

A Practical Introduction to **Data Structures**  
and **Algorithms** in JavaScript

# Algorithms

## **ABSOLUTE BEGINNER'S GUIDE**

No experience necessary!



Kirupa Chinnathambi

# Absolute Beginner's Guide to Algorithms

A Practical Introduction to  
Data Structures and Algorithms  
in JavaScript

**ABSOLUTE  
BEGINNER'S  
GUIDE**



Kirupa Chinnathambi



Pearson

# **Absolute Beginner's Guide to Algorithms: A Practical Introduction to Data Structures and Algorithms in JavaScript**

## **Table of Contents**

Cover

Title Page

Copyright Page

Contents at a Glance

Table of Contents

Part I: Data Structures

1 Introduction to Data Structures

Right Tool for the Right Job

Back to Data Structures

Conclusion

Some Additional Resources

2 Big-O Notation and Complexity Analysis

Its Example Time

Its Big-O Notation Time!

Conclusion

Some Additional Resources

3 Arrays

What Is an Array?

Adding an Item

Deleting an Item

# Table of Contents

Searching for an Item

Accessing an Item

Array Implementation / Use Cases

Arrays and Memory

Performance Considerations

Access

Insertion

Deletion

Searching

Conclusion

Some Additional Resources

## 4 Linked Lists

Meet the Linked List

Finding a Value

Adding Nodes

Deleting a Node

Linked List: Time and Space Complexity

Deeper Look at the Running Time

Space Complexity

Linked List Variations

Singly Linked List

Doubly Linked List

Circular Linked List

Skip List

Implementation

Conclusion

Some Additional Resources

## 5 Stacks

Meet the Stack

A JavaScript Implementation

Stacks: Time and Space Complexity

# Table of Contents

Runtime Performance

Memory Performance

Conclusion

Some Additional Resources

## 6 Queues

Meet the Queue

A JavaScript Implementation

Queues: Time and Space Complexity

Runtime Performance

Memory Performance

Conclusion

Some Additional Resources

## 7 Trees

Trees 101

Height and Depth

Conclusion

Some Additional Resources

## 8 Binary Trees

Meet the Binary Tree

Rules Explained

Binary Tree Variants

What about Adding, Removing, and Finding Nodes?

A Simple Binary Tree Implementation

Conclusion

Some Additional Resources

## 9 Binary Search Trees

Its Just a Data Structure

Adding Nodes

Removing Nodes

Implementing a Binary Search Tree

# Table of Contents

Performance and Memory Characteristics

Conclusion

Some Additional Resources

## 10 Heaps

Meet the Heap

Common Heap Operations

Heap Implementation

Heaps as Arrays

The Code

Performance Characteristics

Removing the Root Node

Inserting an Item

Performance Summary

Conclusion

Some Additional Resources

## 11 Hashtable (aka Hashmap or Dictionary)

A Very Efficient Robot

From Robots to Hashing Functions

From Hashing Functions to Hashtables

Adding Items to Our Hashtable

Reading Items from Our Hashtable

JavaScript Implementation/Usage

Dealing with Collisions

Performance and Memory

Conclusion

Some Additional Resources

## 12 Trie (aka Prefix Tree)

What Is a Trie?

Inserting Words

Finding Items

# Table of Contents

Deleting Items

Diving Deeper into Tries

Many More Examples Abound!

Implementation Time

Performance

Conclusion

Some Additional Resources

## 13 Graphs

What Is a Graph?

Graph Implementation

Representing Nodes

The Code

Conclusion

Some Additional Resources

## Part II: Algorithms

### 14 Introduction to Recursion

Our Giant Cookie Problem

Recursion in Programming

Recursive Function Call

Terminating Condition

Conclusion

Some Additional Resources

### 15 Fibonacci and Going Beyond Recursion

Recursively Solving the Fibonacci Sequence

Recursion with Memoization

Taking an Iteration-Based Approach

Going Deeper on the Speed

Conclusion

Some Additional Resources

### 16 Towers of Hanoi

# Table of Contents

How Towers of Hanoi Is Played

The Single Disk Case

Its Two Disk Time

Three Disks

The Algorithm

The Code Solution

Check Out the Recursiveness!

Its Math Time

Conclusion

Some Additional Resources

## 17 Search Algorithms and Linear Search

Linear Search

Linear Search at Work

JavaScript Implementation

Runtime Characteristics

Conclusion

Some Additional Resources

## 18 Faster Searching with Binary Search

Binary Search in Action

Sorted Items Only, Please

Dealing with the Middle Element

Dividing FTW!

The JavaScript Implementation

Iterative Approach

Recursive Approach

Example of the Code at Work

Runtime Performance

Conclusion

Some Additional Resources

## 19 Binary Tree Traversal



# Table of Contents

Breadth-First Traversal

Depth-First Traversal

Implementing Our Traversal Approaches

Node Exploration in the Breadth-First Approach

Node Exploration in the Depth-First Approach

Looking at the Code

Performance of Our Traversal Approaches

Conclusion

Some Additional Resources

## 20 Depth-First Search (DFS) and Breadth-First Search (BFS)

A Tale of Two Exploration Approaches

Depth-First Search Overview

Breadth-First Search Overview

Yes, They Are Different!

Its Example Time

Exploring with DFS

Exploring with BFS

When to Use DFS? When to Use BFS?

A JavaScript Implementation

Using the Code

Implementation Detail

Performance Details

Conclusion

Some Additional Resources

## 21 Quicksort

A Look at How Quicksort Works

A Simple Look

Another Simple Look

Its Implementation Time

Performance Characteristics

Time Complexity

# Table of Contents

Space Complexity

Stability

Conclusion

Some Additional Resources

## 22 Bubblesort

How Bubblesort Works

Walkthrough

The Code

Conclusion

Some Additional Resources

## 23 Insertion Sort

How Insertion Sort Works

One More Example

Algorithm Overview and Implementation

Performance Analysis

Conclusion

Some Additional Resources

## 24 Selection Sort

Selection Sort Walkthrough

Algorithm Deep Dive

The JavaScript Implementation

Conclusion

Some Additional Resources

## 25 Mergesort

How Mergesort Works

Mergesort: The Algorithm Details

Looking at the Code

Conclusion

Some Additional Resources

# **Table of Contents**

26 Conclusion

Index