Preface

Welcome to *Starting Out with Python,* Second Edition. This book uses the Python language to teach programming concepts and problem-solving skills, without assuming any previous programming experience. With easy-to-understand examples, pseudocode, flowcharts, and other tools, the student learns how to design the logic of programs and then implement those programs using Python. This book is ideal for an introductory programming course or a programming logic and design course using Python as the language.

As with all the books in the *Starting Out With* series, the hallmark of this text is its clear, friendly, and easy-to-understand writing. In addition, it is rich in example programs that are concise and practical. The programs in this book include short examples that highlight specific programming topics, as well as more involved examples that focus on problem solving. Each chapter provides one or more case studies that provide step-by-step analysis of a specific problem and shows the student how to solve it.

Control Structures First, Then Classes

Python is a fully object-oriented programming language, but students do not have to understand object-oriented concepts to start programming in Python. This text first introduces the student to the fundamentals of data storage, input and output, control structures, functions, sequences and lists, file I/O, and objects that are created from standard library classes. Then the student learns to write classes, explores the topics of inheritance and polymorphism, and learns to write recursive functions. Finally, the student learns to develop simple event-driven GUI applications.

Changes in the Second Edition

This book's pedagogy, organization, and clear writing style remain the same as in the previous edition. However, many improvements have been made, which are summarized here:

- This edition is based on Python 3
- A series of online VideoNotes has been developed to accompany this book.
- Many examples of exploring topics with the interactive mode interpreter have been added throughout the book.
- The section covering nested loops in Chapter 5 has been enhanced with additional examples and an additional In the Spotlight section.

©2012 Pearson Education, Inc. Upper Saddle River, NJ. All Rights Reserved.

- The chapter on lists and strings has been split into two chapters, and the material on these topics has been enhanced. In this edition, Chapter 8 is Lists and Tuples, and Chapter 9 is More About Strings.
- Multidimensional lists are covered in this edition.
- A new chapter on dictionaries and sets has been added to this edition.
- Object serialization (pickling) is now covered.
- The material on exception handling in Chapter 7 has been expanded.
- Chapter 11, Classes and Object-Oriented Programming, has expanded material on passing objects as arguments, storing objects in dictionaries, and serializing (pickling) objects.

Brief Overview of Each Chapter

Chapter 1: Introduction to Computers and Programming

This chapter begins by giving a very concrete and easy-to-understand explanation of how computers work, how data is stored and manipulated, and why we write programs in high-level languages. An introduction to Python, interactive mode, script mode, and the IDLE environment is also given.

Chapter 2: Input, Processing, and Output

This chapter introduces the program development cycle, variables, data types, and simple programs that are written as sequence structures. The student learns to write simple programs that read input from the keyboard, perform mathematical operations, and produce screen output. Pseudocode and flowcharts are also introduced as tools for designing programs.

Chapter 3: Simple Functions

This chapter shows the benefits of modularizing programs and using the top-down design approach. The student learns to define and call simple functions (functions that do not return values), pass arguments to functions, and use local variables. Hierarchy charts are introduced as a design tool.

Chapter 4: Decision Structures and Boolean Logic

In this chapter the student learns about relational operators and Boolean expressions and is shown how to control the flow of a program with decision structures. The `if`, `if-else`, and `if-elif-else` statements are covered. Nested decision structures and logical operators are also discussed.

Chapter 5: Repetition Structures

This chapter shows the student how to create repetition structures using the `while` loop and `for` loop. Counters, accumulators, running totals, and sentinels are discussed, as well as techniques for writing input validation loops.

©2012 Pearson Education, Inc. Upper Saddle River, NJ. All Rights Reserved.

Chapter 6: Value-Returning Functions and Modules

This chapter begins by discussing common library functions, such as those for generating random numbers. After learning how to call library functions and use their return value, the student learns to define and call his or her own functions. Then the student learns how to use modules to organize functions.

Chapter 7: Files and Exceptions

This chapter introduces sequential file input and output. The student learns to read and write large sets of data and store data as fields and records. The chapter concludes by discussing exceptions and shows the student how to write exception-handling code.

Chapter 8: Lists and Tuples

This chapter introduces the student to the concept of a sequence in Python and explores the use of two common Python sequences: lists and tuples. The student learns to use lists for arraylike operations, such as storing objects in a list, iterating over a list, searching for items in a list, and calculating the sum and average of items in a list. The chapter discusses slicing and many of the list methods. One- and two-dimensional lists are covered.

