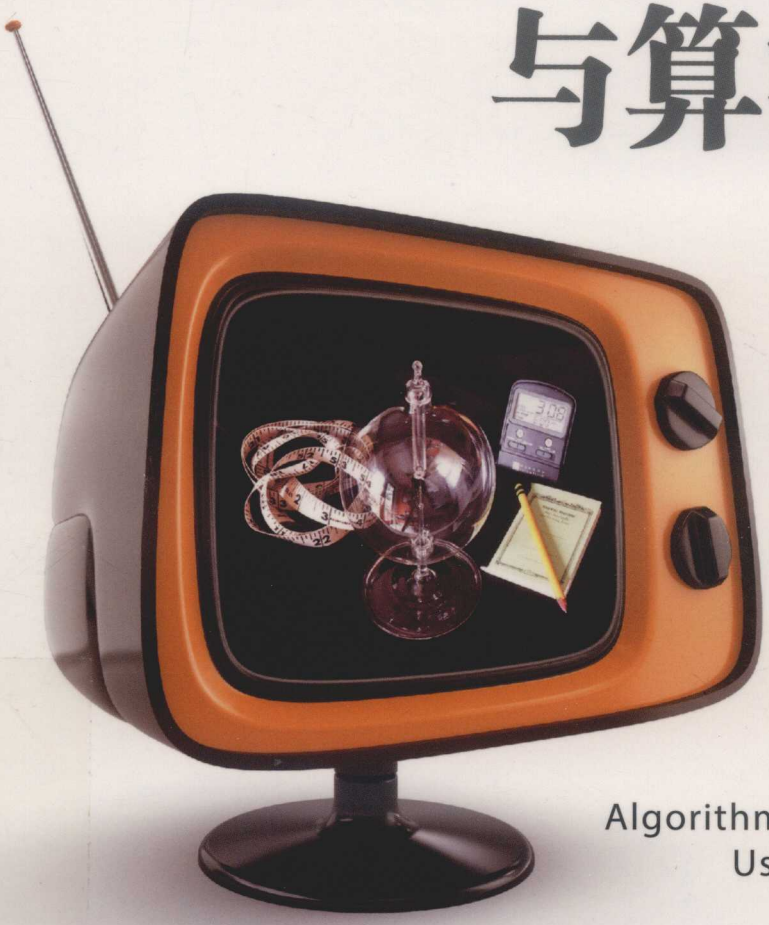
TURING 图灵程序设计丛书

Python数据结构与算法分析（第2版）



[美] 布拉德利·米勒 戴维·拉努姆 著
吕能 刁寿钧 译

Problem Solving with Algorithms and Data Structures Using Python Second Edition

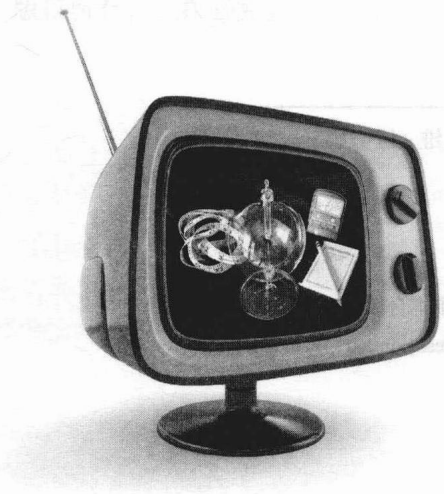
- 只有洞彻数据结构与算法，才能真正精通Python
- 经典计算机科学教材，华盛顿大学等多家高校采用

中国工信出版集团
人民邮电出版社 POSTS & TELECOM PRESS

TURING 图灵程序设计丛书

Python数据结构与算法分析（第2版）

[美] 布拉德利·米勒 著
戴维·拉努姆
周能 刁寿钧 译



Problem Solving with Algorithms and Data Structures Using Python Second Edition

人民邮电出版社
北 京

图书在版编目（CIP）数据

Python数据结构与算法分析 : 第2版 / (美) 布拉德利·米勒 (Bradley N. Miller), (美) 戴维·拉努姆 (David L. Ranum) 著 ; 吕能, 刁寿钧译. -- 北京 : 人民邮电出版社, 2019.9
（图灵程序设计丛书）
ISBN 978-7-115-51721-0

