

Data Structures and Algorithms Using Python

Rance D. Necaise

Department of Computer Science
College of William and Mary



WILEY

JOHN WILEY & SONS, INC.

Contents

Preface	xiii
Chapter 1: Abstract Data Types	1
1.1 Introduction	1
1.1.1 Abstractions	2
1.1.2 Abstract Data Types	3
1.1.3 Data Structures	5
1.1.4 General Definitions	6
1.2 The Date Abstract Data Type	7
1.2.1 Defining the ADT	7
1.2.2 Using the ADT	8
1.2.3 Preconditions and Postconditions	9
1.2.4 Implementing the ADT	10
1.3 Bags	14
1.3.1 The Bag Abstract Data Type	15
1.3.2 Selecting a Data Structure	17
1.3.3 List-Based Implementation	19
1.4 Iterators	20
1.4.1 Designing an Iterator	21
1.4.2 Using Iterators	22
1.5 Application: Student Records	23
1.5.1 Designing a Solution	23
1.5.2 Implementation	26
Exercises	28
Programming Projects	29
Chapter 2: Arrays	33
2.1 The Array Structure	33
2.1.1 Why Study Arrays?	34
2.1.2 The Array Abstract Data Type	34
2.1.3 Implementing the Array	36
2.2 The Python List	41

2.2.1	Creating a Python List	41
2.2.2	Appending Items	42
2.2.3	Extending A List	44
2.2.4	Inserting Items	44
2.2.5	List Slice	45
2.3	Two-Dimensional Arrays	47
2.3.1	The Array2D Abstract Data Type	47
2.3.2	Implementing the 2-D Array	49
2.4	The Matrix Abstract Data Type	52
2.4.1	Matrix Operations	53
2.4.2	Implementing the Matrix	55
2.5	Application: The Game of Life	57
2.5.1	Rules of the Game	57
2.5.2	Designing a Solution	59
2.5.3	Implementation	61
	Exercises	64
	Programming Projects	65
Chapter 3: Sets and Maps		69
3.1	Sets	69
3.1.1	The Set Abstract Data Type	70
3.1.2	Selecting a Data Structure	72
3.1.3	List-Based Implementation	72
3.2	Maps	75
3.2.1	The Map Abstract Data Type	76
3.2.2	List-Based Implementation	77
3.3	Multi-Dimensional Arrays	80
3.3.1	The MultiArray Abstract Data Type	81
3.3.2	Data Organization	81
3.3.3	Variable-Length Arguments	85
3.3.4	Implementing the MultiArray	86
3.4	Application: Sales Reports	89
	Exercises	95
	Programming Projects	96
Chapter 4: Algorithm Analysis		97
4.1	Complexity Analysis	97
4.1.1	Big-O Notation	99
4.1.2	Evaluating Python Code	104
4.2	Evaluating the Python List	108
4.3	Amortized Cost	111
4.4	Evaluating the Set ADT	113

4.5	Application: The Sparse Matrix	115
4.5.1	List-Based Implementation	115
4.5.2	Efficiency Analysis	120
	Exercises	121
	Programming Projects	122
 Chapter 5: Searching and Sorting		 125
5.1	Searching	125
5.1.1	The Linear Search	126
5.1.2	The Binary Search	128
5.2	Sorting	131
5.2.1	Bubble Sort	132
5.2.2	Selection Sort	136
5.2.3	Insertion Sort	138
5.3	Working with Sorted Lists	142
5.3.1	Maintaining a Sorted List	142
5.3.2	Merging Sorted Lists	143
5.4	The Set ADT Revisited	147
5.4.1	A Sorted List Implementation	147
5.4.2	Comparing the Implementations	152
	Exercises	152
	Programming Projects	153
 Chapter 6: Linked Structures		 155
6.1	Introduction	156
6.2	The Singly Linked List	159
6.2.1	Traversing the Nodes	159
6.2.2	Searching for a Node	161
6.2.3	Prepending Nodes	162
6.2.4	Removing Nodes	163
6.3	The Bag ADT Revisited	165
6.3.1	A Linked List Implementation	165
6.3.2	Comparing Implementations	167
6.3.3	Linked List Iterators	168
6.4	More Ways to Build a Linked List	169
6.4.1	Using a Tail Reference	169
6.4.2	The Sorted Linked List	171
6.5	The Sparse Matrix Revisited	174
6.5.1	An Array of Linked Lists Implementation	175
6.5.2	Comparing the Implementations	178
6.6	Application: Polynomials	179
6.6.1	Polynomial Operations	179

