoc</code> .....	490
12.2		Exercises.....	493
12.3		Testing.....	493
	12.3.1	What to Test.....	494
	12.3.2	Three Approaches to Choosing Test Data.....	494
	12.3.2.1	Haphazard Testing.....	495
	12.3.2.2	Black-Box Testing.....	495
	12.3.2.3	White-Box Testing.....	495
	12.3.3	When to Test.....	496
	12.3.3.1	Unit Testing.....	496
	12.3.3.2	Integration Testing.....	496
	12.3.3.3	Acceptance Testing.....	496
	12.3.3.4	Regression Testing.....	497
	12.3.4	Proofs of Program Correctness.....	497
	12.3.5	Unit Testing in Python.....	498
12.3		Exercises.....	502
		Suggestions for Further Reading.....	502
		Summary.....	503
		Review Questions.....	504
		Projects.....	505
<b>[CHAPTER]</b>	<b>13</b>	<b>COLLECTIONS, ARRAYS, AND LINKED STRUCTURES</b>	<b>507</b>
	13.1	Overview of Collections.....	508
	13.1.1	Linear Collections.....	508
	13.1.2	Hierarchical Collections.....	508
	13.1.3	Graph Collections.....	509
	13.1.4	Unordered Collections.....	510
	13.1.5	Operations on Collections.....	510
	13.1.6	Abstraction and Abstract Data Types.....	512
13.1		Exercises.....	513
13.2		Data Structures for Implementing Collections: Arrays.....	513
	13.2.1	The Array Data Structure.....	513
	13.2.2	Random Access and Contiguous Memory.....	516
	13.2.3	Static Memory and Dynamic Memory.....	517
	13.2.4	Physical Size and Logical Size.....	518
13.2		Exercises.....	519
13.3		Operations on Arrays.....	519
	13.3.1	Increasing the Size of an Array.....	520
	13.3.2	Decreasing the Size of an Array.....	521
	13.3.3	Inserting an Item into an Array That Grows.....	521
	13.3.4	Removing an Item from an Array.....	523
	13.3.5	Complexity Trade-Off: Time, Space, and Arrays.....	524
13.3		Exercises.....	525
13.4		Two-Dimensional Arrays (Grids).....	525
	13.4.1	Processing a Grid.....	526
	13.4.2	Creating and Initializing a Grid.....	526
	13.4.3	Defining a Grid Class.....	527
	13.4.4	Ragged Grids and Multidimensional Arrays.....	528

	Exercises.....	528
<b>13.4</b>	Linked Structures .....	529
<b>13.5</b>	<b>13.5.1</b> Singly Linked Structures and Doubly Linked Structures .....	530
	<b>13.5.2</b> Noncontiguous Memory and Nodes.....	531
	<b>13.5.3</b> Defining a Singly Linked Node Class.....	533
	<b>13.5.4</b> Using the Singly Linked Node Class .....	534
<b>13.5</b>	Exercises.....	536
<b>13.6</b>	Operations on Singly Linked Structures .....	536
	<b>13.6.1</b> Traversal .....	536
	<b>13.6.2</b> Searching .....	538
	<b>13.6.3</b> Replacement .....	539
	<b>13.6.4</b> Inserting at the Beginning .....	539
	<b>13.6.5</b> Inserting at the End .....	540
	<b>13.6.6</b> Removing at the Beginning .....	542
	<b>13.6.7</b> Removing at the End .....	543
	<b>13.6.8</b> Inserting at Any Position .....	544
	<b>13.6.9</b> Removing at Any Position .....	547
	<b>13.6.10</b> Complexity Trade-Off: Time, Space, and Singly Linked Structures.....	549
<b>13.6</b>	Exercises.....	550
<b>13.7</b>	Variations on a Link .....	550
	<b>13.7.1</b> A Circular Linked Structure with a Dummy Header Node .....	550
	<b>13.7.2</b> Doubly Linked Structures .....	552
<b>13.7</b>	Exercises.....	555
	Summary .....	555
	Review Questions .....	556
	Projects .....	557

