HZ BOOKS
华章教育

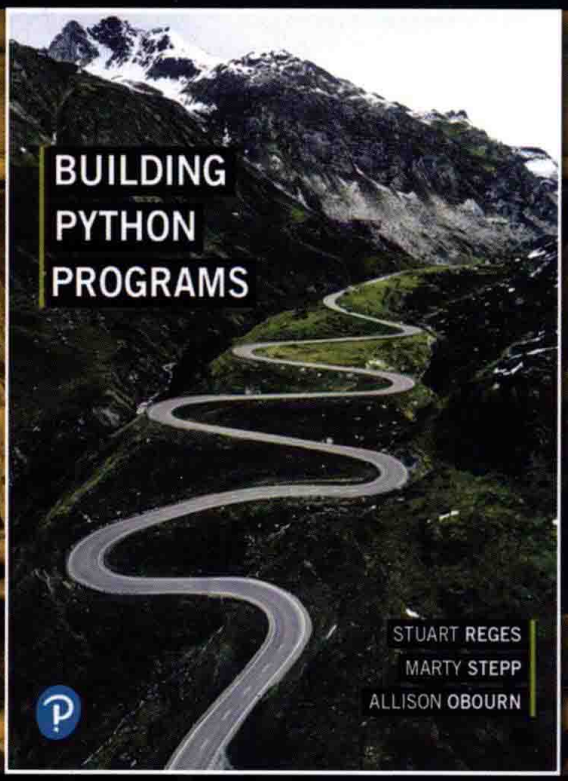
计　算　机　科　学　丛　书

Pearson

Python程序设计与算法思维

斯图尔特·里杰斯（Stuart Reges）
[美]　马蒂·斯特普（Marty Stepp）　著
艾利森·奥伯恩（Allison Obourn）
苏小红　袁永峰　叶麟　等译

