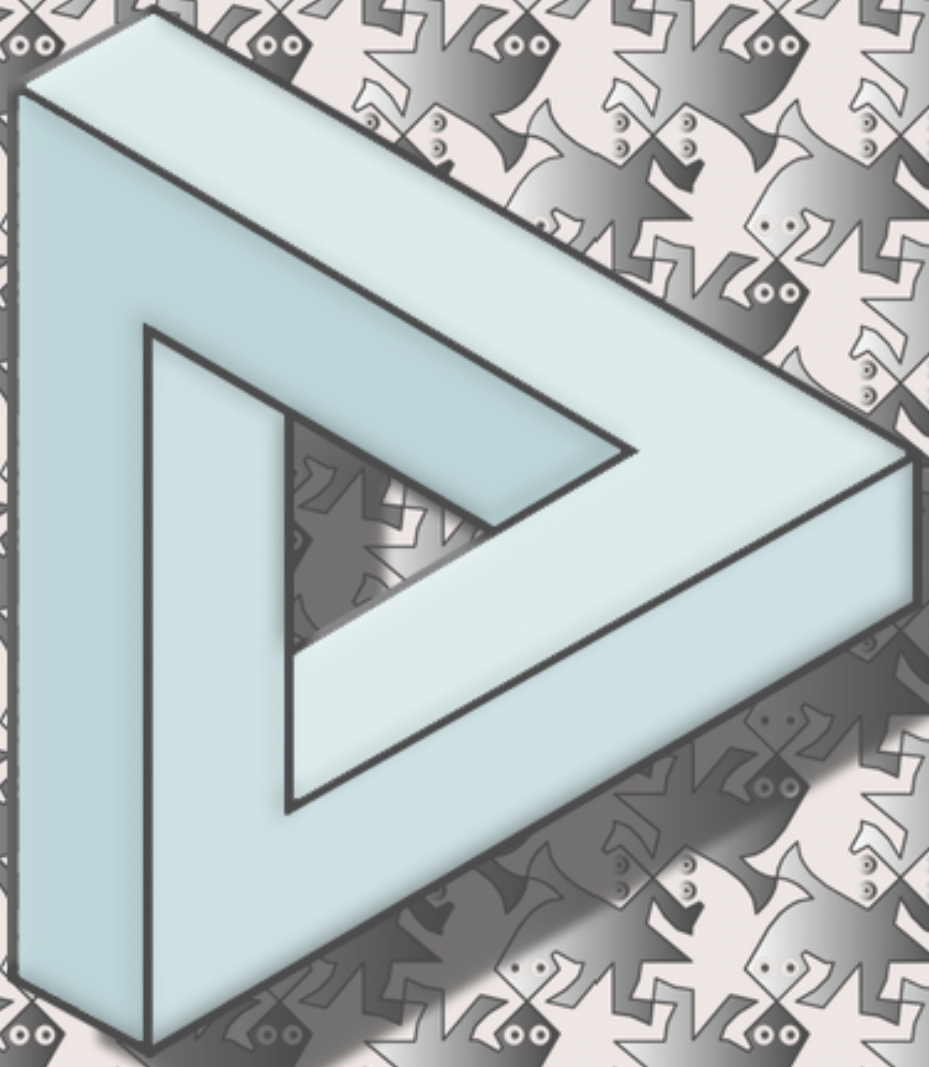
Objects and Games with Python

Margaret Burke



For access to code examples in the text, visit:

https://github.com/ProfessorBurke/PythonObjectsGames

Contact the author at mburke1@ccri.edu

Cover tessellation courtesy of publicdomainvectors.org

All other images created by the author unless otherwise noted.

ISBN: 978-0-578-55446-4

A note to the reader

Welcome to *Objects and Games with Python*! This book is for any beginner who wants to learn general programming, the object-oriented paradigm, or two-dimensional game programming. There are different paths that you can take through this book depending on what you want to learn.

- To learn object-oriented programming concepts, read Chapters 1-7.
- To learn game programming, read Chapters 1 and 2, followed by Chapters 8-11.
- To learn both, read Chapters 1-11.

Chapters 1 and 2 are an overview of foundational programming concepts and an introduction to the pygame game programming library, which is used throughout the book. You should not skip these chapters, even if you have programmed before. (At least skim them and take extra time with anything that is not familiar.) If you have never programmed before, you should take your time with Chapters 1 and 2, as they offer a light introduction to many concepts that beginning programmers sometimes find difficult.