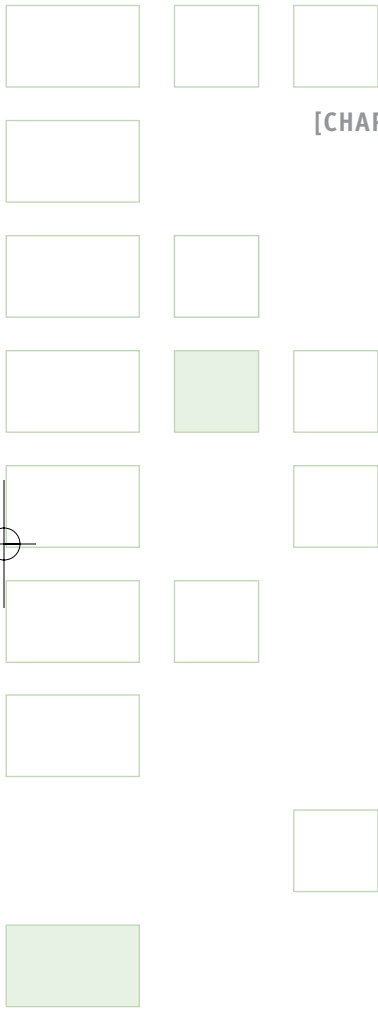
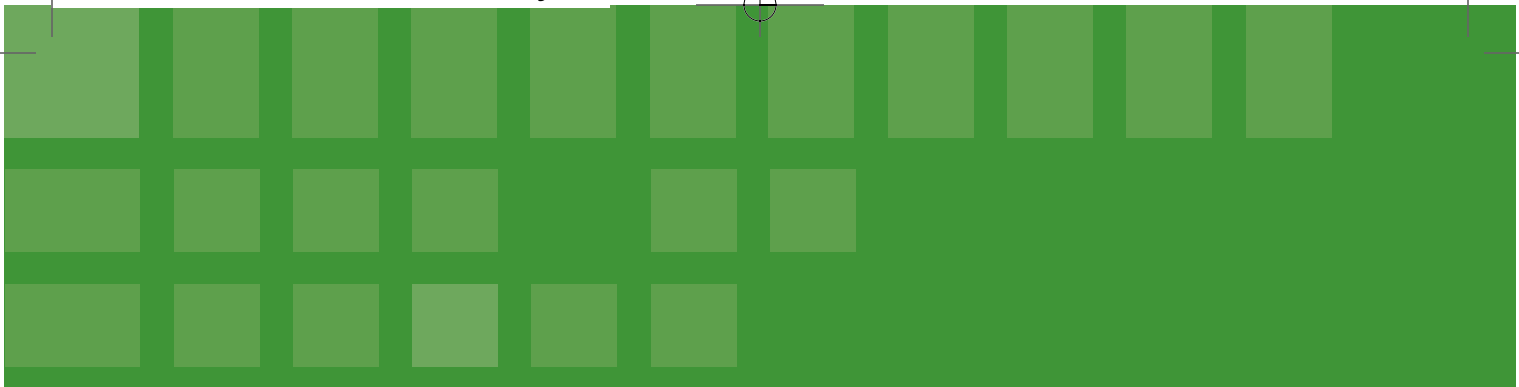
**[CHAPTER] 14 LINEAR COLLECTIONS: STACKS 561**

<b>14.1</b>	Overview of Stacks .....	562
<b>14.2</b>	Using a Stack .....	563
	<b>14.2.1</b> The Stack Interface .....	564
	<b>14.2.2</b> Instantiating a Stack.....	565
	<b>14.2.3</b> Example Application: Matching Parentheses.....	566
<b>14.2</b>	Exercises.....	568
<b>14.3</b>	Three Applications of Stacks .....	569
	<b>14.3.1</b> Evaluating Arithmetic Expressions.....	569
	<b>14.3.2</b> Evaluating Postfix Expressions .....	570
<b>14.3.2</b>	Exercises.....	572
	<b>14.3.3</b> Converting Infix to Postfix .....	572
<b>14.3.3</b>	Exercises.....	575
	<b>14.3.4</b> Backtracking .....	575
	<b>14.3.5</b> Memory Management.....	577
<b>14.4</b>	Implementations of Stacks .....	580
	<b>14.4.1</b> Test Driver.....	580
	<b>14.4.2</b> Array Implementation.....	581
	<b>14.4.3</b> Linked Implementation .....	584
	<b>14.4.4</b> Time and Space Analysis of the Two Implementations.....	587
<b>14.4</b>	Exercises.....	588

<b>14.5</b>	Case Study: Evaluating Postfix Expressions .....	589
<b>14.5.1</b>	Request .....	589
<b>14.5.2</b>	Analysis .....	589
<b>14.5.3</b>	Design .....	592
<b>14.5.3.1</b>	Instance Variables and Methods for Class PFEvaluatorView .....	593
<b>14.5.3.2</b>	Instance Variables and Methods for Class PFEvaluatorModel .....	594
<b>14.5.3.3</b>	Instance Variables and Methods for Class PFEvaluator .....	594
<b>14.5.3.4</b>	Instance Variables and Methods for Class Scanner .....	595
<b>14.5.3.5</b>	Instance and Class Variables and Methods for Class Token .....	595
<b>14.5.4</b>	Implementation .....	596
	Summary .....	599
	Review Questions .....	600
	Projects .....	601

