练习答案

第1章

1.1 7步。

1.2 8步。

1.3 $O(\log n)$。

1.4 $O(n)$。

1.5 $O(n)$。

1.6 $O(n)$。你可能认为，我只对26个字母中的一个这样做，因此运行时间应为$O(n / 26)$。需要牢记的一条简单规则是，大O表示法不考虑乘以、除以、加上或减去的数字。下面这些都不是正确的大O运行时间：$O(n + 26)$、$O(n - 26)$、$O(n * 26)$、$O(n / 26)$，它们都应表示为$O(n)$！为什么呢？如果你好奇，请翻到4.3节，并研究大O表示法中的常量（常量就是一个数字，这里的26就是常量）。

第2章

2.1 在这里，你每天都在列表中添加支出项，但每月只读取支出一次。数组的读取速度快，而插入速度慢；链表的读取速度慢，而插入速度快。由于你执行的插入操作比读取操作多，因此使用链表更合适。另外，仅当你要随机访问元素时，链表的读取速度才慢。鉴于你要读取所有的元素，在这种情况下，链表的读取速度也不慢。因此，对这个问题来说，使用链表是不错的解决方案。

2.2 使用链表。经常要执行插入操作（服务员添加点菜单），而这正是链表擅长的。不需要执行（数组擅长的）查找和随机访问操作，因为厨师总是从队列中取出第一个点菜单。

2.3 有序数组。数组让你能够随机访问——立即获取数组中间的元素，而使用链表无法这样做。要获取链表中间的元素，你必须从第一个元素开始，沿链接逐渐找到这个元素。

2.4 数组的插入速度很慢。另外，要使用二分查找算法来查找用户名，数组必须是有序的。假设有一个名为Adit B的用户在Facebook注册，其用户名将插入到数组末尾，因此每次插入用户名后，你都必须对数组进行排序！

2.5 查找时，其速度比数组慢，但比链表快；而插入时，其速度比数组快，但与链表相当。因此，其查找速度比数组慢，但在各方面都不比链表慢。本书后面将介绍另一种混合数据结构——散列表。这个练习应该能让你对如何使用简单数据结构创建复杂的数据结构有大致了解。

Facebook实际使用的是什么呢？很可能是十多个数据库，它们基于众多不同的数据结构：散列表、B树等。数组和链表是这些更复杂的数据结构的基石。

第3章

3.1 下面是一些你可获得的信息。

- 首先调用了函数greet，并将参数name的值指定为maggie。
- 接下来，函数greet调用了函数greet2，并将参数name的值指定为maggie。
- 此时函数greet处于未完成（挂起）状态。
- 当前的函数调用为函数greet2。
- 这个函数执行完毕后，函数greet将接着执行。

3.2 栈将不断地增大。每个程序可使用的调用栈空间都有限，程序用完这些空间（终将如此）后，将因栈溢出而终止。

第4章

4.1
```
def sum(list):
  if list == []:
    return 0
  return list[0] + sum(list[1:])
```

4.2
```
def count(list):
  if list == []:
    return 0
  return 1 + count(list[1:])
```

4.3
```
def max(list):
  if len(list) == 2:
    return list[0] if list[0] > list[1] else list[1]
  sub_max = max(list[1:])
  return list[0] if list[0] > sub_max else sub_max
```