Ⅰ. ①P… Ⅱ. ①布… ②戴… ③吕… ④刁… Ⅲ. ①软件工具－程序设计 Ⅳ. ①TP311.561

中国版本图书馆CIP数据核字(2019)第155652号

内 容 提 要

了解数据结构与算法是透彻理解计算机科学的前提。随着 Python 日益广泛的应用，Python 程序员需要实现与传统的面向对象编程语言相似的数据结构与算法。本书是用 Python 描述数据结构与算法的开山之作，汇聚了作者多年的实战经验，向读者透彻讲解在 Python 环境下，如何通过一系列存储机制高效地实现各类算法。通过本书，读者将深刻理解 Python 数据结构、递归、搜索、排序、树与图的应用，等等。

本书适合所有 Python 程序员阅读。

◆ 著　[美] 布拉德利·米勒 戴维·拉努姆
译　吕 能 刁寿钧
责任编辑　谢婷婷
责任印制　周昇亮

◆ 人民邮电出版社出版发行　北京市丰台区成寿寺路11号
邮编　100164　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京鑫正大印刷有限公司印刷

◆ 开本：800×1000　1/16
印张：19.25
字数：466千字　2019年9月第1版
印数：1－3 000册　2019年9月北京第1次印刷
著作权合同登记号　图字：01-2018-0949 号

定价：79.00元

读者服务热线：(010)51095183转600　印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京东工商广登字 20170147 号

作者简介

布拉德利 · 米勒
(Bradley N. Miller)

美国路德学院计算机科学名誉教授，曾获美国计算机协会软件系统奖，对Python课程开发有深入研究，由他创立的互动式教科书平台Runestone Interactive与全球600多家教育机构有合作。

戴维 · 拉努姆
(David L. Ranum)

IBM Watson认知软件工程师，医学信息学博士，致力于利用自然语言处理等人工智能技术解决医疗问题，曾在美国路德学院讲授计算机科学课程近三十载。

译者简介

吕能

Twitter软件工程师，开源项目Apache Heron的核心贡献者。先后在浙江大学和美国加州大学洛杉矶分校取得计算机科学学士学位和硕士学位，关注分布式实时数据引擎系统的研发，热衷于普及计算机技术知识。

刁寿钧

腾讯优图实验室后台开发工程师，毕业于复旦大学。先后从事过广告业务与智慧零售、智慧社区业务的开发工作。关注算法与数据库技术，曾协助组织IMG社区的技术沙龙活动。另译有《数据分析实战》。

微信连接

回复“Python”查看相关书单

微博连接

关注@图灵教育 每日分享IT好书

QQ连接

图灵读者官方群I：218139230
图灵读者官方群II：164939616

图灵社区
iTuring.cn

在线出版，电子书，《码农》杂志，图灵访谈

站在巨人的肩上

Standing on Shoulders of Giants

iTuring.cn

版权声明

前　言

致学生

既然你已经开始阅读本书，那么必定对计算机科学感兴趣。你可能也对 Python 这门编程语言感兴趣，并且已经通过之前的课程或自学有了一些编程经验。不论是何种情况，你肯定都希望学习更多知识。

本书将介绍计算机科学，也会介绍 Python。然而，书中的内容并不仅仅局限于这两个方面。学习数据结构与算法对理解计算机科学的本质非常关键。

学习计算机科学与掌握其他高难度学科没什么不同。成功的唯一途径便是循序渐进地学习其中的核心思想。刚开始接触计算机科学的人，需要多多练习来加深理解，从而为学习更复杂的内容做好准备。此外，初学者需要建立起自信心。

本书用作数据结构与算法的入门课的教材。数据结构与算法通常是计算机科学专业的第二门课程，比第一门课程更深入。但是本书的读者对象仍然是初学者，很可能还在第一门课程所教的基本思想和方法中挣扎，但已经准备好进一步探索这一领域并且进一步练习如何解决问题。

如前所述，本书将介绍计算机科学。它既会介绍抽象数据类型及数据结构，也会介绍如何编写算法和解决问题。你会学习一系列的数据结构，并且解决各种经典问题。随着学习的深入，你将反复应用在各章中掌握的工具与技术。

致教师

许多学生在学到这个阶段时会发现，计算机科学除了编程以外还有很多内容。数据结构与算法可以独立于编程来学习和理解。

本书假定学生已经学过计算机科学的入门课程，但入门课程不一定是用 Python 讲解的。他们理解基本的结构体，比如分支、迭代以及函数定义，也接触过面向对象编程，并且能够创建和使用简单的类。同时，学生也能够理解 Python 的基础数据结构，比如序列（列表和字符串）以及字典。

本书有三大特点：

- 通过简单易读的文字而不引入太多编程语法，重点关注如何解决问题，从而向学生介绍基本的数据结构与算法；
- 较早介绍基于大 O 记法的算法分析，并且通篇运用；
- 使用 Python 讲解，以促使初学者能够使用和掌握数据结构与算法。