注重程序分解和问题求解，强调算法思维，解决编写大型程序的难题



Building Python Programs

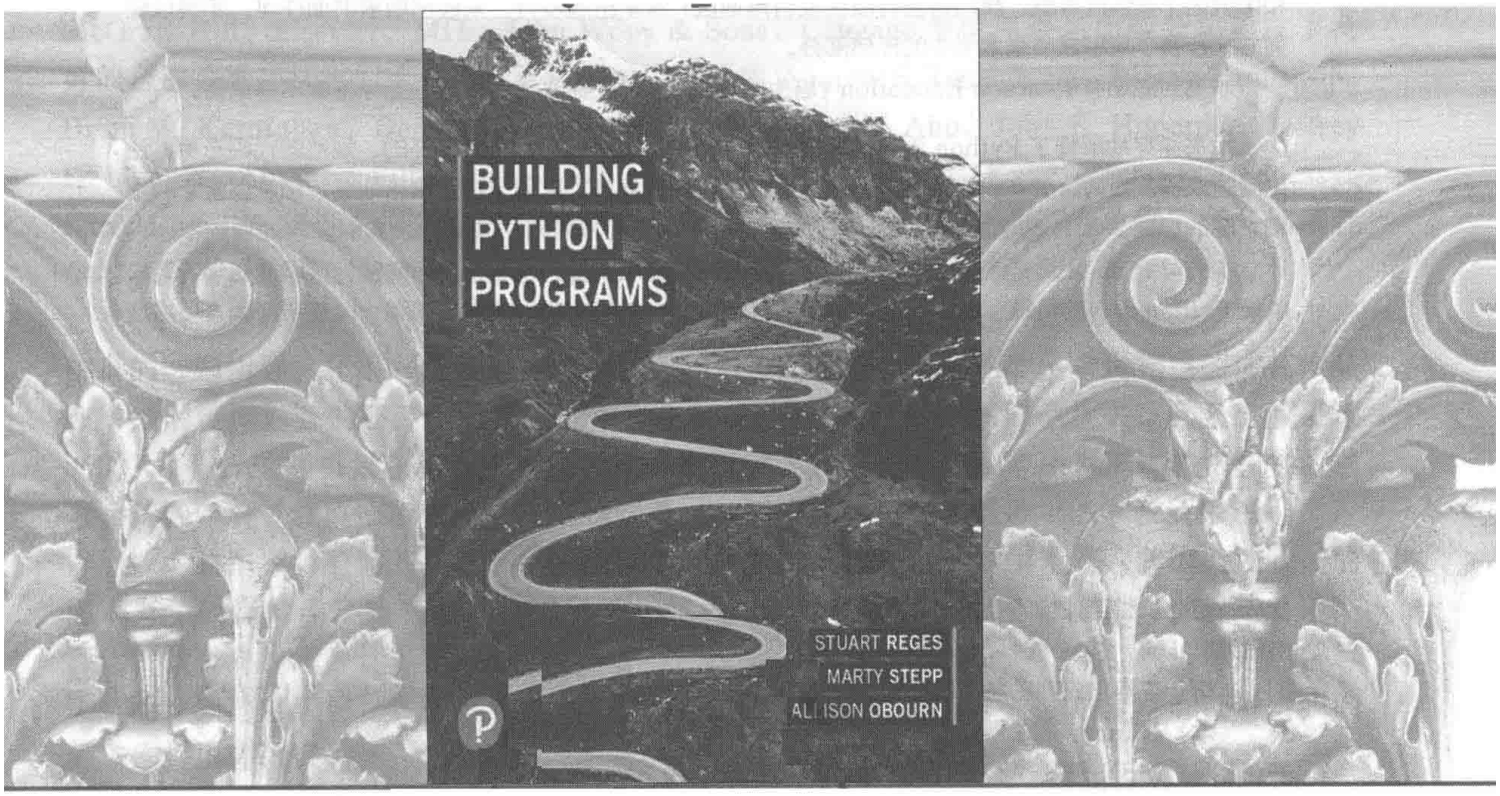
机械工业出版社
China Machine Press

计 算 丛 书

Python程序设计与算法思维

斯图尔特·里杰斯（Stuart Reges）
[美] 马蒂·斯特普（Marty Stepp） 著
艾利森·奥伯恩（Allison Obourn）
苏小红 袁永峰 叶麟 等译

Building Python Programs



机械工业出版社
China Machine Press

图书在版编目（CIP）数据

Python 程序设计与算法思维 /（美）斯图尔特·里杰斯（Stuart Reges），（美）马蒂·斯特普（Marty Stepp），（美）艾利森·奥伯恩（Allison Obourn）著；苏小红等译 . —北京：机械工业出版社，2020.5
（计算机科学丛书）
书名原文：Building Python Programs

ISBN 978-7-111-65514-5

I. P… II. ①斯… ②马… ③艾… ④苏… III. 软件工具－程序设计 IV. TP311.561

中国版本图书馆 CIP 数据核字（2020）第 076299 号

本书版权登记号：图字 01-2019-1418

本书详尽地解释了 Python 语言的每个新概念和每个语法细节，具有良好的、规范的代码示例，注重问题求解，强调算法实践。案例教学由简单到复杂递进展开，以便于读者清晰地理解和掌握整个编程和求解的思路。本书还增加了函数式编程内容，使初学者可以应对未来高并发实时多核处理的程序设计。本书对 Python 语言深入浅出、细致的讲解，以及课后大量的习题和编程实践，可以使初学者轻松掌握 Python 语言的精髓，并学以致用，解决科学研究、工程实践中的实际问题，以及切身体会程序设计之美。

出版发行：机械工业出版社（北京市西城区百万庄大街 22 号 邮政编码：100037）
责任编辑：游 静　　责任校对：李秋荣
印　　刷：三河市宏图印务有限公司　　版　　次：2020 年 6 月第 1 版第 1 次印刷
开　　本：185mm × 260mm 1/16　　印　　张：39.5
书　　号：ISBN 978-7-111-65514-5　　定　　价：139.00 元

客服电话：（010）88361066 88379833 68326294　　投稿热线：（010）88379604
华章网站：www.hzbook.com　　读者信箱：hzjsj@hzbook.com

