

Data Structures and Algorithms using Python

Efficiently using data structures to collect, organize, and retrieve information is one of the core abilities modern computer engineers are expected to have. Python currently is one of the most popular programming languages, and as such, it has become vital for students to understand this concept in this language.

This student-friendly textbook provides a complete view of data structures and algorithms using the Python programming language, striking a balance between theory and practical application. All major algorithms have been discussed and analysed in detail, and the corresponding codes in Python have been provided. Diagrams and examples have been extensively used for better understanding. Running time complexities are also discussed for each algorithm, allowing the student to better understand how to select the appropriate one.

The book has been written with both undergraduate and graduate students in mind. Each chapter ends with a large number of problems, including multiple choice questions, to help consolidate the knowledge gained. This will also be helpful with competitive examinations for engineering in India such as GATE and NET. As such, the book will be a vital resource for students as well as professionals who are looking for a handbook on data structures in Python.

Subrata Saha is Head of the Department of Computer Applications at Techno India Hooghly, West Bengal. He has more than 20 years of teaching experience in various subjects in computer science and engineering, including data structures and algorithms, programming in C, C++, Java, and Python, basic computation, operating systems, and so on. He has previously published *Basic Computations and Programming with C* with the Cambridge University Press in 2016.

Cambridge University Press & Assessment
978-1-009-27697-9 – Data Structures and Algorithms Using Python
Subrata Saha
Frontmatter
[More Information](#)

Data Structures and Algorithms using Python

Subrata Saha



CAMBRIDGE
UNIVERSITY PRESS

University Printing House, Cambridge CB2 8BS, United Kingdom
One Liberty Plaza, 20th Floor, New York, NY 10006, USA
477 Williamstown Road, Port Melbourne, vic 3207, Australia
314 to 321, 3rd Floor, Plot No.3, Splendor Forum, Jasola District Centre, New Delhi 110025, India
103 Penang Road, #05–06/07, Visioncrest Commercial, Singapore 238467

Cambridge University Press is part of the University of Cambridge.

It furthers the University's mission by disseminating knowledge in the pursuit of education, learning and research at the highest international levels of excellence.

www.cambridge.org

Information on this title: www.cambridge.org/9781009276979

© Subrata Saha 2023

This publication is in copyright. Subject to statutory exception and to the provisions of relevant collective licensing agreements, no reproduction of any part may take place without the written permission of Cambridge University Press.

First published 2023

Printed in India

A catalogue record for this publication is available from the British Library

ISBN 978-1-009-27697-9 Paperback

Cambridge University Press has no responsibility for the persistence or accuracy of URLs for external or third-party internet websites referred to in this publication, and does not guarantee that any content on such websites is, or will remain, accurate or appropriate.

*To my beloved wife,
Mrs Sriparna Saha, and my princess, Miss Shreya
Saha, whose constant support motivated me to
write this book*

Cambridge University Press & Assessment
978-1-009-27697-9 – Data Structures and Algorithms Using Python
Subrata Saha
Frontmatter
[More Information](#)

Contents

<i>Preface</i>	xxi
<i>Acknowledgments</i>	xxiii
1. DATA STRUCTURE PRELIMINARIES	1
1.1 Concept of Data Type	1
1.1.1 Primitive data type	1
1.1.2 User defined data type	2
1.1.3 Abstract data type	2
1.2 What Is Data Structure?	2
1.3 Definition and Brief Description of Various Data Structures	2
1.3.1 Array	3
1.3.2 Linked list	4
1.3.3 Stack	4
1.3.4 Queue	5
1.3.5 Graph	6
1.3.6 Tree	7
1.3.7 Heap	9
1.4 Data Structures versus Data Types	9
1.5 Operations on Data Structures	10
<i>Data Structure Preliminaries At a Glance</i>	11
<i>Multiple Choice Questions</i>	11
<i>Review Exercises</i>	14