**[CHAPTER] 15 LINEAR COLLECTIONS: QUEUES 603**

<b>15.1</b>	Overview of Queues .....	604
<b>15.2</b>	The Queue Interface and Its Use .....	605
<b>15.2</b>	Exercises .....	608
<b>15.3</b>	Two Applications of Queues .....	609
<b>15.3.1</b>	Simulations .....	609
<b>15.3.2</b>	Round-Robin CPU Scheduling .....	611
<b>15.3</b>	Exercises .....	612
<b>15.4</b>	Implementations of Queues .....	612
<b>15.4.1</b>	A Linked Implementation .....	612
<b>15.4.2</b>	An Array Implementation .....	614
<b>15.4.2.1</b>	A First Attempt .....	615
<b>15.4.2.2</b>	A Second Attempt .....	615
<b>15.4.2.3</b>	A Third Attempt .....	616
<b>15.4.3</b>	Time and Space Analysis for the Two Implementations .....	617
<b>15.4</b>	Exercises .....	618
<b>15.5</b>	Case Study: Simulating a Supermarket Checkout Line .....	618
<b>15.5.1</b>	Request .....	618
<b>15.5.2</b>	Analysis .....	618
<b>15.5.3</b>	The Interface .....	619
<b>15.5.4</b>	Classes and Responsibilities .....	620
<b>15.6</b>	Priority Queues .....	627
<b>15.6</b>	Exercise .....	632
<b>15.7</b>	Case Study: An Emergency Room Scheduler .....	633
<b>15.7.1</b>	Request .....	633
<b>15.7.2</b>	Analysis .....	633
<b>15.7.3</b>	Classes .....	635
<b>15.7.4</b>	Design and Implementation .....	635
	Summary .....	638
	Review Questions .....	638
	Projects .....	640



**[CHAPTER] 16**

**LINEAR COLLECTIONS: LISTS**

**643**

16.1	Overview of Lists.....	644
16.2	Using Lists.....	645
16.2.1	Index-Based Operations.....	646
16.2.2	Content-Based Operations.....	646
16.2.3	Position-Based Operations.....	647
16.2.4	Interfaces for Lists.....	652
16.2	Exercises.....	654
16.3	Applications of Lists.....	654
16.3.1	Heap-Storage Management.....	654
16.3.2	Organization of Files on a Disk.....	656
16.3.3	Implementation of Other ADTs.....	657
16.4	Indexed List Implementations.....	658
16.4.1	An Array-Based Implementation of an Indexed List.....	658
16.4.2	A Linked Implementation of an Indexed List.....	660
16.4.3	Time and Space Analysis for the Two Implementations.....	663
16.4	Exercises.....	665
16.5	Implementing Positional Lists.....	665
16.5.1	The Data Structures for a Linked Positional List.....	665
16.5.2	Methods Used to Navigate from Beginning to End.....	667
16.5.3	Methods Used to Navigate from End to Beginning.....	670
16.5.4	Insertions into a Positional List.....	670
16.5.5	Removals from a Positional List.....	671
16.5.5	Time and Space Analysis of Positional List Implementations.....	672
16.5	Exercises.....	672
16.6	Iterators.....	673
16.6.1	Using an Iterator in Python.....	674
16.6.2	Implementing an Iterator.....	676
16.6	Exercises.....	677
16.7	Case Study: Developing a Sorted List.....	678
16.7.1	Request.....	678
16.7.2	Analysis.....	678
16.7.3	Design.....	679
16.7.4	Implementation (Coding).....	680
	Summary.....	681
	Review Questions.....	682
	Projects.....	683

**[CHAPTER] 17**

**RECURSION**

**685**

17.1	$n \log n$ Sorting.....	686
17.1.1	Overview of Quicksort.....	686
17.1.2	Partitioning.....	687
17.1.3	Complexity Analysis of Quicksort.....	689
17.1.4	Implementation of Quicksort.....	690
17.1.5	Merge Sort.....	692
17.1.6	Complexity Analysis for Merge Sort.....	695
17.1	Exercises.....	696