文艺复兴以来，源远流长的科学精神和逐步形成的学术规范，使西方国家在自然科学的各个领域取得了垄断性的优势；也正是这样的优势，使美国在信息技术发展的六十多年间名家辈出、独领风骚。在商业化的进程中，美国的产业界与教育界越来越紧密地结合，计算机学科中的许多泰山北斗同时身处科研和教学的最前线，由此而产生的经典科学著作，不仅擘划了研究的范畴，还揭示了学术的源变，既遵循学术规范，又自有学者个性，其价值并不会因年月的流逝而减退。

近年，在全球信息化大潮的推动下，我国的计算机产业发展迅猛，对专业人才的需求日益迫切。这对计算机教育界和出版界都既是机遇，也是挑战；而专业教材的建设在教育战略上显得举足轻重。在我国信息技术发展时间较短的现状下，美国等发达国家在其计算机科学发展的几十年间积淀和发展的经典教材仍有许多值得借鉴之处。因此，引进一批国外优秀计算机教材将对我国计算机教育事业的发展起到积极的推动作用，也是与世界接轨、建设真正的世界一流大学的必由之路。

机械工业出版社华章公司较早意识到“出版要为教育服务”。自 1998 年开始，我们就将工作重点放在了遴选、移译国外优秀教材上。经过多年的不懈努力，我们与 Pearson、McGraw-Hill、Elsevier、MIT、John Wiley & Sons、Cengage 等世界著名出版公司建立了良好的合作关系，从它们现有的数百种教材中甄选出 Andrew S. Tanenbaum、Bjarne Stroustrup、Brian W. Kernighan、Dennis Ritchie、Jim Gray、Afred V. Aho、John E. Hopcroft、Jeffrey D. Ullman、Abraham Silberschatz、William Stallings、Donald E. Knuth、John L. Hennessy、Larry L. Peterson 等大师名家的一批经典作品，以“计算机科学丛书”为总称出版，供读者学习、研究及珍藏。大理石纹理的封面，也正体现了这套丛书的品位和格调。

“计算机科学丛书”的出版工作得到了国内外学者的鼎力相助，国内的专家不仅提供了中肯的选题指导，还不辞劳苦地担任了翻译和审校的工作；而原书的作者也相当关注其作品在中国的传播，有的还专门为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近 500 个品种，这些书籍在读者中树立了良好的口碑，并被许多高校采用为正式教材和参考书籍。其影印版“经典原版书库”作为姊妹篇也被越来越多实施双语教学的学校所采用。

权威的作者、经典的教材、一流的译者、严格的审校、精细的编辑，这些因素使我们的图书有了质量的保证。随着计算机科学与技术专业学科建设的不断完善和教材改革的逐渐深化，教育界对国外计算机教材的需求和应用都将步入一个新的阶段，我们的目标是尽善尽美，而反馈的意见正是我们达到这一终极目标的重要帮助。华章公司欢迎老师和读者对我们的工作提出建议或给予指正，我们的联系方法如下：

华章网站：www.hzbook.com
电子邮件：hzjsj@hzbook.com
联系电话：（010）88379604
联系地址：北京市西城区百万庄南街 1 号
邮政编码：100037

华章科技图书出版中心

译者序

Building Python Programs

纵观现代计算机从诞生之初发展到万物互联的当下，计算机编程语言在不同计算时代的交织变化中，或顺应时代大放异彩，或故步自封销声匿迹。比如，C 语言之于 Unix，HTML 之于 WWW，Java 之于 Android。计算如此多彩，引无数语言竞“挥毫”。正如历史画卷中的主角一样，计算时代造就了编程语言，编程语言代表了计算时代。而本书的 Python 语言恰恰是当下“数据为王，智能未来”时代的不二之选。来看看这些耳熟能详的名字，NumPy、Pandas、Matplotlib、SciPy、Scikit-Learn、TensorFlow、Keras，等等，Python 的出现让你与顶尖技术前沿不再相隔十万八千里。这也就是每每学生产生为什么要学习 Python 语言的疑问时，我们最想让大家有所体会、有所领略的精妙所在。