2. INTRODUCTION TO ALGORITHM	15
2.1 What Is an Algorithm?	15
2.2 Importance of an Algorithm	16
2.3 Different Approaches to Designing an Algorithm	16
2.4 Algorithm Design Tools: Flowchart and Pseudocode	16
2.4.1 Flowchart	17
2.4.2 Pseudocode	19
2.5 Control Structures Used in an Algorithm	20
2.6 Time and space complexity	24
2.7 Best Case, Worst Case, Average Case Time Complexity	24
2.8 Time–Space Trade-off	25
2.9 Frequency Count and Its Importance	25
2.10 Analyzing Algorithms	27
2.10.1 Big O Notation	27
2.10.2 Ω (Omega) Notation	29
2.10.3 Θ (Theta) Notation	30
2.10.4 Other Useful Notations	30
2.11 Divide and Conquer Strategy	31
2.12 Dynamic Programming	32
2.13 Greedy Method	34
<i>Introduction to Algorithm At a glance</i>	35
<i>Multiple Choice Questions</i>	36
<i>Review Exercises</i>	39
3. ARRAY	41
3.1 Definition	41
3.2 Creating an Array	41
3.3 Accessing Elements of an Array	45
3.4 Operations on an Array	48
3.4.1 Adding Elements to an Array	48
3.4.2 Removing Elements from an Array	49
3.4.3 Slicing of an Array	51
3.4.4 Searching Element in an Array	53
3.4.5 Updating Elements in an Array	54
3.4.6 Concatenation of Arrays	55

3.4.7	Multiplication or Repetition on Array	56
3.5	Representation of Polynomials	57
3.6	Two Dimensional Array	59
3.7	Creation of a Two Dimensional Array	59
3.8	Accessing Elements of a Two Dimensional Array	62
3.9	Representation of a Two Dimensional Array in Memory	63
3.9.1	Row Major Representation	63
3.9.2	Column Major Representation	64
3.10	Operations on Two Dimensional Arrays	65
3.10.1	Matrix Addition	65
3.10.2	Matrix Multiplication	66
3.10.3	Transpose of a Matrix	66
3.10.4	Slicing of a Matrix	66
3.11	Sparse Matrix	68
3.12	Programming Examples	74
	<i>Array at a Glance</i>	79
	<i>Multiple Choice Questions</i>	79
	<i>Review Exercises</i>	81
	<i>Problems for Programming</i>	84
4.	PYTHON DATA STRUCTURES	85
4.1	Lists	85
4.1.1	Creating a List	85
4.1.2	Accessing List Elements	86
4.1.3	Operations on a List	88
4.1.3.1	Adding Elements to a List	88
4.1.3.2	Removing Elements from a List	89
4.1.3.3	Slicing of a List	91
4.1.3.4	Searching Element in a List	93
4.1.3.5	Updating Elements in a List	94
4.1.3.6	Concatenation of Lists	95
4.1.3.7	Multiplication or Repetition of List	96
4.1.4	Nested List	97
4.1.5	List Functions	97
4.1.6	List Methods	98

4.1.7	Looping in a List	99
4.1.8	List vs. Array	100
4.2	Tuples	101
4.2.1	Creating a Tuple	101
4.2.2	Accessing Tuple Elements	102
4.2.3	Operations on Tuple	103
4.2.4	Nested Tuples	105
4.2.5	Tuple Methods	105
4.2.6	Looping in a Tuple	106
4.3	Sets	107
4.3.1	Creating a Set	108
4.3.2	Operations on a Set	108
4.3.2.1	Adding Elements to a Set	109
4.3.2.2	Removing Elements from a Set	109
4.3.2.3	Searching an Element in a Set	110
4.3.3	Set Methods	111
4.3.4	Frozenset	114
4.4	Dictionaries	115
4.4.1	Creating a Dictionary	115
4.4.2	Accessing Values in a Dictionary	116
4.4.3	Operations on a Dictionary	116
4.4.3.1	Adding and Modifying an Element in a Dictionary	116
4.4.3.2	Removing an Element from a Dictionary	117
4.4.3.3	Membership Test in a Dictionary	118
4.4.4	Looping in a Dictionary	119
4.4.5	Nested Dictionaries	120
4.4.6	Dictionary Methods	120
4.5	Comparative Study	122
4.6	Programming Examples	122
	<i>Python Data Structures at a Glance</i>	137
	<i>Multiple Choice Questions</i>	138
	<i>Review Exercises</i>	141
	<i>Problems for Programming</i>	143

