Data Structures

Andrew Rosen

Contents

1 Introduction **7**
- 1.1 What is a Data Structures Course . . . 7
- 1.2 Why This Book? . . . 7
  - 1.2.1 Where Does This Book Fit Into a Computer Science Curriculum . . . 7
  - 1.2.2 What Are My Base Assumptions about the Reader? . . . 7
- 1.3 To The Instructor . . . 8
- 1.4 To The Student . . . 8

2 The Array **9**
- 2.1 Array Operations . . . 9
- 2.2 Finding Values in an Array . . . 9

3 Analyzing Algorithms **11**
  - 3.0.1 Cost . . . 11
- 3.1 Big O Notation . . . 11
  - 3.1.1 Space Complexity . . . 11
- 3.2 The Formal Mathematics of Big O Notation . . . 11
- 3.3 Other Notations . . . 11

4 Lists **13**
- 4.1 What is a list? . . . 13
- 4.2 ArrayLists . . . 14
  - 4.2.1 Generics . . . 14
  - 4.2.2 Building an ArrayList . . . 14
  - 4.2.3 More Restrictive or Permissive Generics . . . 14
- 4.3 LinkedLists . . . 14
  - 4.3.1 Building a LinkedList . . . 14
- 4.4 Analysis . . . 14

5 Stacks **15**
- 5.1 Building a Stack . . . 15
- 5.2 Mazes - Stacks and Backtracking . . . 15
- 5.3 Discrete Finite Automata . . . 15

6 Queues **17**
- 6.1 Circular Arrays . . . 17

7 Recursion **19**
7.1 Recursive Mathematics . . . 19
7.2 Recursive Problem Solving . . . 19
7.2.1 Recursive Backtracking . . . 19
7.2.2 Recursive Combinations . . . 19
7.3 Recursion and Puzzles . . . 19
7.4 Recursion and Art . . . 19
7.5 Recursion and Nature . . . 19

8 Trees **21**
8.1 Binary Search Trees . . . 21
8.2 Heaps . . . 21
8.2.1 Priority Queues . . . 21
8.3 Trees and Heaps in Java . . . 21

9 Sorting **23**
9.1 Quadratic-Time Algorithms . . . 23
9.1.1 Bubble Sort . . . 23
9.1.2 Selection Sort . . . 23
9.1.3 Insertion Sort . . . 23
9.2 Log-Linear Sorting Algorithms . . . 23
9.2.1 Tree Sort . . . 23
9.2.2 Heap Sort . . . 23
9.2.3 Quick Sort . . . 23
9.2.4 Merge Sort . . . 23
9.3 Unique Sorting Algorithms . . . 23
9.3.1 Shell Sort . . . 23
9.3.2 Radix Sort . . . 23
9.4 State of the Art Sorting Algorithms . . . 23
9.4.1 Tim Sort . . . 23
9.4.2 Quick Sort . . . 23
9.4.3 Distributing and Parallelization . . . 23

10 Sets and Maps **25**
10.1 Sets . . . 25
10.2 Maps . . . 25
10.3 Hash Tables . . . 25
10.3.1 Creating a Hash Function . . . 25
10.4 Map Reduce . . . 25

11 Graphs **27**
11.1 Introduction and History . . . 28
11.2 Qualities of a Graph . . . 28
11.2.1 Undirected Edges . . . 28
11.2.2 Directed Edges . . . 28
11.2.3 Weighted Edges . . . 28
11.3 Directed Acyclic Graphs . . . 28
11.4 Building a Graph . . . 28
11.4.1 Adjacency List . . . 28
11.4.2 Adjacency Matrix . . . 28

11.5 Graph Algorithms 28
11.5.1 Searching and Traversing 28
11.5.2 Shortest Path 28
11.5.3 Topological Sorting 28
11.5.4 Minimum Spanning Trees 28
11.6 Graphs, Humans, and Networks 28
11.6.1 The Small World 28
11.6.2 Scale Free Graphs 28
11.7 Graphs in Art and Nature - Voronoi Tessellation 28
11.8 Distributed Hash Tables 28
11.9 A Nontechnical Introduction to NP-Completeness 28
11.9.1 The Traveling Salesperson Problem (TSP) 28
11.9.2 The Longest Path Problem 28
11.9.3 The Rudrata/Hamiltonian Path Problem 28

12 Other Data Structures **31**
12.1 Skip Lists 31

Chapter 1

Introduction

1.1 What is a Data Structures Course

Data Structures is all about defining the different ways we can organize data.

1.2 Why This Book?

1.2.1 Where Does This Book Fit Into a Computer Science Curriculum

Education in Computer Science is based around three core topics: translating the steps of solving a problem into a language a computer can understand, organizing data for solving problems, and techniques that can be used to solve problems. These courses typically covered in a university's introductory course, data structures course, and algorithms course respectively, although different universities decide exactly what content fits in which course. Of course, there is are lot more concepts in computer science, from operating systems and low level programming, to networks and how computers talk to each other. However, all these concepts rely on the knowledge gained in the core courses of programming, data structures, and algorithms.