Chapter 9: More About Strings

In this chapter the student learns to process strings at a detailed level. String slicing and algorithms that step through the individual characters in a string are discussed, and several built-in functions and string methods for character and text processing are introduced.

Chapter 10: Dictionaries and Sets

This chapter introduces the dictionary and set data structures. The student learns to store data as key-value pairs in dictionaries, search for values, change existing values, add new key-value pairs, and delete key-value pairs. The student learns to store values as unique elements in sets and perform common set operations such as union, intersection, difference, and symmetric difference. The chapter concludes with a discussion of object serialization and introduces the student to the Python `pickle` module.

Chapter 11: Classes and Object-Oriented Programming

This chapter compares procedural and object-oriented programming practices. It covers the fundamental concepts of classes and objects. Attributes, methods, encapsulation and data hiding, `__init__` functions (which are similar to constructors), accessors, and mutators are discussed. The student learns how to model classes with UML and how to find the classes in a particular problem.

Chapter 12: Inheritance

The study of classes continues in this chapter with the subjects of inheritance and polymorphism. The topics covered include superclasses, subclasses, how `__init__` functions work in inheritance, method overriding, and polymorphism.

©2012 Pearson Education, Inc. Upper Saddle River, NJ. All Rights Reserved.

Chapter 13: Recursion

This chapter discusses recursion and its use in problem solving. A visual trace of recursive calls is provided and recursive applications are discussed. Recursive algorithms for many tasks are presented, such as finding factorials, finding a greatest common denominator (GCD), and summing a range of values in a list, and the classic Towers of Hanoi example are presented.

Chapter 14: GUI Programming

This chapter discusses the basic aspects of designing a GUI application using the `tkinter` module in Python. Fundamental widgets, such as labels, button, entry fields, radio buttons, check buttons, and dialog boxes, are covered. The student also learns how events work in a GUI application and how to write callback functions to handle events.

Appendix A: Installing Python

This appendix explains how to download and install the Python 3 interpreter.

Appendix B: Introduction to IDLE

This appendix gives an overview of the IDLE integrated development environment that comes with Python.

Appendix C: The ASCII Character Set

As a reference, this appendix lists the ASCII character set.

Appendix D: Answers to Checkpoints

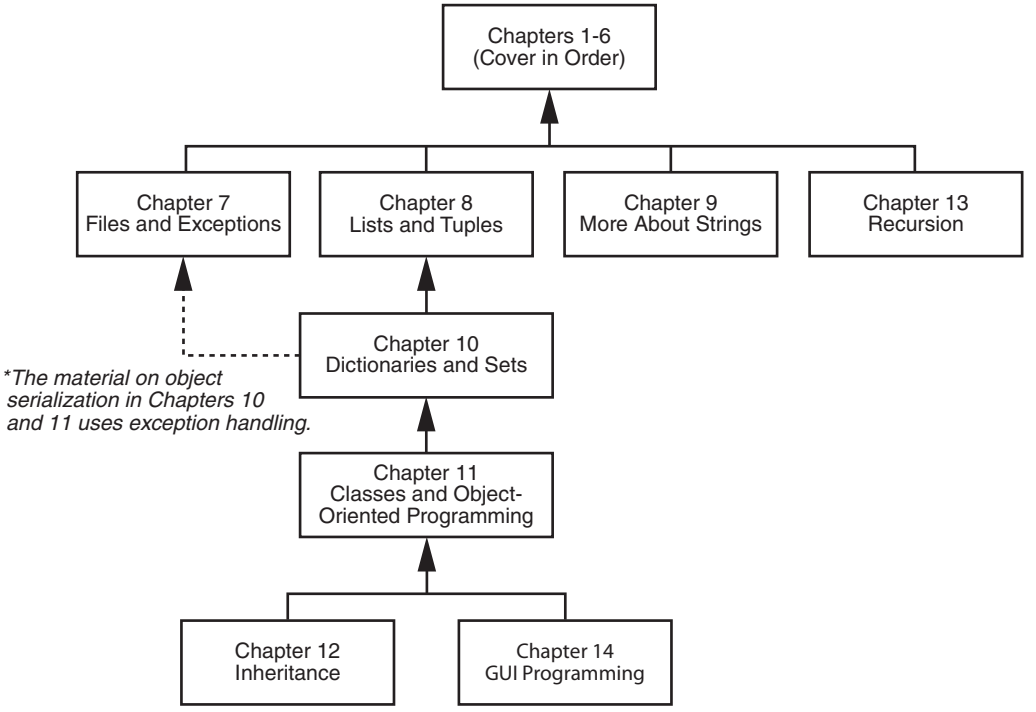
This appendix gives the answers to the Checkpoint questions that appear throughout the text.