5. STRINGS	145
5.1 Introduction	145
5.2 Basic String Operations	147
5.2.1 Slicing Operations on String	148
5.2.2 Concatenation and Repeating Operation on Strings	149
5.3 Looping through a String	149
5.4 String Methods	150
5.5 String Comparison	157
5.6 Regular Expressions	158
5.7 Pattern Matching Algorithms	165
5.7.1 Brute Force Pattern Matching Algorithm	166
5.7.2 Knuth Morris Pratt Pattern Matching Algorithm	168
5.8 Programming Example	175
<i>Strings at a Glance</i>	181
<i>Multiple Choice Questions</i>	182
<i>Review Exercises</i>	185
<i>Problems for Programming</i>	186
6. RECURSION	187
6.1 Definition	187
6.2 Types of Recursion	189
6.3 Recursion vs. Iteration	192
6.4 Some Classical Problems on Recursion	192
6.4.1 Towers of Hanoi Problem	192
6.4.2 Eight Queen Problem	194
6.5 Advantages and Disadvantages of Recursion	196
6.6 Analysis of Recursive Algorithms	196
6.7 Programming Examples	199
<i>Recursion at a Glance</i>	201
<i>Multiple Choice Questions</i>	202
<i>Review Exercises</i>	205
<i>Problems for Programming</i>	206

7. LINKED LIST	207
7.1 Definition	208
7.2 Advantages of a Linked List	208
7.3 Types of Linked Lists	208
7.4 Implementing a Singly Linked List	210
7.5 Operations on Singly Linked List	211
7.5.1 Creating a Singly Linked List	212
7.5.2 Displaying a Singly Linked List	213
7.5.3 Inserting a New Element in a Singly Linked List	214
7.5.3.1 Inserting an Element at the Beginning of a List	215
7.5.3.2 Inserting an Element at the End of a List	216
7.5.3.3 Inserting a Node after a Specified Node	217
7.5.3.4 Inserting a Node before a Specified Node	219
7.5.4 Deleting a Node from a Singly Linked List	220
7.5.4.1 Deleting the First Node	220
7.5.4.2 Deleting the Last Node	221
7.5.4.3 Deleting any Intermediate Node	222
7.6 Applications of a Singly Linked List	228
7.7 Implementing a Circular Singly Linked List	233
7.8 Operations on a Circular Singly Linked List	233
7.8.1 Creating a Circular Singly Linked List	234
7.8.2 Displaying a Circular Singly Linked List	235
7.8.3 Inserting a New Element in a Circular Linked List	236
7.8.3.1 Inserting an Element at the Beginning of a Circular Linked List	237
7.8.3.2 Inserting an Element at the End of a List	239
7.8.3.3 Inserting a Node after a Specified Node	240
7.8.3.4 Inserting a Node before a Specified Node	242
7.8.4 Deleting a Node from a Circular Singly Linked List	243
7.8.4.1 Deleting the First Node	244
7.8.4.2 Deleting the Last Node	245
7.8.4.3 Deleting any Intermediate Node	246
7.9 Applications of a Circular Singly Linked List	253
7.10 Implementing a Doubly Linked List	255
7.11 Operations on a Doubly Linked List	256