本书的译者在教授“高级语言程序设计（Python）”课程的过程中深感一本好教材的重要性。一本好教材不仅关注知识的传递，更注重对学生编程思维、解决问题能力的培养。本书作为面向零基础初学者的教科书，详尽地解释了 Python 语言的每个新概念和每个语法细节，具有良好的、规范的代码示例，注重问题求解，强调算法实践。案例教学由简单到复杂递进展开，以便于读者清晰地理解和掌握整个编程和求解的思路。本书还根据 ACM 与 IEEE 的计算学科本科课程指南（CS2013）的最新要求，增加了函数式编程的内容，使初学者可以应对未来高并发实时多核处理的程序设计。本书对 Python 语言深入浅出、细致的讲解，以及课后大量的习题和编程实践，可以使初学者轻松掌握 Python 语言的精髓，并学以致用，解决科学研究、工程实践中的实际问题，以及切身体会程序设计之美。

我们要感谢对本书翻译工作提供帮助的老师们。本书由苏小红、袁永峰、叶麟、张羽、孙承杰五位老师主译，他们都是从事程序设计语言课程教学的一线老师，具体分工如下：第 1、9、10 章由叶麟负责，第 2、3 章由孙承杰负责，第 4、11、12 章由袁永峰负责，第 5、8 章由苏小红负责，第 6、7 章由张羽负责。机械工业出版社华章公司的编辑张梦玲在本书的整个翻译过程中提供了许多帮助，在此予以衷心感谢。

译文虽经多次修改和校对，但由于译者的水平有限，加之时间仓促，疏漏及错误在所难免，我们真诚地希望读者不吝赐教，感激之至。

译　者

于哈尔滨工业大学

2020 年 3 月

Python 编程语言近年来已经变得非常受欢迎。能够快速学习 Python 简单直观的语法让人印象深刻，也让许多用户创建了流行的程序库。Python 由 Guido van Rossum 设计，Python 社区称他为“仁慈的独裁者”（BDFL）。他说选择 Python 这个名字“略带随意性”，同时也因为他是“《 Monty Python 飞行马戏团》（一部英国喜剧片）的忠实粉丝”。谁不想学习以一个喜剧演员团体名称命名的编程语言呢?

本书旨在用于计算机科学的第一门课程。我们对亚利桑那大学的数百名本科生进行了课堂测试，其中大多数不是计算机科学专业的学生。本教材采用了与我们之前编写的《 Java 程序设计教程（第 4 版）》相同的风格。Java 教材在我们的课堂测试中被证明是有效的，该测试覆盖自 2007 年以来在华盛顿大学学习的数千名学生。

计算机科学导论课程在许多大学有着悠久的历史，都是通过率很低的“杀手”课程。但正如道格拉斯·亚当斯在《银河系漫游指南》中所说的那样：“不要惊慌失措。”学生如果递进式学习，就可以掌握这门课程。

Python 拥有很多特性，这使其成为用于第一门计算机科学课程的有吸引力的语言。它具有简单、简洁但功能强大的语法，使得学习轻松愉快，并且非常适用于编写许多常用的程序。学生可以只用一行代码来编写自己的第一个 Python 程序，而不像 Java 或 C++ 这样的大多数语言需要编写若干行。Python 包含内置的解释器和“读取 – 求值 – 输出”循环（REPL，交互式解释器），用于快速运行和测试代码，鼓励学生测试和探索语言。Python 还提供了丰富的库，学生可以将它们用于图形、动画、数学、科学计算、游戏等。本书基于撰写时最新的语言版本 Python 3，覆盖了该语言版本的现代特性和习惯用法。

本书基于“回归基础”的方法，侧重于过程式编程和程序分解。这也被称为“对象在后”方法，而不是某些学校采用的“对象先行”方法。根据我们多年的经验，许多科学家、工程师等都可以学习过程式编程。一旦我们建立了面向过程的坚实技术基础，就转向面向对象的编程。在本书的最后，学生将学习这两种编程风格。