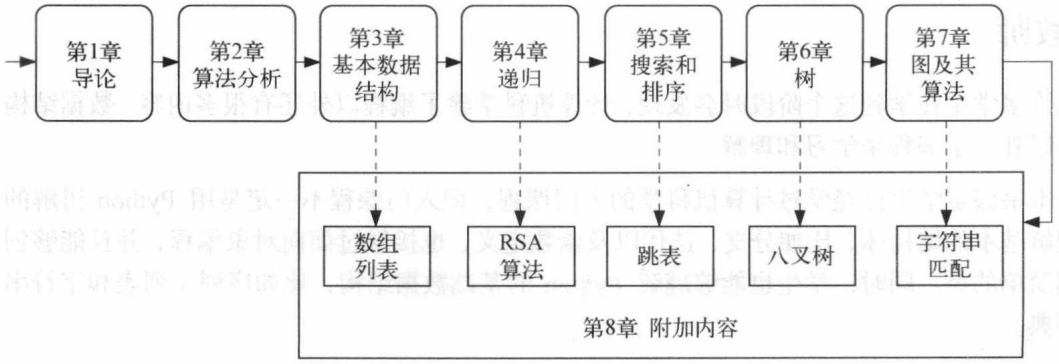
学生将首先学习线性数据结构，包括栈、队列、双端队列以及列表。我们用 Python 列表以及链表实现这些数据结构。然后学习与树有关的非线性数据结构，了解连接节点和引用结构（链表）等一系列技术。最后，学生将通过运用链式结构、链表以及 Python 字典的实现，学习图的相关知识。对于每一种结构，本书都尽力在使用 Python 提供的内建数据类型的同时展现众多的实现技巧。这种讲法在向学生揭示各种主要实现方法的同时，也强调 Python 的易用性。

Python 是一门非常适合于讲解算法的语言，语法干净简洁，用户环境直观，基本的数据类型十分强大和易用。其交互性在不需要额外编写驱动函数的情况下为测试数据结构单元提供了直观环境。而且，Python 为算法提供了教科书式的表示法，基本上不需要再用伪代码。这一特性有助于通过数据结构与算法来描述众多与之有关、相当有趣的现代问题。

我们相信，对于初学者来说，投入时间学习与算法和数据结构相关的基本思想是非常有益的。我们也相信，Python 是一门教授初学者入门的优秀语言。其他许多语言要求学生学习非常高级的编程概念，这会阻碍他们掌握真正需要的基础知识，从而可能导致失败，而这样的失败并不是计算机科学本身造成的，而是由于所使用的语言不当。我们的目标是提供一本教科书，量体裁衣般地聚焦于他们需要掌握的内容，以他们能理解的方式编写，创造和发展一个有助于他们成功的环境。

本书结构

本书紧紧地围绕着运用经典数据结构和技术来解决问题。下面的组织结构图展示了充分利用本书的不同方式。



第 1 章做一些背景知识的准备，我们来复习一下计算机科学、问题解决、面向对象编程以及 Python。基础扎实的学生可以跳过第 1 章，直接学习第 2 章。不过，正所谓温故而知新，适当的复习和回顾必然是值得的。

第 2 章介绍算法分析的内在思想，重点讲解大 *O* 记法，还将分析本书一直使用的重要 Python 数据结构。这可以帮助学生理解如何权衡各种抽象数据类型的不同实现。第 2 章也包含了在运行时使用的 Python 原生类型的实验测量例子。

第 3~7 章全面介绍在经典计算机科学问题中出现的数据结构与算法。尽管在阅读顺序上并无严格要求，但是许多话题之间都存在一定的依赖关系，所以应该按照本书的顺序学习。比如，第 3 章介绍栈，第 4 章利用栈解释递归，第 5 章利用递归实现二分搜索。

第 8 章是选学内容，包含彼此独立的几节。每一节都与之前的某一章有关。正如前面的组织结构图所示，你既可以在学习完第 7 章以后再一起学习第 8 章中的各节内容，也可以把它们与对应的那一章放在一起学习。例如，希望更早介绍数组的教师，可以在讲完第 3 章以后直接跳到 8.2 节。