Your author wishes that a book could teach you to be a programmer, but at best it will be your static, steadfast companion and guide. To learn to program, you must program. To achieve any mastery from this book, you should take full advantage of the *Try this*, *Skill building*, and *Challenge sections*. Ideally, read this book as part of a college course or with a friend, and discuss what you are reading and programming. Challenge yourself and challenge each other by asking questions about how Python works and by exploring systematically to find the answer.

There are other strategies you can use to make the most of this book, such as:

- Take notes and reorganize them after each chapter. In a book, concepts are introduced slowly. Once you have completed a chapter (or beyond), you will have seen "the big picture" and you should be able to write a concise summary of everything you know on a topic. You will probably find there are holes in your knowledge when you write your summary. Fill them in by re-reading or experimenting with Python.
- Create a "cheat sheet" of syntax and concepts and add to it after each section. This will be a useful reference for you as you program.
- Rewrite the examples in the book. Try to do this without looking at the book but refer to the book when you need to. If you do not understand what a line of code (or block of code) does, review that part of the chapter and try again.
- Desk-check the examples in the book. If you do not understand what a line of code does, review that part of the chapter. Run the program in the debugger to check your desk-checks. The debugger and desk-checking are covered in Chapter 1.
- Keep an engineering notebook. An engineering notebook is a notebook of the steps you took while you were designing, programming, or debugging a program. (Or installing software.) You will learn what detail is important to record and what detail is irrelevant. The notebook can be an invaluable tool in recreating steps that worked and in finding causes of bugs.

Most importantly, have fun learning Python, object-oriented programming and design, and game programming with pygame!

Contents

A note to the reader iii
Contents iv
Chapter 1. Python programming overview 1
1.1 Hello, Python! 1
Try this 4
1.2 Numeric data 4
Try this 9
1.3 Flow of control 9
1.4 String data 9
Try this 11
1.5 Input, output, and type casting 11
Try this 12
1.6 Comments and docstrings 12
Try this 13
1.7 Mathematical expressions 13
1.7.1 Mathematical operators 13
Try this 14
1.7.2 Order of operations 14
Try this 15
1.7.3 The math library 16
Try this 18
1.8 Importing and drawing images with pygame 18
Try this 21
1.9 Conditional statements 22
1.9.1 Forms of if statements 22
Try this 25
1.9.2 Relational operators and the Boolean data type 25
Try this 27
1.9.3 Comparing strings 27
Try this 28
1.9.4 Boolean variables 28
Try this 29
1.9.5 Boolean operators 29
Try this 30
1.9.6 Order of operations revisited 30
Try this 31
1.10 Generating random numbers 31
Try this 32
1.11 Repetition structures 32
1.11.1 While loops 32
Try this 37
1.11.2 Input validation loops 37
Try this 38

1.11.3 Augmented assignment operators 38
Try this 39
1.11.4 Loop patterns 39
Try this 43
1.11.5 For loops 43
Try this 46
1.11.6 Event loops 46
Try this 50
1.12 Testing, debugging, and desk-checking 50
1.12.1 Testing 50
1.12.2 Debugging 51
1.12.3 Desk-checking 54
Try this 55
Skill building 55
Challenge 57
Chapter 2. Functional decomposition and collections 59
2.1 Functions 59
2.1.1 Function syntax and flow of control 59
Try this 63
2.1.2 Functional decomposition 63
Try this 69
2.1.3 Input validation functions 69
Try this 72
2.1.4 Scope 72
Try this 76
2.2 Lists 76
2.2.1 The list data type 76
2.2.2 Lists are objects 78
2.2.3 Iterating over lists 79
Try this 81
2.2.4 Operations on lists 81
Try this 84
Try this 86
Try this 86
Try this 88
2.2.5 Passing objects as parameters 88
Try this 90
2.2.6 Multi-dimensional lists 90
Try this 94
2.3 Dictionaries 94
Try this 98
2.4 More on strings 98
2.4.1 String formatting 98
Try this 100
2.4.1 String methods 100
Try this 104
2.5 Files 104