<b>17.2</b>	Recursive List Processing.....	696
<b>17.2.1</b>	Basic Operations on a Lisp-Like List.....	696
<b>17.2.2</b>	Recursive Traversals of a Lisp-Like List.....	698
<b>17.2.3</b>	Building a Lisp-Like List.....	700
<b>17.2.4</b>	The Internal Structure of a Lisp-Like List.....	702
<b>17.2.5</b>	Lists and Functional Programming.....	703
<b>17.2</b>	Exercises.....	704
<b>17.3</b>	Recursion and Backtracking.....	705
<b>17.3.1</b>	A General Recursive Strategy.....	705
<b>17.3.2</b>	The Maze Problem Revisited.....	706
<b>17.3.3</b>	The Eight Queens Problem.....	709
<b>17.4</b>	Recursive Descent and Programming Languages.....	714
<b>17.4.1</b>	Introduction to Grammars.....	714
<b>17.4.2</b>	Recognizing, Parsing, and Interpreting Sentences in a Language.....	717
<b>17.4.3</b>	Lexical Analysis and the Scanner.....	717
<b>17.4.4</b>	Parsing Strategies.....	718
<b>17.5</b>	Case Study: A Recursive Descent Parser.....	719
<b>17.5.1</b>	Request.....	719
<b>17.5.2</b>	Analysis.....	719
<b>17.5.3</b>	Classes.....	720
<b>17.5.4</b>	Implementation (Coding).....	720
<b>17.6</b>	The Costs and Benefits of Recursion.....	722
<b>17.6.1</b>	No, Maybe, and Yes.....	723
<b>17.6.2</b>	Getting Rid of Recursion.....	723
<b>17.6.3</b>	Tail Recursion.....	724
	Summary.....	725
	Review Questions.....	726
	Projects.....	727

**[CHAPTER] 18 HIERARCHICAL COLLECTIONS: TREES 733**

<b>18.1</b>	An Overview of Trees.....	734
<b>18.1.1</b>	Tree Terminology.....	734
<b>18.1.2</b>	General Trees and Binary Trees.....	736
<b>18.1.3</b>	Recursive Definitions of Trees.....	736
<b>18.1</b>	Exercise.....	737
<b>18.2</b>	Why Use a Tree?.....	737
<b>18.3</b>	The Shape of Binary Trees.....	740
<b>18.3</b>	Exercises.....	742
<b>18.4</b>	Three Common Applications of Binary Trees.....	743
<b>18.4.1</b>	Heaps.....	743
<b>18.4.2</b>	Binary Search Trees.....	744
<b>18.4.3</b>	Expression Trees.....	745
<b>18.4</b>	Exercises.....	747
<b>18.5</b>	Binary Tree Traversals.....	747
<b>18.5</b>	Exercise.....	749
<b>18.6</b>	A Binary Tree ADT.....	749
<b>18.6.1</b>	The Interface for a Binary Tree ADT.....	750
<b>18.6.2</b>	Processing a Binary Tree.....	752



	<b>18.6.3</b>	Implementing a Binary Tree.....	753
	<b>18.6.4</b>	The String Representation of a Tree .....	756
		Exercises.....	757
<b>18.6</b>		Developing a Binary Search Tree .....	757
<b>18.7</b>	<b>18.7.1</b>	The Binary Search Tree Interface .....	757
	<b>18.7.2</b>	Data Structures for the Implementation of BST .....	758
	<b>18.7.3</b>	Searching a Binary Search Tree.....	759
	<b>18.7.4</b>	Inserting an Item into a Binary Search Tree.....	760
	<b>18.7.5</b>	Removing an Item from a Binary Search Tree .....	762
	<b>18.7.6</b>	Complexity Analysis of Binary Search Trees .....	763
		Exercises.....	763
<b>18.7</b>		Case Study: Parsing and Expression Trees.....	764
<b>18.8</b>	<b>18.8.1</b>	Request .....	764
	<b>18.8.2</b>	Analysis .....	764
	<b>18.8.3</b>	Design and Implementation of the Node Classes .....	765
	<b>18.8.4</b>	Design and Implementation of the <code>Parser</code> Class .....	767
<b>18.9</b>		An Array Implementation of Binary Trees .....	769
<b>18.9</b>		Exercises.....	770
<b>18.10</b>		Implementing Heaps .....	771
<b>18.10</b>		Exercises.....	774
<b>18.11</b>		Using a Heap to Implement a Priority Queue.....	774
		Summary .....	775
		Review Questions .....	776
		Projects.....	777
	<b>[CHAPTER] 19</b>	<b>UNORDERED COLLECTIONS: SETS AND DICTIONARIES 779</b>	
	<b>19.1</b>	Using Sets .....	780
	<b>19.1.1</b>	The Python <code>set</code> Class.....	781
	<b>19.1.2</b>	A Sample Session with Sets .....	782
	<b>19.1.3</b>	Applications of Sets .....	783
	<b>19.1.4</b>	Implementations of Sets .....	783
	<b>19.1.5</b>	Relationship Between Sets and Dictionaries.....	783
<b>19.1</b>		Exercises.....	784
<b>19.2</b>		List Implementations of Sets and Dictionaries .....	784
	<b>19.2.1</b>	Sets.....	784
	<b>19.2.2</b>	Dictionaries .....	785
	<b>19.2.3</b>	Complexity Analysis of the List Implementations of Sets and Dictionaries.....	788
<b>19.2</b>		Exercises.....	789
<b>19.3</b>		Hashing Strategies .....	789
	<b>19.3.1</b>	The Relationship of Collisions to Density .....	790
	<b>19.3.2</b>	Hashing with Non-Numeric Keys .....	792
	<b>19.3.3</b>	Linear Probing .....	794
	<b>19.3.4</b>	Quadratic Probing.....	796
	<b>19.3.5</b>	Chaining .....	797
	<b>19.3.6</b>	Complexity Analysis.....	798
<b>19.3</b>		Exercises.....	800