新版改进

- 所有的代码都用 Python 3.2 写成。
- 第 1 章介绍 Python 的集合以及异常处理。
- 不再使用第三方绘图包。新版中的所有图形都通过原生的 `turtle` 模块绘制。
- 新加的第 2 章着重讲解算法分析。此外，这一章还包括对全书出现的关键 Python 数据结构的分析。
- 第 3 章新增了关于链表实现的一节。
- 将“动态规划”一节移到了第 4 章的最后。
- 更关注图递归算法，包括递归树绘制以及递归迷宫搜索程序。
- 所有的数据结构源代码都被放在一个 Python 包中，方便学生在完成作业时使用。
- 新版包含各章例子的完整源代码，从而避免了从各处复制代码的麻烦。
- 第 6 章改进了二叉搜索树。
- 第 6 章新增了关于平衡二叉树的一节。
- 第 8 章介绍 C 风格的数组和数组管理。

致谢

在完成本书的过程中，我们得到了众多朋友的帮助。感谢我们的同事 Steve Hubbard 为本书的第 1 版提供了大量的反馈，并为新版提供了众多新素材。感谢各地的同事给我们发邮件指出第

1 版存在的错误，并且为新版内容提供意见。

感谢迪科拉市 Java John's 咖啡馆的朋友 Mary 和 Bob，以及其他服务员，他们允许我俩在 Brad 休假期间成为店里的“常驻作者”。David 在常驻咖啡馆写书的那几个月中竟然没有成为咖啡爱好者。对了，在一间店名带 Java 的咖啡馆里写一本 Python 的书，确实有点讽刺。

感谢 Franklin, Beedle & Associates 出版公司的各位员工，特别是 Jim Leisy 和 Tom Sumner。与他们的合作非常愉快。最后，还要特别感谢我们两人的妻子 Jane Miller 和 Brenda Ranum。她们的爱与支持使得本书终成现实。

电子书

扫描如下二维码，即可购买本书电子版。

布拉德利·米勒（Bradley N. Miller）
戴维·拉努姆（David L. Ranum）

目　录

第1章 导 论

1.1 本章目标

- 复习计算机科学、编程以及解决问题方面的知识。
- 理解抽象这一概念及其在解决问题的过程中所发挥的作用。
- 理解并建立抽象数据类型的概念。
- 复习 Python。

1.2 入门

自从第一台利用转接线和开关来传递计算指令的电子计算机诞生以来，人们对编程的认识历经了多次变化。与社会生活的其他许多方面一样，计算机技术的变革为计算机科学家提供了越来越多的工具和平台去施展他们的才能。高效的处理器、高速网络以及大容量内存等一系列新技术，要求计算机科学家掌握更多复杂的知识。然而，在这一系列快速的变革之中，仍有一些基本原则始终保持不变。计算机科学被认为是一门利用计算机来解决问题的学科。

你肯定在学习解决问题的基本方法上投入过大量的时间，并且相信自己拥有根据问题描述构建解决方案的能力。你肯定也体会到了编写计算机程序的困难之处。大型难题及其解决方案的复杂性往往会掩盖问题解决过程的核心思想。

本章将为后续各章重点解释两个重要的话题。首先，本章会复习计算机科学以及数据结构与算法的研究必须符合的框架，尤其是学习这些内容的原因以及为什么说理解它们有助于更好地解决问题。其次，本章会复习 Python。尽管不会提供完整、详尽的 Python 参考资料，但是会针对阅读后续各章所需的基础知识及基本思想，给出示例以及相应的解释。

1.3 何谓计算机科学

要定义计算机科学，通常十分困难，这也许是因为其中的“计算机”一词。你可能已经意识到，计算机科学并不仅是研究计算机本身。尽管计算机在这一学科中是非常重要的工具，但也仅

仅只是工具而已。

计算机科学的研究对象是问题、解决问题的过程，以及通过该过程得到的解决方案。给定一个问题，计算机科学家的目标是开发一个能够逐步解决该问题的**算法**。算法是具有有限步骤的过程，依照这个过程便能解决问题。因此，算法就是解决方案。

可以认为计算机科学就是研究算法的学科。但是必须注意，某些问题并没有解决方案。尽管这一话题已经超出了本书讨论的范畴，但是对于学习计算机科学的人来说，认清这一事实非常重要。结合上述两类问题，可以将计算机科学更完善地定义为：研究问题及其解决方案，以及研究目前无解的问题的学科。

在描述问题及其解决方案时，经常用到“**可计算**”一词。若存在能够解决某个问题的算法，那么该问题便是可计算的。因此，计算机科学也可以被定义为：研究可计算以及不可计算的问题，即研究算法的存在性以及不存在性。在上述任意一种定义中，“计算机”一词都没有出现。解决方案本身是独立于计算机的。