Organization of the Text

The text teaches programming in a step-by-step manner. Each chapter covers a major set of topics and builds knowledge as students progress through the book. Although the chapters can be easily taught in their existing sequence, you do have some flexibility in the order that you wish to cover them. Figure P-1 shows chapter dependencies. Each box represents a chapter or a group of chapters. An arrow points from a chapter to the chapter that must be covered before it.

©2012 Pearson Education, Inc. Upper Saddle River, NJ. All Rights Reserved.

Figure P-1 Chapter dependencies



Features of the Text

Concept Statements Each major section of the text starts with a concept statement. This statement concisely summarizes the main point of the section.

Example Programs Each chapter has an abundant number of complete and partial example programs, each designed to highlight the current topic.

In the Spotlight Case Studies Each chapter has one or more In the Spotlight case studies that provide detailed, step-by-step analysis of problems and show the student how to solve them.

VideoNotes Online videos developed specifically for this book are available for viewing at `www.pearsonhighered.com/gaddis/videonotes`. Icons appear throughout the text alerting the student to videos about specific topics.

Notes Notes appear at several places throughout the text. They are short explanations of interesting or often misunderstood points relevant to the topic at hand.

Tips Tips advise the student on the best techniques for approaching different programming problems.

Warnings Warnings caution students about programming techniques or practices that can lead to malfunctioning programs or lost data.

©2012 Pearson Education, Inc. Upper Saddle River, NJ. All Rights Reserved.

Checkpoints	Checkpoints are questions placed at intervals throughout each chapter. They are designed to query the student's knowledge quickly after learning a new topic.
Review Questions	Each chapter presents a thorough and diverse set of review questions and exercises. They include Multiple Choice, True/False, Algorithm Workbench, and Short Answer.
Programming Exercises	Each chapter offers a pool of programming exercises designed to solidify the student's knowledge of the topics currently being studied.

Supplements

Student Online Resources

Many student resources are available for this book from the publisher. The following items are available on the Gaddis Series resource page at `www.pearsonhighered.com/gaddis`:

- The source code for each example program in the book
- Access to the book's companion VideoNotes

Instructor Resources

The following supplements are available to qualified instructors only:

- Answers to all of the Review Questions
- Solutions for the exercises
- PowerPoint presentation slides for each chapter
- Test bank

Visit the Addison-Wesley Instructor Resource Center (`www.pearsonhighered.com/irc`) or send an email to `computing@pearson.com` for information on how to access them.

Acknowledgments

I would like to thank the following faculty reviewers for their insight, expertise, and thoughtful recommendations:

Desmond K. H. Chun
Chabot Community College

Bob Husson
Craven Community College

Shyamal Mitra
University of Texas at Austin

Ken Robol
Beaufort Community College

Eric Shaffer
University of Illinois at Urbana-Champaign

Ann Ford Tyson
Florida State University

Linda F. Wilson
Texas Lutheran University

©2012 Pearson Education, Inc. Upper Saddle River, NJ. All Rights Reserved.

I would like to thank my family for their love and support in all my many projects. I would also like to thank Christopher Rich for his assistance in this revision. I am extremely fortunate to have Michael Hirsch as my editor and Stephanie Sellinger as editorial assistant. Michael's support and encouragement makes it a pleasure to write chapters and meet deadlines. I am also fortunate to have Yez Alayan as marketing manager and Kathryn Ferranti as marketing coordinator. They do a great job getting my books out to the academic community. I had a great production team led by Jeff Holcomb, Managing Editor, and Kayla Smith-Tarbox, Production Project Manager. Thanks to you all!

About the Author

Tony Gaddis is the principal author of the *Starting Out With* series of textbooks. Tony has nearly two decades of experience teaching computer science courses, primarily at Haywood Community College. He is a highly acclaimed instructor who was previously selected as the North Carolina Community College "Teacher of the Year" and has received the Teaching Excellence award from the National Institute for Staff and Organizational Development. The *Starting Out With* series includes introductory books covering C++, Java™, Microsoft® Visual Basic®, Microsoft® C#®, Python®, Programming Logic and Design, and Alice, all published by Addison-Wesley. More information about all these books can be found at `www.pearsonhighered.com/gaddisbooks`.

©2012 Pearson Education, Inc. Upper Saddle River, NJ. All Rights Reserved.