7.11.1	Inserting an Element at the Beginning of a Doubly Linked List	257
7.11.2	Inserting an Element at the End of a Doubly Linked List	258
7.11.3	Inserting a Node at a Specified Position in a Doubly Linked List	259
7.11.4	Deleting a Node from a Doubly Linked List	261
7.11.4.1	Deleting the First Node	261
7.11.4.2	Deleting the Last Node	262
7.11.4.3	Deleting any Intermediate Node	263
7.12	Implementation of a Circular Doubly Linked List	270
7.13	Operations on a Circular Doubly Linked List	271
7.13.1	Inserting an Element at the Beginning of a Circular Doubly Linked List	271
7.13.2	Inserting an Element at the End of a Circular Doubly Linked List	273
7.13.3	Deleting the First Node from a Circular Doubly Linked List	274
7.13.4	Deleting the Last Node from a Circular Doubly Linked List	276
7.14	Header Linked List	283
7.15	Advantages of a Header Linked List	296
7.16	Disadvantages of a Linked List	296
7.17	Programming Examples	297
	<i>Linked List at a Glance</i>	303
	<i>Multiple Choice Questions</i>	304
	<i>Review Exercises</i>	308
	<i>Problems for Programming</i>	309
8.	STACK	311
8.1	Definitions and Concept	311
8.2	Operations Associated with Stacks	312
8.3	Representation of a Stack	312
8.3.1	Array Representation	312
8.3.2	Python List Representation	314
8.3.3	Linked Representation	317
8.4	Multiple Stacks	321
8.5	Applications of a Stack	321
8.5.1	Parenthesis Checking Problem	322
8.5.2	Conversion and Evaluation of Different Arithmetic Expressions	325
8.5.2.1	Different Notations of Arithmetic Expressions	325
8.5.2.2	Conversion of an Infix Expression into a Postfix Expression	327

8.5.2.3	Evaluation of a Postfix Expression	331
8.5.2.4	Conversion of a Postfix Expression into an Infix Expression	334
8.5.2.5	Conversion of an Infix Expression into a Prefix Expression	335
8.5.3	Reversing any Sequence	339
8.5.4	Recursion	341
	<i>Stack at a Glance</i>	343
	<i>Multiple Choice Questions</i>	344
	<i>Review Exercises</i>	348
	<i>Problems for Programming</i>	349
9.	QUEUE	351
9.1	Definitions and Concept	351
9.2	Operations Associated with Queues	352
9.3	Representation of a Queue	352
9.3.1	Array Representation of a Queue	353
9.3.2	Circular Queue	360
	9.3.2.1 Operations on a Circular Queue	361
9.3.3	Python List Representation of a Queue	369
9.3.4	Linked Representation of a Queue	371
	9.3.4.1 Using a Header List	371
	9.3.4.2 Using a Single Circular Linked List with a Single Tail Pointer	375
9.4	Multiple Queues	377
9.5	Special Queues	378
	9.5.1 DEQueue	378
	9.5.2 Priority Queue	381
9.6	Applications of a Queue	384
	<i>Queue at a Glance</i>	385
	<i>Multiple Choice Questions</i>	385
	<i>Review Exercises</i>	387
	<i>Problems for Programming</i>	388
10.	TREES	391
10.1	Definition and Concept	391
10.2	Terminology	392
10.3	Types of Trees	393

10.3.1	General Tree	393
10.3.2	Forest	393
10.3.3	Binary Tree	394
10.3.4	Strictly Binary Tree	394
10.3.5	Complete Binary Tree	395
10.3.6	Full Binary Tree	395
10.3.7	Extended Binary Tree	396
10.3.8	Binary Search Tree (BST)	396
10.3.9	Expression Tree	397
10.3.10	Tournament Tree	398
10.4	Representation of a Binary Tree	398
10.4.1	Array Representation of a Binary Tree	398
10.4.2	Linked list Representation of a Binary Tree	399
10.5	Binary Tree Traversal	400
10.5.1	Preorder Traversal of a Binary Tree	401
10.5.2	Inorder Traversal of a Binary Tree	403
10.5.3	Postorder Traversal of a Binary Tree	404
10.5.4	Level Order Traversal of a Binary Tree	406
10.6	Construction of a Binary Tree from the Traversal Path	406
10.7	Conversion of a General Tree to a Binary Tree	408
10.8	Binary Search Tree (BST)	410
10.9	Operations on a Binary Search Tree – Recursive and Non-recursive	412
10.9.1	Insertion of a New Node in a Binary Search Tree	412
10.9.2	Searching a Node in a Binary Search Tree	414
10.9.3	Traversing a Binary Search Tree	415
10.9.4	Deletion of a Node from a Binary Search Tree	415
10.9.5	Find the Largest Node from a Binary Search Tree	420
10.9.6	Finding the Smallest Node from a Binary Search Tree	421
10.9.7	Counting the Total Number of Nodes in a Binary Search Tree	422
10.9.8	Counting the Number of External Nodes in a Binary Search Tree	422
10.9.9	Counting the Number of Internal Nodes in a Binary Search Tree	423
10.9.10	Finding the Height of a Binary Search Tree	423
10.9.11	Finding the Mirror Image of a Binary Search Tree	424
10.10	Threaded Binary Tree	436
10.10.1	Representation of a Threaded Binary Tree	439

