Preface

Data structures play a vital role in storing and organizing data within an application. It is important to choose the right data structure to significantly improve the application's performance, as it is highly desirable to be able to scale the application as the data quantity increases. This new edition teaches you essential Python data structures and the most common and important algorithms for building easy, maintainable applications. It also allows you to implement these algorithms with working examples and easy to follow step-by-step instructions.

In this book, you will learn the essential Python data structures and the most common algorithms. With this easy-to-read book, you will learn how to create complex data structures such as linked lists, stacks, heaps, queues, trees, and graphs as well as sorting algorithms including bubble sort, insertion sort, heapsort, and quicksort. We also describe various selection algorithms such as randomized and deterministic selection and provide a detailed discussion of various data structure algorithms and design paradigms such as greedy algorithms, divide-and-conquer, and dynamic programming. In addition, complex data structures such as trees and graphs are explained with easy pictorial examples to understand the concepts of these useful data structures. You will also learn various important string processing and pattern-matching algorithms such as KMP and Boyer-Moore algorithms along with their easy implementation in Python.

Who this book is for

This book is intended for Python developers who are studying data structures and algorithms at a beginner or intermediate level, as chapters provide practical examples and an easy approach to complex algorithms. It may also be useful for engineering students on a course in data structures and algorithms, as it covers almost all the algorithms, concepts, and designs that are studied. This book is also designed for software developers who want to deploy various applications using a specific data structures, as this book provides efficient ways to store relevant data.

It is assumed that the reader has some basic knowledge of the Python; however, it is not necessary, as we provide a quick overview of Python and object-oriented concepts.

What this book covers

Chapter 1, *Python Data Types and Structures*, introduces the basic data types and structures in Python. It will provide an overview of several built-in data structures available in Python that are pivotal for understanding the internals of data structures.

Chapter 2, *Introduction to Algorithm Design*, provides details about algorithm design issues and techniques. This chapter will compare different analyzing algorithms via running time and computation complexity, which will tell us which ones perform better than others for a given problem.

Chapter 3, *Algorithm Design Techniques and Strategies*, covers various important data structure design paradigms such as greedy algorithms, dynamic programming, divide-and-conquer. We will learn to create data structures via a number of primary principles, such as robustness, adaptability and reusability, and learn to separate structure from a function.

Chapter 4, *Linked Lists*, covers linked lists, which are one of the most common data structures and are often used to implement other structures, such as stacks and queues. In this chapter, we describe linked lists, their operation, and implementation. We compare their behavior to arrays and discuss the relative advantages and disadvantages of each.

Chapter 5, *Stacks and Queues*, describes stack and queue data structures in detail. It also discusses the behavior and demonstrates some implementations of these linear data structures. We give examples of typical real-life example applications.

Chapter 6, *Trees*, considers how trees form the basis of many of the most important advanced data structures. In this chapter we look at how to implement a binary tree. We will examine how to traverse trees and retrieve and insert values.

Chapter 7, *Heaps and Priority Queues*, looks into priority queues as important data structures and shows how to implement them using heap.

Chapter 8, *Hash Tables*, describes symbol tables, gives some typical implementations, and discusses various applications. We will look at the process of hashing, give an implementation of a hash table, and discuss the various design considerations.

Chapter 9, *Graphs and Algorithms*, looks at some of the more specialized structures, including graphs and spatial structures. We will learn to represent data through nodes and vertices and create structures such as directed and undirected graphs. We will also learn different algorithms for minimum spanning trees such as Prim's algorithm and Kruskal's algorithm.

Chapter 10, *Searching*, discusses the most common searching algorithms including, binary search and interpolation searching algorithms. We also give examples of their use for various data structures. Searching a data structure is a fundamental task and there are a number of approaches.

Chapter 11, *Sorting*, looks at the most common approaches to sorting. This will include bubble sort, insertion sort, selection sort, quick sort, and heap sort algorithms with detailed explanations, along with their Python implementations.

Chapter 12, *Selection Algorithms*, discusses how selection algorithms are commonly used to find the i^{th} smallest element from the list. It is an important operation related to sorting algorithms, and broadly related to the data structures and algorithms.

Chapter 13, *String Matching Algorithms*, covers basic concepts and definitions related to strings. In this chapter, various string and pattern matching algorithms are discussed in detail such as the naïve approach, and the **Knuth-Morris-Pratt** (**KMP**) and Boyer-Moore pattern matching algorithms.

Appendix, *Answers to the Questions*, provides answers to the exercises at the end of each chapter. Please feel free to check the appendix at the end of the book.

There is also bonus content available online related to tree algorithms at `https://static.packt-cdn.com/downloads/9781801073448_Bonus_Content.pdf`.

To get the most out of this book

The code in this book needs to be run on Python 3.10 or higher. Python's interactive environment can also be used to run the code snippets. It is advised to learn the algorithms and concepts by executing the code provided in the book to better understand the algorithms. The book is aimed to give practical exposure to the readers, so it is recommended to do the programming for all the algorithms to get maximum out of this book.

Download the example code files

The code bundle for the book is hosted on GitHub at `https://github.com/PacktPublishing/Hands-On-Data-Structures-and-Algorithms-with-Python-Third-Edition`. In case there's an update to the code, it will be updated on the existing GitHub repository.

We also have other code bundles from our rich catalog of books and videos available at `https://github.com/PacktPublishing/`. Check them out!

Download the color images

We also provide a PDF file that has color images of the screenshots/diagrams used in this book. You can download it here: `https://static.packt-cdn.com/downloads/9781801073448_ColorImages.pdf`.

Conventions used

There are a number of text conventions used throughout this book.

`CodeInText`: Indicates code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles. Here is an example: "The '`not in`' operator returns `True` if it does not find a variable in the specified sequence and `False` if it is found."

A block of code is set as follows:

```
p = "Hello India"
q = 10
r = 10.2
print(type(p))
print(type(q))
print(type(r))
```

When we wish to draw your attention to a particular part of a code block, the relevant lines or items are set in bold:

```
while self.slots[h] != None:
    if self.slots[h].key == key:
        return self.slots[h].value
    h = (h + j * (self.prime_num - (self.h2(key) % self.prime_num))) %
self.size
    j = j + 1
return None
```

Any command-line input or output is written as follows:

```
sudo apt-get install python3.10
```

Bold: Indicates a new term, an important word, or words that you see onscreen. For example, words in menus or dialog boxes appear in the text like this. Here is an example: "Each position in the hash table is often called a **slot** or **bucket** that can store an element."

Warnings or important notes appear like this.

Tips and tricks appear like this.

Get in touch

Feedback from our readers is always welcome.

General feedback: Email `feedback@packtpub.com` and mention the book title in the subject of your message. If you have questions about any aspect of this book, please email us at `questions@packtpub.com`.

Errata: Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you have found a mistake in this book, we would be grateful if you would report this to us. Please visit `www.packtpub.com/submit-errata`, selecting your book, clicking on the **Errata Submission Form link**, and entering the details.

Piracy: If you come across any illegal copies of our works in any form on the Internet, we would be grateful if you would provide us with the location address or website name. Please contact us at `copyright@packtpub.com` with a link to the material.

If you are interested in becoming an author: If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, please visit `authors.packtpub.com`.

Share Your Thoughts

Once you've read *Hands-On Data Structures and Algorithms with Python - Third Edition*, we'd love to hear your thoughts! Please `click here to go straight to the Amazon review page` for this book and share your feedback.

Your review is important to us and the tech community and will help us make sure we're delivering excellent quality content.