以下是我们的方法和材料的主要特点：

- **专注于解决问题**。许多教科书在介绍新构造时都会关注语言细节，我们则专注于解决问题。每个构造可以解决哪些新问题？新手可能遇到哪些陷阱？使用新构造的常见方法是什么？
- **强调算法思维**。过程式方法允许我们强调算法问题的解决：将大问题分解为更小的问题，使用伪代码来细化算法，并解决用算法表示大型程序的困难。
- **彻底讨论主题**。我们发现许多介绍性教材快速涵盖了新的语法和概念，然后就迅速进入下一个主题。我们觉得打开教科书的学生正是那种想要更彻底、更细致地解释和讨论棘手问题的学生。在本教材中，我们倾向于使用更长的解释，其中包含比其他教材更多的说明、图表和代码示例。
- **分层方法**。编程涉及许多难以一次学到的概念。教新手编写代码就像试图建造一个纸牌屋：每张新牌都必须小心放置。如果太匆忙而试图同时放置太多纸牌，整个结构就会崩溃。我们逐层教授新概念，让学生逐步加深对概念的理解。
- **强调良好的编码风格**。我们展示了使用正确和一致的编程风格，以及设计的代码。书

中显示的所有完整程序都已经过全面注释和正确分解。在整本书中，我们讨论了习惯用法，好的和坏的编程风格，以及如何选择优雅和适当的方法来分解和解决问题。

- **精心挑选的语言子集**。我们不是试图向学生展示每一种语言结构和特性，而是解释和使用我们认为最适合解决入门级问题的 Python 语言的核心子集。
- **案例研究**。我们通过一个典型的案例研究结束大多数章节，向学生展示如何分阶段开发复杂程序以及如何在开发过程中对其进行测试。这种结构允许我们在丰富的上下文中演示每个新的编程结构，这是无法通过短代码示例来实现的。

层次和依赖关系

许多介绍性的计算机科学教材都是面向语言的，但本书前几章的方法是分层的。例如，Python 有许多控制结构（包括循环和 if/else 语句），许多教材在一章中包含所有控制结构。虽然这对于已经知道如何编程的人来说可能有意义，但对于正在学习如何编程的新手而言，这可能带来压力。我们发现将这些控制结构分散到不同的章节来讲解更加有效，这样，学生可以一次学习一个结构，而不是一次性学习全部结构。

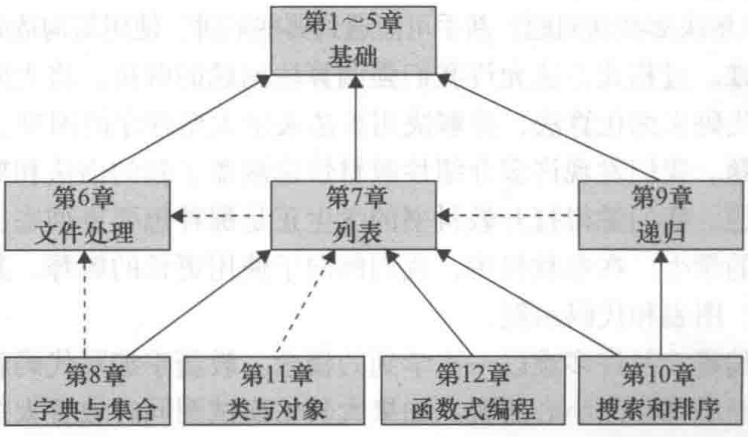
下表显示了前七章中的分层方法：

第 1 ～ 7 章的分层

章节	控制流	数据	技术	输入 / 输出
1	函数	字符串字面量	分解	print
2	确定循环（for 循环）	表达式 / 变量，整数，实数	局部变量，全局常量，伪代码	
3	参数，返回	使用对象	使用参数 / 返回值进行分解	控制台输入，图形
4	条件（if/else）	字符串	前置 / 后置条件，抛出异常	
5	不确定循环（while 循环）	布尔逻辑	断言，健壮的程序	
6		文件对象	基于行的处理，基于标记的处理	文件 I/O
7		列表	遍历	文件作为列表