10.10.2 Operations on an Inorder Threaded Binary Tree	439
10.10.2.1 Inorder Traversal of an Inorder Threaded Binary Tree	440
10.10.2.2 Inserting a New Node in an Inorder Threaded Binary Search Tree	440
10.10.2.3 Deletion of a Node from an Inorder Threaded Binary Search Tree	443
10.11 AVL Tree	449
10.11.1 Operations on an AVL Tree	450
10.11.1.1 Inserting a Node in an AVL Tree	450
10.11.1.2 Deleting a Node from an AVL Tree	455
10.12 Red–Black Tree	456
10.12.1 Inserting a New Node in a Red–Black Tree	457
10.12.2 Deleting a Node from a Red–Black Tree	460
10.13 Huffman Coding	462
10.14 M-way Search Trees	465
10.15 B Tree	466
10.15.1 Inserting a New Element in a B Tree	467
10.15.2 Deleting Elements from a B Tree	468
10.15.3 Searching an Element from a B Tree	470
10.16 B+ Tree	471
10.16.1 Inserting a New Element in a B+ Tree	471
10.16.2 Deleting Elements from a B+ Tree	473
10.17 B* Tree	476
10.18 2–3 Tree	476
10.19 Trie Tree	477
<i>Strings at a Glance</i>	478
<i>Multiple Choice Questions</i>	478
<i>Review Exercises</i>	481
<i>Problems for Programming</i>	485
11. HEAP	487
11.1 Definition and Concept	487
11.2 Representation of a Heap in Memory	488
11.3 Operations on a Heap	488
11.3.1 Inserting a New Element in a Heap	488

11.3.2 Deleting an Element from a Heap	491
11.4 Applications of Heap	495
11.4.1 Implementing a Priority Queue Using Heap	495
<i>Heap at a Glance</i>	498
<i>Multiple Choice Questions</i>	499
<i>Review Exercises</i>	501
<i>Problems for Programming</i>	501
12. GRAPH	503
12.1 Definition and Concept	503
12.2 Terminology	504
12.3 Representation of a Graph	506
12.3.1 Adjacency Matrix Representation	506
12.3.2 Incidence Matrix Representation	507
12.3.3 Adjacency List Representation	508
12.3.4 Adjacency Multi-list Representation	509
12.4 Operations on a Graph	511
12.4.1 Insertion Operation	511
12.4.2 Deletion Operation	511
12.4.3 Graph Traversal	521
12.4.3.1 Breadth First Search Algorithm	521
12.4.3.2 Depth First Search Algorithm	525
12.5 Minimum Spanning Tree	528
12.5.1 Prim's Algorithm	529
12.5.2 Kruskal's Algorithm	532
12.6 Shortest Path Algorithm	537
12.6.1 Within a Given Source and Destination	537
12.6.2 Among All Pairs of Vertices	542
12.7 Applications of Graph	549
<i>Graph at a Glance</i>	549
<i>Multiple Choice Questions</i>	550
<i>Review Exercises</i>	553
<i>Problems for Programming</i>	554