4.4 二分查找的基线条件是数组只包含一个元素。如果要查找的值与这个元素相同，就找到了！否则，就说明它不在数组中。

在二分查找的递归条件中，你把数组分成两半，将其中一半丢弃，并对另一半执行二分查找。

4.5 $O(n)$。

4.6 $O(n)$。

4.7 $O(1)$。

4.8 $O(n^2)$。

第5章

5.1 一致。

5.2 不一致。

5.3 不一致。

5.4 一致。

5.5 散列函数C和D可实现均匀分布。

5.6 散列函数B和D可实现均匀分布。

5.7 散列函数B、C和D可实现均匀分布。

第6章

6.1 最短路径的长度为2。

6.2 最短路径的长度为2。

6.3 A不可行，B可行，C不可行。

6.4 1——起床，2——锻炼，3——洗澡，4——刷牙，5——穿衣服，6——打包午餐，7——吃早餐。

6.5 A是树，B不是树，C是树。C是一棵横着的树。树是图的子集，因此树都是图，但图可能是树，也可能不是。

第7章

7.1 A为8；B为60；C使用狄克斯特拉算法无法找出最短路径，因为存在负权边。

第8章

8.1 一种贪婪策略是，选择可装入卡车剩余空间内的最大箱子，并重复这个过程，直到不

能再装入箱子为止。使用这种算法不能得到最优解。

8.2　不断地挑选可在余下的时间内完成的价值最大的活动，直到余下的时间不够完成任何活动为止。使用这种算法不能得到最优解。

8.3　不是。

8.4　是。

8.5　是。

8.6　是。

8.7　是。

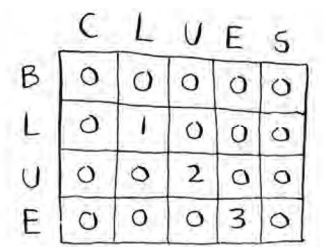
8.8　是。

第9章

9.1　要。在这种情况下，你可偷来MP3播放器和iPhone和吉他，总价值为4500美元。

9.2　你应携带水、食物和相机。

9.3



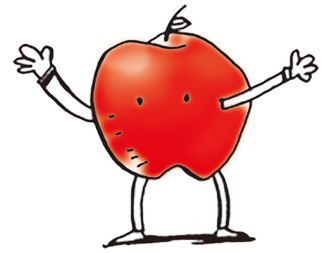
第10章

10.1　可使用*归一化*（normalization）。你可计算每位用户的平均评分，并据此来调整用户的评分。例如，你可能发现Pinky的平均评分为星3，而Yogi的平均评分为3.5星。因此，你稍微调高Pinky的评分，使其平均评分也为3.5星。这样就能基于同样的标准比较他们的评分了。

10.2　可在使用KNN时给意见领袖的评分更大权重。假设有3个邻居——Joe、Dave和意见领袖Wes Anderson，他们给Caddyshack的评分分别为3星、4星和5星。可不计算这些评分的平均值 (3 + 4 + 5) / 3 = 4星，而给Wes Anderson的评分更大权重：(3 + 4 + 5 + 5 + 5) / 5 = 4.4星。

10.3　太少了。如果考虑的邻居太少，结果很可能存在偏差。一个不错的经验规则是：如果有*N*位用户，应考虑sqrt(*N*)个邻居。

算法图解



Grokking Algorithms

An illustrated guide for programmers and other curious people

- 你一定能看懂的算法基础书
- 代码示例基于Python
- 400多个示意图，生动介绍算法执行过程
- 展示不同算法在性能方面的优缺点
- 教会你用常见算法解决每天面临的实际编程问题

本书完成了一项不可能完成的任务：让算法变得有趣、易懂！

—— **Sander Rossel，COAS Software Systems**

你渴望像看喜欢的小说一样学习算法吗？如果是，本书正是你梦寐以求的！

—— **Sankar Ramanathan，IBM Analytics**

如今，使用算法进行优化已渗透到了生活的方方面面。如果你正寻找优秀的算法入门书，本书就是你的首选。

—— **Amit Lamba，Tech Overture**

看了这本书我才知道，原来学习算法一点都不乏味！

—— **Christopher Haupt，Mobirobo**

MANNING

图灵社区：iTuring.cn

微　　博：@图灵教育　@图灵社区

分类建议　计算机 / 编程与开发 / 算法

人民邮电出版社网址：www.ptpress.com.cn

ISBN 978-7-115-44763-0

ISBN 978-7-115-44763-0

定价：49.00元