Algorithms

# 背景

我们在开始这本书的学习和实践前，先要想清楚，我们究竟为什么要学。

之前，我们已经学习了一部分《Data structure and algorithm in C》，通过这本书的实践，也学习到了一些算法的知识。这次我们换成了《Algorithms》这本。

## 主要目标

1. 通过java快速实现各类算法。

java语言我们更加熟悉，java抽象出来的集合类更加丰富，java语言整体的生态更加丰富。通过java语言，把各种算法的基本思路都实践一遍，付出的代价应该会比C语言更加小。

1. 为未来的使用场景做铺垫。

我们应当承认，在日常工作中，用到算法的场景非常非常少，几乎就是没有。但是，没有使用场景并不意味着没有用。现在没有碰到场景并不意味着以后也不会碰到。

1. 为未来更加高级的目标作铺垫。比如我们后续计划要学习的数据库原理实践(mini sql server)，如果不了解红黑树，那是不可想象的。另外，目前非常火热的AI，对于
2. 学习java版本的算法，对于理解java语言本身是非常有帮助的，至少对于学习java集合类是非常有帮助的。我们在学习了各种算法之后，可以看看java集合类的实现方式。尤其是像HashMap这种非常热门的集合类。
3. 我们知道，互联网大厂的面试，那是非常严格的。尤其是对于社招，最看重的当然是实践能力，而不是学历。在大厂的面试中，算法是最为重要，也是最为基本的一环。如果没有深入了解算法、没有非常熟练的实践经验，根本没法通过面试。
4. 无论是科技部经理还是架构部经理，都是技术型的经理，对于算法这类基础技术栈，那是十分看重的。后续要在科技部立足，精通算法那是必须的。

7.为未来刷题(PAT/LeetCode)做好铺垫。

## Bad part(阻力)

虽然是java版的算法书，但是我们也不能掉以轻心。因为对于算法来说，采用哪种语言实现并不重要。重要的是对算法原理的理解、对于复杂度的计算。因此，虽然我们对java比价了解，但是我们也要做好打持久战的准备。算法并不是一门轻松的技术，需要花费大量的时间精力，我们要提前有心理准备。

java语言实现，性能是一个很大的问题，因此，我们即便是能够通过java熟练实现算法，那也只是实现了算法而已。要落地到具体的应用，要进行真正的算法竞赛，那和C语言这类更加贴近硬件的语言相比，还是有很大差距的。

因为算法最大的用武之地，是在高性能场景，比如数据库、浏览器、搜索引擎这些。这些场景都不可能由java来实现。这一点，我们也要做好充分。

我们要非常清楚，通过java实现算法，仅仅只是学习算法的一种途径，并不是实现算法应用的目标。我们用java实现了各种算法，也不要得意。还是要对C、C++这类保持足够的敬意。

## 阅读步骤

和其他英文教材一样，这本书的特点也是非常非常的详细，并且针对部分算法，有形象的图示，这个是非常有用的。但是我们一上来就细读、通读书本是不太合适的，因为内容实在是太多了。

1. 把书本中所有的代码扫一遍，实践一遍

书本中正式的代码文件有46个，就算一天一个，也要46天，预计一个半月。当然，一天实践一个代码，那几乎是不可能的。

1. 用实践的代码文件，去做一下各种类型的算法题

可以找PAT、leetcode等网站刷题。可以有针对性的找一些特定类型的题，覆盖各个算法主题。最好也能够覆盖各种难度。

通过做题，我们应该能够领悟到一个道理，就是重视理论，重视方法。正所谓思而不学则惘，我们如果没有提前掌握各种算法、数据结构知识，就去刷算法题，然后自己琢磨解题思路，这是不可想象的。在其他领域、其他场景也是一样，要能够实现某个现实生活中的目标，把某个目标落地，光靠自己研究、琢磨那是不行的，要有必要的理论储备，通过理论指导行动。我们在现实生活中的绝大多数实践活动，前人都已经做过了，那么充分借鉴前人的成败得失，通过前人总结的经验去做事，那就显得非常重要。不要以为自己处于当代，就绝地自己是整个历史中最牛逼的。这种想法是不对的。

1. 333
2. 444
3. 555

# 1 Fundamentals

第一章主要是介绍算法的一些基础：ADT、时间复杂度等

## 1.1Basic Programming Model

|  |
| --- |
| BinarySear |

## 1.2 Data Abstraction

## 1.3 Bags ,Queues and Stacks

|  |
| --- |
| Evaluate.java  ResizingArrayStack.java |

## 1.4 Analysis of Algorithms

## 1.5 Case study :Union-Find

# 2 Sorting

## 2.1 Elementary Sorts

## 2.2 Mergsort

## 2.3 Quicksort

## 2.4 Priority Queues

## 2.5 Applications

# 3 Searching

## 3.1 Symbol tables

## 3.2 Binary Search Trees

## 3.3 Balanced Search Trees

## 3.4 Hash Tables

## 3.5 Applications

# 4 Graphs

## 4.1 Undirected Graphs

## 4.2 Directed Graphs

## 4.3 Minimum Spanning Trees

## 4.4 Shortest Paths

# 5 Strings

## 5.1 String Sorts

## 5.2 Tries

## 5.3 Substring Search

## 5.4 Regular Expressions

## 5.5 Data Compression

# 6.Context