This textbook is all about Data Structures, the middle section between learning how to program and the more advanced problem solving concepts we learn in Computer Science. Here, we focus on mastering the different ways to organize data, recognize the internal and performative differences between each structure, and learn to recognize the best (if there is one) for a given situation.

1.2.2 What Are My Base Assumptions about the Reader?

This textbook assumes that the student has taken a programming course that has covered the basics. Namely: data types such as ints, doubles, booleans, and strings; if statements, for and while loops; and object orient programming. The first writeup of the textbook will be done in Java, but I will try to add as much Python into the book as well.

1.3 To The Instructor

1.4 To The Student

Chapter 2

The Array

2.1 Array Operations

2.2 Finding Values in an Array

Chapter 3

Analyzing Algorithms

3.0.1 Cost

Time

Space

Energy

Other costs - Bandwidth

3.1 Big O Notation

3.1.1 Space Complexity

3.2 The Formal Mathematics of Big O Notation

3.3 Other Notations

Chapter 4

Lists

The first data structure we will be studying is the list. The list is by far the most relatable data structure, as humans deal with lists on a regular basis.

4.1 What is a list?

When you get right down to it, lists are defined by order.

```
public static <E> boolean isPermutation(List<E> listA, List<E> listB) {

        if(listA.size() != listB.size()) {
                return false;
        }
        for(int i  = 0; i < listA.size() ; i++){
                E item =  listA.get(i);
                int countA = 0;
                int countB = 0;

                for (E element : listA) {
                        if(item.equals(element)){
                                countA++;
                        }
                }
                for (E element : listB) {
                        if(item.equals(element)){
                                countB++;
                        }
                }
                if(countA != countB) {
                        return false;
                }
        }
        return true;
}
```

4.2 ArrayLists

4.2.1 Generics

4.2.2 Building an ArrayList

4.2.3 More Restrictive or Permissive Generics

4.3 LinkedLists

4.3.1 Building a LinkedList

4.4 Analysis

Chapter 5

Stacks

5.1 Building a Stack

5.2 Mazes - Stacks and Backtracking

5.3 Discrete Finite Automata

Chapter 6

Queues

A queue (pronounced by saying the first letter and ignoring all the others) is a data structure which emulates the real word functionality of standing in a line (or queue, for those from the Commonwealth),

6.1 Circular Arrays

Chapter 7

Recursion

7.1 Recursive Mathematics

7.2 Recursive Problem Solving

7.2.1 Recursive Backtracking

7.2.2 Recursive Combinations

7.3 Recursion and Puzzles

7.4 Recursion and Art

7.5 Recursion and Nature

Chapter 8

Trees

8.1 Binary Search Trees

8.2 Heaps

8.2.1 Priority Queues

8.3 Trees and Heaps in Java

Chapter 9

Sorting

9.1 Quadratic-Time Algorithms

9.1.1 Bubble Sort

9.1.2 Selection Sort

9.1.3 Insertion Sort

9.2 Log-Linear Sorting Algorithms

9.2.1 Tree Sort

9.2.2 Heap Sort

9.2.3 Quick Sort

9.2.4 Merge Sort

9.3 Unique Sorting Algorithms

9.3.1 Shell Sort

9.3.2 Radix Sort

9.4 State of the Art Sorting Algorithms

9.4.1 Tim Sort

9.4.2 Quick Sort

9.4.3 Distributing and Parallelization

Chapter 10

Sets and Maps

10.1 Sets

10.2 Maps

10.3 Hash Tables

10.3.1 Creating a Hash Function

10.4 Map Reduce

Chapter 11

Graphs

11.1 Introduction and History

11.2 Qualities of a Graph

11.2.1 Undirected Edges

11.2.2 Directed Edges

11.2.3 Weighted Edges

11.3 Directed Acyclic Graphs

11.4 Building a Graph

11.4.1 Adjacency List

11.4.2 Adjacency Matrix

11.5 Graph Algorithms

11.5.1 Searching and Traversing

Breadth First Search

Depth First Search

11.5.2 Shortest Path

11.5.3 Topological Sorting

11.5.4 Minimum Spanning Trees

11.6 Graphs, Humans, and Networks

11.6.1 The Small World

The Milgram Experiment

The Less-Known Milgram Experiment

11.6.2 Scale Free Graphs

11.7 Graphs in Art and Nature - Voronoi Tessellation

11.8 Distributed Hash Tables

Figure 11.1: The wings of a dragonfly. Credit: Joi Ito (CC BY 2.0)

Chapter 12

Other Data Structures

12.1 Skip Lists