在研究问题解决过程的同时，计算机科学也研究**抽象**。抽象思维使得我们能分别从逻辑视角和物理视角来看待问题及其解决方案。举一个常见的例子。

试想你每天开车去上学或上班。作为车的使用者，你在驾驶时会与它有一系列的交互：坐进车里，插入钥匙，启动发动机，换挡，刹车，加速以及操作方向盘。从抽象的角度来看，这是从逻辑视角来看待这辆车，你在使用由汽车设计者提供的功能来将自己从某个地方运送到另一个地方。这些功能有时候也被称作**接口**。

另一方面，修车工看待车辆的角度与司机截然不同。他不仅需要知道如何驾驶，而且更需要知道实现汽车功能的所有细节：发动机如何工作，变速器如何换挡，如何控制温度，等等。这就是所谓的物理视角，即看到表面之下的实现细节。

使用计算机也是如此。大多数人用计算机来写文档、收发邮件、浏览网页、听音乐、存储图像以及打游戏，但他们并不需要了解这些功能的实现细节。大家都是从逻辑视角或者使用者的角度来看待计算机。计算机科学家、程序员、技术支持人员以及系统管理员则从另一个角度来看待计算机。他们必须知道操作系统的原理、网络协议的配置，以及如何编写各种脚本来控制计算机。他们必须能够控制用户不需要了解的底层细节。

上面两个例子的共同点在于，抽象的用户（或称客户）只需要知道接口是如何工作的，而并不需要知道实现细节。这些接口是用户用于与底层复杂的实现进行交互的方式。下面是抽象的另一个例子，来看看 Python 的 `math` 模块。一旦导入这一模块，便可以进行如下的计算。

```
>>> import math
>>> math.sqrt(16)
4.0
>>>
```

这是一个**过程抽象**的例子。我们并不需要知道平方根究竟是如何计算出来的，而只需要知道

计算平方根的函数名是什么以及如何使用它。只要正确地导入模块，便可以认为这个函数会返回正确的结果。由于其他人已经实现了平方根问题的解决方案，因此我们只需要知道如何使用该函数即可。这有时候也被称为过程的“黑盒”视角。我们仅需要描述接口：函数名、所需参数，以及返回值。所有的计算细节都被隐藏了起来，如图 1-1 所示。

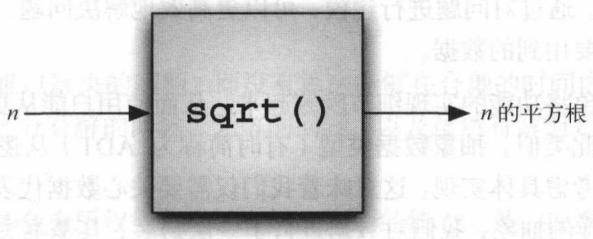


图 1-1 过程抽象

1.3.1 何谓编程

编程是指通过编程语言将算法编码以使其能被计算机执行的过程。尽管有众多的编程语言和不同类型的计算机，但是首先得有一个解决问题的算法。如果没有算法，就不会有程序。

计算机科学的研究对象并不是编程。但是，编程是计算机科学家所做工作的一个重要组成部分。通常，编程就是为解决方案创造表达方式。因此，编程语言对算法的表达以及创造程序的过程是这一学科的基础。

通过定义表达问题实例所需的数据，以及得到预期结果所需的计算步骤，算法描述出了问题的解决方案。编程语言必须提供一种标记方式，用于表达过程和数据。为此，编程语言提供了众多的控制语句和数据类型。

控制语句使算法步骤能够以一种方便且明确的方式表达出来。算法至少需要能够进行顺序执行、决策分支、循环迭代的控制语句。只要一种编程语言能够提供这些基本的控制语句，它就能够被用于描述算法。

计算机中的所有数据实例均由二进制字符串来表达。为了赋予这些数据实际的意义，必须要有**数据类型**。数据类型能够帮助我们解读二进制数据的含义，从而使我们能从待解决问题的角度来看待数据。这些内建的底层数据类型（又称原生数据类型）提供了算法开发的基本单元。

举例来说，大部分编程语言都为整数提供了相应的数据类型。根据整数（如 23、654 以及–19）的常见定义，计算机内存中的二进制字符串可以被理解成整数。除此以外，数据类型也描述了该类数据能参与的所有运算。对于整数来说，就有加减乘除等常见运算。并且，对于数值类型的数据，以上运算均成立。

我们经常遇到的困难是，问题及其解决方案都过于复杂。尽管由编程语言提供的简单的控制语句和数据类型能够表达复杂的解决方案，但它们在解决问题的过程中仍然存在不足。因此，我