

Contents

Foreword	v
Preface	vii
Contributor List	xiii
1 The way of the program	1
1.1 The Python programming language	1
1.2 What is a program?	3
1.3 What is debugging?	4
1.4 Formal and natural languages	6
1.5 The first program	8
1.6 Glossary	8
2 Variables, expressions and statements	11
2.1 Values and types	11
2.2 Variables	12
2.3 Variable names and keywords	13
2.4 Statements	15
2.5 Evaluating expressions	16
2.6 Operators and operands	17

2.7	Order of operations	17
2.8	Operations on strings	18
2.9	Composition	19
2.10	Comments	19
2.11	Glossary	20
3	Functions	23
3.1	Function calls	23
3.2	Type conversion	24
3.3	Type coercion	24
3.4	Math functions	25
3.5	Composition	26
3.6	Adding new functions	26
3.7	Definitions and use	29
3.8	Flow of execution	29
3.9	Parameters and arguments	30
3.10	Variables and parameters are local	31
3.11	Stack diagrams	32
3.12	Functions with results	33
3.13	Glossary	34
4	Conditionals and recursion	37
4.1	The modulus operator	37
4.2	Boolean expressions	37
4.3	Logical operators	38
4.4	Conditional execution	39
4.5	Alternative execution	39
4.6	Chained conditionals	40

4.7	Nested conditionals	41
4.8	The <code>return</code> statement	42
4.9	Recursion	42
4.10	Stack diagrams for recursive functions	44
4.11	Infinite recursion	45
4.12	Keyboard input	45
4.13	Glossary	46
5	Fruitful functions	49
5.1	Return values	49
5.2	Program development	50
5.3	Composition	53
5.4	Boolean functions	54
5.5	More recursion	55
5.6	Leap of faith	57
5.7	One more example	58
5.8	Checking types	58
5.9	Glossary	60
6	Iteration	61
6.1	Multiple assignment	61
6.2	The <code>while</code> statement	62
6.3	Tables	64
6.4	Two-dimensional tables	66
6.5	Encapsulation and generalization	67
6.6	More encapsulation	68
6.7	Local variables	69
6.8	More generalization	70
6.9	Functions	71
6.10	Glossary	72

7	Strings	73
7.1	A compound data type	73
7.2	Length	74
7.3	Traversal and the <code>for</code> loop	74
7.4	String slices	76
7.5	String comparison	76
7.6	Strings are immutable	77
7.7	A <code>find</code> function	78
7.8	Looping and counting	78
7.9	The <code>string</code> module	79
7.10	Character classification	80
7.11	Glossary	81
8	Lists	83
8.1	List values	83
8.2	Accessing elements	84
8.3	List length	85
8.4	List membership	86
8.5	Lists and <code>for</code> loops	86
8.6	List operations	87
8.7	List slices	88
8.8	Lists are mutable	88
8.9	List deletion	89
8.10	Objects and values	91
8.11	Aliasing	92
8.12	Cloning lists	92
8.13	List parameters	93
8.14	Nested lists	94

8.15	Matrices	94
8.16	Strings and lists	95
8.17	Glossary	96
9	Tuples	97
9.1	Mutability and tuples	97
9.2	Tuple assignment	98
9.3	Tuples as return values	99
9.4	Random numbers	99
9.5	List of random numbers	100
9.6	Counting	101
9.7	Many buckets	102
9.8	A single-pass solution	104
9.9	Glossary	105
10	Dictionaries	107
10.1	Dictionary operations	108
10.2	Dictionary methods	109
10.3	Aliasing and copying	110
10.4	Sparse matrices	110
10.5	Hints	111
10.6	Long integers	113
10.7	Counting letters	113
10.8	Glossary	114
11	Files and exceptions	117
11.1	Text files	119
11.2	Writing variables	120
11.3	Directories	123

11.4	Pickling	123
11.5	Exceptions	124
11.6	Glossary	126
12	Classes and objects	129
12.1	User-defined compound types	129
12.2	Attributes	130
12.3	Instances as arguments	131
12.4	Sameness	131
12.5	Rectangles	133
12.6	Instances as return values	134
12.7	Objects are mutable	134
12.8	Copying	135
12.9	Glossary	137
13	Classes and functions	139
13.1	Time	139
13.2	Pure functions	140
13.3	Modifiers	141
13.4	Which is better?	142
13.5	Prototype development versus planning	143
13.6	Generalization	144
13.7	Algorithms	144
13.8	Glossary	145
14	Classes and methods	147
14.1	Object-oriented features	147
14.2	<code>printTime</code>	148
14.3	Another example	149

14.4	A more complicated example	150
14.5	Optional arguments	151
14.6	The initialization method	152
14.7	Points revisited	153
14.8	Operator overloading	154
14.9	Polymorphism	155
14.10	Glossary	157
15	Sets of objects	159
15.1	Composition	159
15.2	Card objects	159
15.3	Class attributes and the <code>__str__</code> method	161
15.4	Comparing cards	162
15.5	Decks	163
15.6	Printing the deck	163
15.7	Shuffling the deck	165
15.8	Removing and dealing cards	166
15.9	Glossary	167
16	Inheritance	169
16.1	Inheritance	169
16.2	A hand of cards	170
16.3	Dealing cards	171
16.4	Printing a Hand	171
16.5	The <code>CardGame</code> class	172
16.6	<code>OldMaidHand</code> class	173
16.7	<code>OldMaidGame</code> class	175
16.8	Glossary	179

17	Linked lists	181
17.1	Embedded references	181
17.2	The <code>Node</code> class	181
17.3	Lists as collections	183
17.4	Lists and recursion	184
17.5	Infinite lists	185
17.6	The fundamental ambiguity theorem	186
17.7	Modifying lists	186
17.8	Wrappers and helpers	187
17.9	The <code>LinkedList</code> class	188
17.10	Invariants	189
17.11	Glossary	190
18	Stacks	191
18.1	Abstract data types	191
18.2	The Stack ADT	192
18.3	Implementing stacks with Python lists	192
18.4	Pushing and popping	193
18.5	Using a stack to evaluate postfix	194
18.6	Parsing	194
18.7	Evaluating postfix	195
18.8	Clients and providers	196
18.9	Glossary	197
19	Queues	199
19.1	The Queue ADT	199
19.2	Linked Queue	200
19.3	Performance characteristics	201

19.4	Improved Linked Queue	201
19.5	Priority queue	203
19.6	The <code>Golfer</code> class	205
19.7	Glossary	206
20	Trees	207
20.1	Building trees	208
20.2	Traversing trees	209
20.3	Expression trees	209
20.4	Tree traversal	210
20.5	Building an expression tree	212
20.6	Handling errors	216
20.7	The animal tree	216
20.8	Glossary	219
A	Debugging	221
A.1	Syntax errors	221
A.2	Runtime errors	223
A.3	Semantic errors	227
B	Creating a new data type	231
B.1	Fraction multiplication	232
B.2	Fraction addition	234
B.3	Euclid's algorithm	234
B.4	Comparing fractions	235
B.5	Taking it further	236
B.6	Glossary	236
C	Recommendations for further reading	239
C.1	Python-related web sites and books	240
C.2	Recommended general computer science books	241