2.5.1 Writing to files ... 104
Try this ... 106
2.5.2 Reading from files ... 106
Try this ... 110
2.5.3 More about files ... 110
2.5.4 Exceptions ... 111
Try this ... 113
Skill building ... 113
Challenge ... 115
Chapter 3. Objects and classes ... 117
3.1 Objects and classes ... 117
Try this ... 125
3.2 Methods ... 125
3.2.1 Accessors, mutators, and initializers ... 125
3.2.2 Persistence and garbage collection ... 131
3.2.3 Sequentially searching a list of objects ... 131
Try this ... 132
3.3 Information hiding, scope, and visibility ... 132
3.3.1 Information hiding ... 132
Try this ... 134
3.3.2 Visibility design decisions ... 135
Try this ... 139
Skill building ... 139
Challenge ... 142
Chapter 4. Designing with objects ... 144
4.1 Associations between objects ... 144
4.1.1 Bidirectional navigation associations ... 145
4.1.2 Single-direction navigation associations ... 145
4.1.3 Composition associations ... 145
4.1.4 Association classes ... 146
Try this ... 150
4.2 Class-level field attributes ... 150
Try this ... 155
4.3 Use cases ... 155
Try this ... 158
4.4 Collaborations and responsibilities ... 158
4.4.1 Candidate objects in the *Request passport stamp* use case ... 159
4.4.2 The `ReaderAccount`, `Passport`, and `Stamp` objects ... 159
4.4.3 A sequence diagram of the *Request passport stamp* use case ... 161
4.4.4 Inventing a class and revising a use case ... 163
4.4.5 Implementing and testing our classes ... 164
Try this ... 171
Skill building ... 171
Challenge ... 172
Chapter 5. Inheritance and polymorphism ... 174
5.1 Inheritance ... 174
Try this ... 179

5.2 Visibility in inheritance hierarchies 179
Try this 180
5.3 Polymorphism, overriding methods, and dynamic binding 180
Try this 185
5.4 Abstract and concrete base classes 185
Try this 188
Skill building 189
Challenge 190
Chapter 6. Designing object-oriented systems 192
6.1 Binary files 192
6.1.1 Binary files and the pickle library 192
6.1.2 Pickling special cases 195
Try this 198
6.2 Design patterns 198
6.2.1 Command pattern 198
Try this 215
6.2.2 Iterator pattern 215
Try this 228
6.2.3 Observer pattern 228
Skill building 242
Challenge 243
Chapter 7 Multiple inheritance 245
7.1 Overview of multiple inheritance 245
Try this 250
7.2 Resolving naming conflicts and invoking super() 250
7.2.1 Naming conflicts 250
7.2.2 Invoking superclass methods 252
Try this 254
7.3 Keyword arguments and positional arguments 255
Try this 257
7.4 A multiple inheritance example with a mixin 258
Try this 270
Skill building 270
Challenge 271
Chapter 8. Events and Surfaces 273
8.1 Events 273
8.1.1 Event overview 273
8.1.2 Mouse and system events 276
Try this 283
8.1.3 Keyboard events 283
Try this 289
8.2 Surfaces and drawing 289
8.2.1 Review of Surface operations so far 289
8.2.2 More Surface properties 290
Try this 300
8.2.3 Drawing to Surface objects 300
8.2.4 Transforming Surface objects 306

Try this 310
Skill building 310
Challenge 311
Chapter 9. Motion 312
9.1 Review of the coordinate system and motion 312
9.2 Boundary checking 314
Try this 316
9.3 Calculations for motion 316
9.3.1 The distance formula 316
9.3.2 Vectors 316
9.3.3 Unit vectors 320
Try this 324
9.4 Projectile Motion 324
9.4.1 Falling 325
9.4.2 Projectiles 327
Try this 334
9.5 Drag 334
Try this 338
9.6 Scrolling backgrounds 338
Try this 345
9.7 Reading terrain from a file 345
Try this 347
Skill building 347
Challenge 348
Chapter 10. Sprites, collisions, and sound 349
10.1 Sprites and groups 349
Try this 355
10.2 More about sprites and groups 355
10.2.1 Groups with many sprites 355
10.2.2 Layering groups 357
10.2.3 Changing sprite appearance 360
10.2.4 Player-controlled sprites 362
Try this 364
Try this, continued 364
10.2.5 Sprites with floating-point calculated motion 364
Try this 368
10.3 Collisions 368
10.3.1 Basic collision detection with rectangles 369
10.3.2 Collision detection with circles and masks 372
10.3.3 Collisions between groups 374
Try this 376
10.4 Terrain and collisions 376
10.4.1 Terrain sprites 376
10.4.2 Adding scrolling 383
10.4.3 Platform collisions 387
Try this 393
10.5 Using sprite sheets 393

Try this 399
10.6 Sound and music 399
Try this 404
Skill building 404
Challenge 405
11. Organization, levels, and scoring 406
11.1 Scoring 406
11.1.1 Fonts 406
11.1.2 Scorekeeper sprite 409
Try this 413
11.2 Levels 413
Try this 428
11.3 Organization 428
11.3.1 Modules and packages 428
11.3.2 Directory structure for assets and using `os.path.join` 430
Try this 431
Skill building 431
Challenge 432
References 433
Index 434