6.6.2	The Polynomial ADT	181
6.6.3	Implementation	181
	Exercises	189
	Programming Projects	190
Chapter 7: Stacks		193
7.1	The Stack ADT	193
7.2	Implementing the Stack	195
7.2.1	Using a Python List	195
7.2.2	Using a Linked List	196
7.3	Stack Applications	198
7.3.1	Balanced Delimiters	199
7.3.2	Evaluating Postfix Expressions	202
7.4	Application: Solving a Maze	206
7.4.1	Backtracking	207
7.4.2	Designing a Solution	208
7.4.3	The Maze ADT	211
7.4.4	Implementation	214
	Exercises	218
	Programming Projects	219
Chapter 8: Queues		221
8.1	The Queue ADT	221
8.2	Implementing the Queue	222
8.2.1	Using a Python List	222
8.2.2	Using a Circular Array	224
8.2.3	Using a Linked List	228
8.3	Priority Queues	230
8.3.1	The Priority Queue ADT	230
8.3.2	Implementation: Unbounded Priority Queue	232
8.3.3	Implementation: Bounded Priority Queue	235
8.4	Application: Computer Simulations	237
8.4.1	Airline Ticket Counter	237
8.4.2	Implementation	239
	Exercises	244
	Programming Projects	246
Chapter 9: Advanced Linked Lists		247
9.1	The Doubly Linked List	247
9.1.1	Organization	247
9.1.2	List Operations	248
9.2	The Circular Linked List	253

9.2.1	Organization	253
9.2.2	List Operations	254
9.3	Multi-Linked Lists	259
9.3.1	Multiple Chains	259
9.3.2	The Sparse Matrix	260
9.4	Complex Iterators	262
9.5	Application: Text Editor	263
9.5.1	Typical Editor Operations	263
9.5.2	The Edit Buffer ADT	266
9.5.3	Implementation	268
	Exercises	275
	Programming Projects	275
Chapter 10:	Recursion	277
10.1	Recursive Functions	277
10.2	Properties of Recursion	279
10.2.1	Factorials	280
10.2.2	Recursive Call Trees	281
10.2.3	The Fibonacci Sequence	283
10.3	How Recursion Works	283
10.3.1	The Run Time Stack	284
10.3.2	Using a Software Stack	286
10.3.3	Tail Recursion	289
10.4	Recursive Applications	290
10.4.1	Recursive Binary Search	290
10.4.2	Towers of Hanoi	292
10.4.3	Exponential Operation	296
10.4.4	Playing Tic-Tac-Toe	297
10.5	Application: The Eight-Queens Problem	299
10.5.1	Solving for Four-Queens	301
10.5.2	Designing a Solution	303
	Exercises	307
	Programming Projects	308
Chapter 11:	Hash Tables	309
11.1	Introduction	309
11.2	Hashing	311
11.2.1	Linear Probing	312
11.2.2	Clustering	315
11.2.3	Rehashing	318
11.2.4	Efficiency Analysis	320
11.3	Separate Chaining	321