13. SEARCHING AND SORTING	557
13.1 Introduction to Searching	557
13.1.1 Linear Search	558
13.1.2 Binary Search	559
13.1.3 Interpolation Search	564
13.2 Introduction to Sorting	566
13.2.1 Bubble Sort	567
13.2.2 Selection Sort	571
13.2.3 Insertion Sort	574
13.2.4 Quick Sort	577
13.2.5 Merge Sort	583
13.2.6 Heap Sort	587
13.2.7 Radix Sort	591
13.2.8 Shell Sort	594
13.3 Comparison of Different Sorting Algorithms	598
13.4 Concept of Internal and External Sorting	598
<i>Searching and Sorting at a Glance</i>	598
<i>Multiple Choice Questions</i>	600
<i>Review Exercises</i>	602
<i>Programming Exercises</i>	604
14. HASHING	605
14.1 Definitions and Concept	605
14.2 Hash Functions	607
14.2.1 Division Method	607
14.2.2 Multiplication Method	608
14.2.3 Mid-square Method	609
14.2.4 Folding Method	610
14.2.5 Length Dependent Method	610
14.2.6 Digit Analysis Method	611
14.3 Collision Resolution Technique	612
14.3.1 Open Addressing	612
14.3.1.1 Linear Probing	613
14.3.1.2 Quadratic Probing	615

14.3.1.3 Double Hashing	617
14.3.2 Chaining	620
14.4 Rehashing	623
14.5 Applications of Hashing	623
<i>Hashing at a Glance</i>	623
<i>Multiple Choice Questions</i>	624
<i>Review Exercises</i>	627
<i>Appendix: Answers of Multiple Choice Questions</i>	629
<i>Index</i>	633

Cambridge University Press & Assessment
978-1-009-27697-9 – Data Structures and Algorithms Using Python
Subrata Saha
Frontmatter
[More Information](#)

Preface

In computer science and engineering, data structure and algorithm are two very important parts. Data structure is the logical representation of data, so that insertion, deletion, and retrieval can be done efficiently, and an algorithm is the step-by-step procedure to solve any problem. By studying different data structures, we are able to know their merits and demerits, which enriches our knowledge and our ability to apply the appropriate data structures at proper places when we try to write new applications. Studying different standard algorithms provides us much knowledge about solving new problems. Both data structures and algorithms are interrelated and are complementary to each other. By studying both data structures and algorithms, we may acquire a solid foundation of writing good code. This comprehensive knowledge helps to understand new frameworks as well.

With the studying of data structures and algorithms, it is very important to implement them using proper languages. Several books have been written on this topic using the C language. But today Python has become very popular because of its features such as being easy, open source, object oriented, portable, multi-threaded, extensive libraries, embeddable, etc. Hence, in this book data structures and algorithms are implemented using Python.

About the Book

This book is written to serve the purposes of a textbook for undergraduate courses of computer science and information technology and for undergraduate and postgraduate courses of computer application where data structure is one of the subjects in the syllabus. In this book different data structures and algorithms are discussed in a lucid manner so that students can understand the concept easily. All the relevant data structures and their operations are discussed with diagrams and examples for better understanding. After discussing relevant algorithms in detail, the algorithmic representation and the corresponding code in Python are given. Various programming examples along with new

problems for practice and a set of MCQ questions at the end of each chapter increase the self-learning process of the students.

The salient features of this book are:

- This book is written in very simple English for better understanding the complex concepts.
- In order to make the presentation visually interactive for students, neat labelled diagrams are provided wherever necessary. For each topic, explanations are clear and concise, avoiding verbosity as much as possible.
- Each topic is discussed in detail with proper examples.
- Algorithms are presented with algorithmic representation as well as with the corresponding code in Python.
- Complexity analysis is discussed for almost all problems discussed in this book and in very lucid manner for better understanding.
- A large number of solved programming examples.
- MCQ questions and their solutions.

Acknowledgments

I thank my beloved students for their constant motivation which made me write this book. My students are extremely fond of and fascinated by my ‘teaching from the ground up’ style and their demand inspired me to write this book. My heartiest thanks to them.

I would like to thank my student Sayandip Naskar for drawing the figures of stack and queue.

My heartiest thanks to my family members and friends for their constant support, encouragement, and unconditional love, which helped to conclude the book.

Finally, I would like to thank all the reviewers of this book for their critical comments and suggestions. I convey my sincere gratitude to Mr Agnibesh Das and the entire editing team at Cambridge University Press, India, for their great work.

Cambridge University Press & Assessment
978-1-009-27697-9 – Data Structures and Algorithms Using Python
Subrata Saha
Frontmatter
[More Information](#)