19.4	Case Study: Profiling Hashing Strategies.....	800
19.4.1	Request .....	800
19.4.2	Analysis .....	800
19.4.3	Design .....	803
19.4.4	Implementation .....	803
19.5	Hashing Implementation of Dictionaries .....	806
19.5	Exercises.....	810
19.6	Hashing Implementation of Sets .....	811
19.6	Exercises.....	812
19.7	Sorted Sets and Dictionaries.....	813
	Summary .....	814
	Review Questions .....	815
	Projects.....	816
<b>[CHAPTER] 20</b>	<b>GRAPHS</b>	<b>819</b>
20.1	Graph Terminology.....	820
20.1	Exercises.....	824
20.2	Why Use Graphs? .....	824
20.3	Representations of Graphs .....	825
20.3.1	Adjacency Matrix.....	825
20.3.2	Adjacency List .....	826
20.3.3	Analysis of the Two Representations.....	828
20.3.4	Further Run-Time Considerations .....	829
20.3	Exercises.....	829
20.4	Graph Traversals.....	830
20.4.1	A Generic Traversal Algorithm .....	830
20.4.2	Breadth-First and Depth-First Traversals.....	831
20.4.3	Graph Components .....	834
20.4	Exercises.....	834
20.5	Trees Within Graphs.....	835
20.5.1	Spanning Trees and Forests.....	835
20.5.2	Minimum Spanning Tree.....	835
20.5.3	Algorithms for Minimum Spanning Trees.....	836
20.6	Topological Sort .....	838
20.7	The Shortest-Path Problem .....	840
20.7.1	Dijkstra's Algorithm .....	840
20.7.2	The Initialization Step .....	841
20.7.3	The Computation Step .....	842
20.7.4	Analysis .....	843
20.7	Exercises.....	843
20.8	Developing a Graph ADT .....	844
20.8.1	Example Use of the Graph ADT .....	844
20.8.2	The Class <code>LinkedDirectedGraph</code> .....	846
20.8.3	The Class <code>LinkedVertex</code> .....	850
20.8.4	The Class <code>LinkedEdge</code> .....	852

20.9

Case Study: Testing Graph Algorithms .....	853
20.9.1 Request .....	853
20.9.2 Analysis .....	853
20.9.3 The Classes <code>GraphDemoView</code> and <code>GraphDemoModel</code> .....	855
20.9.4 Implementation (Coding) .....	856
Summary .....	860
Review Questions .....	861
Projects.....	863

[APPENDIX] **A**

**PYTHON RESOURCES 865**

A.1  
A.2

Installing Python on Your Computer .....	866
Using the Terminal Command Prompt, IDLE, and Other IDEs.....	866

[APPENDIX] **B**

**INSTALLING THE `turtlegraphics` AND `images` LIBRARIES 869**

[APPENDIX] **C**

**APIS FOR GRAPHICS AND IMAGE PROCESSING 871**

C.1  
C.2

The <code>turtlegraphics</code> API.....	871
The <code>images</code> API.....	872

[APPENDIX] **D**

**TRANSITION FROM PYTHON TO JAVA AND C++ 875**

GLOSSARY/INDEX

877