11.4 Hash Functions	323
11.5 The HashMap Abstract Data Type	325
11.6 Application: Histograms	330
11.6.1 The Histogram Abstract Data Type	330
11.6.2 The Color Histogram	334
Exercises	337
Programming Projects	338
Chapter 12: Advanced Sorting	339
12.1 Merge Sort	339
12.1.1 Algorithm Description	340
12.1.2 Basic Implementation	340
12.1.3 Improved Implementation	342
12.1.4 Efficiency Analysis	345
12.2 Quick Sort	347
12.2.1 Algorithm Description	348
12.2.2 Implementation	349
12.2.3 Efficiency Analysis	353
12.3 How Fast Can We Sort?	353
12.4 Radix Sort	354
12.4.1 Algorithm Description	354
12.4.2 Basic Implementation	356
12.4.3 Efficiency Analysis	358
12.5 Sorting Linked Lists	358
12.5.1 Insertion Sort	359
12.5.2 Merge Sort	362
Exercises	367
Programming Projects	368
Chapter 13: Binary Trees	369
13.1 The Tree Structure	369
13.2 The Binary Tree	373
13.2.1 Properties	373
13.2.2 Implementation	375
13.2.3 Tree Traversals	376
13.3 Expression Trees	380
13.3.1 Expression Tree Abstract Data Type	382
13.3.2 String Representation	383
13.3.3 Tree Evaluation	384
13.3.4 Tree Construction	386
13.4 Heaps	390
13.4.1 Definition	391

13.4.2 Implementation	395
13.4.3 The Priority Queue Revisited	398
13.5 Heapsort	400
13.5.1 Simple Implementation	400
13.5.2 Sorting In Place	400
13.6 Application: Morse Code	404
13.6.1 Decision Trees	405
13.6.2 The ADT Definition	406
Exercises	407
Programming Projects	410
Chapter 14: Search Trees	411
14.1 The Binary Search Tree	412
14.1.1 Searching	413
14.1.2 Min and Max Values	415
14.1.3 Insertions	417
14.1.4 Deletions	420
14.1.5 Efficiency of Binary Search Trees	425
14.2 Search Tree Iterators	427
14.3 AVL Trees	428
14.3.1 Insertions	430
14.3.2 Deletions	433
14.3.3 Implementation	435
14.4 The 2-3 Tree	440
14.4.1 Searching	442
14.4.2 Insertions	443
14.4.3 Efficiency of the 2-3 Tree	449
Exercises	451
Programming Projects	452
Appendix A: Python Review	453
A.1 The Python Interpreter	453
A.2 The Basics of Python	454
A.2.1 Primitive Types	455
A.2.2 Statements	456
A.2.3 Variables	457
A.2.4 Arithmetic Operators	458
A.2.5 Logical Expressions	459
A.2.6 Using Functions and Methods	461
A.2.7 Standard Library	462
A.3 User Interaction	463
A.3.1 Standard Input	463

A.3.2	Standard Output	464
A.4	Control Structures	467
A.4.1	Selection Constructs	467
A.4.2	Repetition Constructs	469
A.5	Collections	472
A.5.1	Strings	472
A.5.2	Lists	473
A.5.3	Tuples	475
A.5.4	Dictionaries	475
A.6	Text Files	477
A.6.1	File Access	477
A.6.2	Writing to Files	478
A.6.3	Reading from Files	479
A.7	User-Defined Functions	480
A.7.1	The Function Definition	480
A.7.2	Variable Scope	483
A.7.3	Main Routine	483
Appendix B: User-Defined Modules		485
B.1	Structured Programs	485
B.2	Namespaces	486
Appendix C: Exceptions		489
C.1	Catching Exceptions	489
C.2	Raising Exceptions	490
C.3	Standard Exceptions	491
C.4	Assertions	491
Appendix D: Classes		493
D.1	The Class Definition	493
D.1.1	Constructors	494
D.1.2	Operations	495
D.1.3	Using Modules	497
D.1.4	Hiding Attributes	498
D.2	Overloading Operators	500
D.3	Inheritance	502
D.3.1	Deriving Child Classes	503
D.3.2	Creating Class Instances	504
D.3.3	Invoking Methods	505
D.4	Polymorphism	507