第 1 ～ 5 章是按顺序进行设计的，然后从第 6 章开始具有更大的学习灵活性。尽管第 7 章（列表）中的案例研究涉及从文件中读取，而第 6 章介绍了该主题，但仍旧可以跳过第 6 章。

下图显示了各章的依赖关系。强依赖关系绘制为实箭头。我们建议学习时不要覆盖强依赖顺序之外的章节。弱依赖关系绘制为虚箭头。弱依赖关系表示后一章简要提到了前一章中的主题，但是如果必要的话，仍然可以阅读和探索这一章而不必考虑前面的章节。



各章依赖图

以下是各章之间弱依赖关系的更详细解释：

- 第 7 章中的一些示例，以及第 8 章中关于字典和集合的一些示例，会从文件中读取数据。文件输入 / 输出在第 6 章中介绍。但是，为了讨论列表或其他数据集合，不需要进行整体文件读取，因此如果需要，可以跳过第 6 章。
- 第 11 章中关于类和对象的一些示例提到了引用语义的概念，它在第 7 章中介绍。但是在第 11 章中重新解释了引用的概念，因此如果需要，可以在学习列表之前先学习类。
- 第 9 章中的一些递归函数会处理列表，并且一个递归函数会以递归方式反转文件的所有行。因此第 9 章在一定程度上依赖于第 7 章。但是，第 9 章中的几乎所有递归函数都可以只使用第 1 ～ 5 章的核心内容来编写和理解。

从图中可以看出，第 7 章可能是前五章之后最重要的章节，其内容被许多其他章节使用。常见的章节交换顺序是先学习第 1 ～ 5 章，然后学习第 7 章，再返回第 6 章学习关于文件的额外知识。

补充材料[⊖]

所有自测题的答案都在本书配套网站 http://www.buildingpythonprograms.com/ 中提供。

此外，配套网站还为学生提供了以下额外资源：

- 在线的补充内容。
- 所有案例研究的源代码和数据文件以及其他完整的程序示例。
- 第 3 章中使用的 DrawingPanel 类。
- 链接到基于 Web 的编程练习工具。

教师可以访问以下资源[⊜]：

- 适用于讲课的 PowerPoint 幻灯片。
- 习题和编程项目的解决方案，以及许多项目的家庭作业规范文档。
- 考试题和解决方案的关键点。

要访问教师资源，请发送电子邮件至 authors@buildingpythonprograms.com 与我们联系。有关资源的其他问题，请联系作者或 Pearson 代表。

致谢

感谢 Pearson 的工作人员，他们帮助制作了这本书。Rose Kernan 管理该项目，是我们在图书制作过程中的主要联系人。Rose 做了非凡的工作，她在这个过程中始终勤奋、积极、乐于助人。Amanda Brands 是内容制作人，她在此过程中提供了出色的支持。感谢 Martha McMaster 对本教材进行校对，感谢 Shelly Gerger-Knechtl 进行文案编辑和索引编制。感谢营销经理 Yvonne Vannatta 和编辑助理 Meghan Jacoby。还要感谢 Pearson 的艺术和合成团队，他们对本书进行了排版。

感谢 Pearson 的主编 Matt Goldstein。十多年前，Matt 就认可了我们的工作，并与我们

⊖ 关于配套网站资源，大部分需要访问码，访问码只有原英文版提供，中文版无法使用。——编辑注

⊜ 关于本书教辅资源，只有使用本书作为教材的教师才可以申请，需要的教师请联系机械工业出版社华章公司，电话 010-88378991，邮箱 wangguang@hzbook.com。——编辑注

合作出版了《Java 程序设计教程》第 1 版。Matt 一直是一个坚定的支持者，我们总是很乐意与他合作。

最后但同样重要的是，我们要感谢亚利桑那大学的 CSC 110 学生，他们对本书的草稿进行了课堂测试，提供了有用的建议，并改正了草稿中的错误。

Stuart Reges，华盛顿大学
Marty Stepp，斯坦福大学
Allison Obourn，亚利桑那大学

第 1 章
Building Python Programs

Python 编程简介

本章首先回顾一些关于计算机和计算机编程的基本术语。其中，许多概念将在后面的章节中出现，所以我们在开始深入研究如何使用 Python 编程的细节之前，先来复习一下这些概念。

我们将通过阅读将文本输出到控制台窗口的简单程序来开始 Python 的探索之旅。我们将探索所有 Python 程序都会出现的许多元素，同时使用的程序在结构上又相当简单。

在回顾了 Python 程序的基本元素之后，接下来我们将学习如何将 Python 程序分解为多个函数，以探索程序分解技术。使用这种技术，可以将复杂的任务分解为更易于管理的较小子任务，并且可以避免程序中的冗余。

1.1 计算的基本概念

如今，在我们的日常生活中计算机的使用已非常普遍，并且互联网使我们能够访问几乎无穷无尽的信息。其中一些信息是重要新闻，如 cnn.com 的新闻头条。计算机可以让我们与家人分享照片，并将路线导航到最近的比萨店去吃晚餐。

许多实际问题正在通过计算机得到解决，其中一些问题与台式机或笔记本电脑中的问题不太一样。计算机允许我们对人类基因组进行排序并在其中搜索 DNA 模式。最近制造的车载计算机可以监控每辆车的状态和运动。数字设备，例如 Apple 的 iPhone，实际上在其小巧的外壳内同样安装有计算机。即使是 Roomba 真空清洁机器人也有执行复杂指令的计算机，用来控制机器人在清洁地板时如何躲避家具。

但是，什么使计算机成为计算机？计算器是计算机吗？用纸和笔的人是计算机吗？接下来的若干小节将试图解决这个问题，同时介绍一些有助于你学习编程的基本术语。

1.1.1 为何编程

在大多数大学里，计算机科学的第一门课程是编程。许多计算机科学家对此感到困扰，因为它给人留下一种计算机科学就是编程的印象。虽然许多训练有素的计算机科学家确实花大量时间进行编程，但计算机学科还有很多其他内容。那么，我们为什么要先学习编程呢？

一位名叫 Don Knuth 的斯坦福大学计算机科学家回答了这个问题，他说大多数计算机科学家的共同点是我们都在某种程度上使用**算法**。

> **算法**
> 如何完成一项任务的逐步描述。

Knuth 是算法方面的专家，所以他自然会偏向于将算法视为计算机科学的中心。不过，他声称最重要的不是算法本身，而是计算机科学家用来开发它们的思维过程。Knuth 的说法是：

经常有人说，一个人在将事物教给别人之后才会真正理解它。实际上，一个人在将事物教给计算机之后才会真正理解它，即将其表达为算法。[⊖]

Knuth 描述的是大多数计算机科学中常见的思维过程，他将其称为算法思维。我们研究编程不是因为它是计算机科学最重要的方面，而是因为它是解释计算机科学家解决问题的方法的最佳方式。

算法的概念有助于理解计算机是什么以及计算机科学是什么。Merriam-Webster 字典将“计算机”一词定义为“可以进行计算的东西”。使用该定义，各种设备都可以作为计算机，从计算器到 GPS 导航系统再到儿童玩具。在发明电子计算机之前，通常将人称为计算机。例如，19 世纪的数学家查尔斯·皮尔斯最初被聘为美国政府的“助理计算机”，因为他的工作涉及数学计算。

从广义上讲，“计算机”一词可以应用于许多设备。但是当计算机科学家提到计算机时，他们通常会想到一种可以编程执行任何算法的通用计算设备。因此，计算机科学是计算设备的研究和计算本身的研究，包括算法。

算法可以表示为计算机程序，这就是本书的全部内容。但在我们研究如何编程之前，有必要回顾一些有关计算机的基本概念。

1.1.2　硬件和软件

计算机是一种操纵数据并执行一系列指令（**程序**）的机器。

程序
计算机可以执行的指令列表。

将计算机与计算器等简单机器区分开来的一个关键特性是其多功能性。同一台计算机可以执行许多不同的任务（玩游戏，计算所得税，连接到世界各地的其他计算机），具体取决于它在特定时刻运行的程序。计算机不仅可以运行当前存在的程序，还可以运行尚未编写完成的新程序。

组成计算机的物理组件统称为**硬件**。最重要的硬件之一是中央处理单元（CPU）。CPU 是计算机的“大脑”：执行指令的就是它。同样重要的是计算机的**内存**（通常称为随机存取存储器（RAM），因为计算机可以随时访问内存的任何部分）。计算机使用内存来存储正在执行的程序及其数据。RAM 的大小有限，并且在计算机关闭时不保留其内容。因此，计算机通常还使用**硬盘**作为更大的永久存储区域。

计算机程序统称为**软件**。在计算机上运行的主要软件是操作系统。**操作系统**提供了一个许多程序可以同时运行的环境，它也提供了程序、硬件和用户（使用计算机的人）之间的桥梁。在操作系统内运行的程序通常称为**应用程序（app）**。

当用户选择要运行操作系统中的一个程序时（例如，通过双击桌面上的程序图标），会发生以下情形：该程序的指令从硬盘加载到计算机的内存中，操作系统为该程序分配内存，运行程序的指令从内存送到 CPU 并按顺序执行。

⊖ Knuth, Don. *Selected Papers on Computer Science*. Stanford. CA: Center for the Study of Language and Information, 1996.

1.1.3 数字领域

在上一节中，我们看到计算机是可以编程的通用设备。正是由于其运行方式，你经常会听到人们将现代计算机称为**数字**计算机。

> **数字**
> 基于以离散增量增加的数字，例如整数 0，1，2，3 等。

因为计算机是数字的，所以存储在计算机上的所有内容都存储为整数序列。这包括每个程序和每一份数据。例如，MP3 文件只是存储音频信息的一长串整数。今天我们已经习惯了数字音乐、数字图片和数字电影，但是在 20 世纪 40 年代，当构建第一台计算机时，以整数形式存储复杂数据的想法是相当不寻常的。

计算机不仅是数字的，将所有信息存储为整数，而且它也是**二进制的**，这意味着它将整数存储为**二进制数**。

> **二进制数**
> 仅由 0 和 1 组成的数字，也称为基数为 2 的数字。

人类通常使用**十进制**或基数为 10 的数字，这符合我们的生理学（10 个手指和 10 个脚趾）。但是，当设计第一台计算机时，我们希望系统易于创建并且非常可靠。事实证明，将系统建立在二元现象（例如，电路打开或关闭）之上，要比建立在必须彼此区分的 10 种不同状态（例如，10 种不同的电压水平）之上要简单得多。

从数学的角度来看，你可以使用二进制数轻松存储内容，就像使用十进制数一样。同时，构建使用二进制数的物理设备更容易，这就是计算机使用二进制数的原因。

然而，这确实意味着不熟悉计算机的人就不熟悉其原理。因此，值得花一点时间来回顾二进制数是如何工作的。要使用二进制数进行计数，与十进制数一样，从 0 开始计数，但是用完数字的速度要快很多。

所以，以二进制计数，你说：

```
0
1
```

然后就已经没有数字了。这就像以十进制计数达到 9 时一样。用完这些数字后，转到下一个数位。所以，接下来的两个二进制数是：

```
10
11
```

然后，你又没有数字了。这就像以十进制计数达到 99 时一样。接着，你继续到下一个数位形成三位数字 100。在二进制中，每当你看到一串数字，如 111111，你知道只要再加个 1 就可以使原先所有数位都变为 0 并且前面加个 1，就像十进制一样，当你看到一个像 999999 这样的数字时，你知道只要再加个 1，原先所有数位都变为 0 并且前面加个 1。表 1-1 显示了如何使用二进制数来对十进制中的数字 0 ～ 8 进行计数。