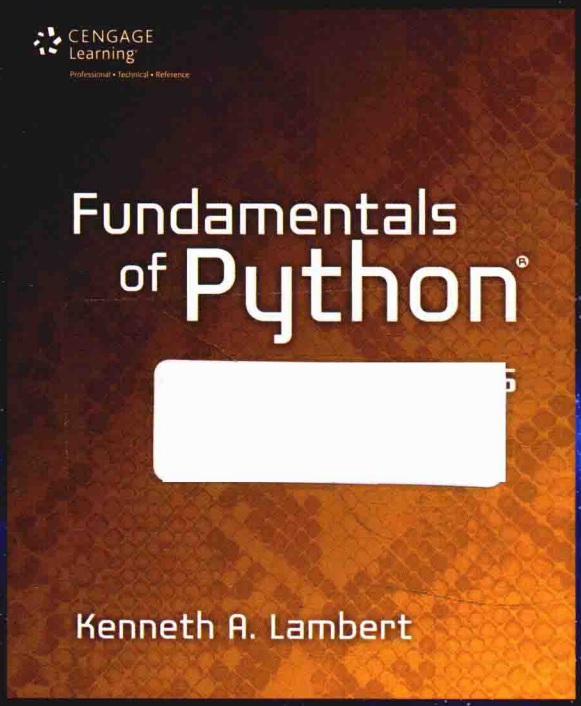
国外著名高等院校
信息科学与技术优秀教材

CENGAGE Learning

异步图书
www.epubit.com.cn

数据结构
（Python语言描述）

[美] Kenneth A. Lambert 著　李军 译



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

国外著名高等院校
信息科学与技术优秀教材

数据结构
（Python语言描述）

[美] Kenneth A. Lambert 著　李军 译

人民邮电出版社
北京

图书在版编目（CIP）数据

数据结构 : Python语言描述 / (美) 兰伯特 (Kenneth A. Lambert) 著 ; 李军译. -- 北京 : 人民邮电出版社, 2017.12 (2018.2重印)
国外著名高等院校信息科学与技术优秀教材
ISBN 978-7-115-46461-3

Ⅰ. ①数… Ⅱ. ①兰… ②李… Ⅲ. ①数据结构－高等学校－教材②软件工具－程序设计－高等学校－教材
Ⅳ. ①TP311.12②TP311.561

中国版本图书馆CIP数据核字(2017)第190907号

版权声明

978-1-285-75200-6
Cengage Learning Asia Pte.Ltd.
151 Lorong Chuan, #02-08 New Tech Park, Singapore 556741
本书封面贴有 **Cengage Learning** 防伪标签，无标签者不得销售。

◆ 著　[美] Kenneth A. Lambert
译　李　军
责任编辑　陈冀康
责任印制　焦志炜

◆ 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
北京市艺辉印刷有限公司印刷

◆ 开本：787×1092　1/16
印张：20
字数：462 千字　2017 年 12 月第 1 版
印数：3 901-6 900册　2018 年 2 月北京第 3 次印刷

著作权合同登记号　图字：01-2017-3664 号

定价：69.00 元

读者服务热线：(010)81055410　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

前　言

欢迎使用本书。本书是作为通过数据结构学习编程并解决问题课程的教材。本书所介绍的内容，通常是本科生的 CS2 课程所要教授的内容。尽管本书使用 Python 编程语言，但是，你只需要掌握这一高级编程语言的基础知识，也就是本书第 1 章所介绍的内容就可以了。

读者将学到的内容

本书介绍了计算的 4 个主要方面：

- 编程基础——数据类型、控制结构、算法开发以及使用函数的程序设计，这些是在使用计算机来解决问题的时候需要掌握的基本思想。我们将使用 Python 编程语言来介绍这些基本的主题，并且通过理解它们来解决更广泛的问题。
- 面向对象编程——面向对象编程是用于开发大型软件系统的主要的编程范型。本书将介绍 OOP 的基本原理，以便你能够成功地应用它们。和其他的教材不同，本书帮助你开发专业品质的集合类框架，从而阐述这些原理。
- 数据结构——大多数有用的程序都依赖于数据结构来解决问题。在最具体的层级，数据结构包括数组和各种类型的链表结构，我们将使用这些数据结构来实现各种不同类型的集合结构，例如栈、队列、列表、树、包、集、字典和图。我们还将学习使用复杂度分析来评估这些集合的不同实现的空间/时间代价。
- 软件开发生命周期——本书会在众多的案例学习的整个过程中来介绍软件开发技术，而不是将软件开发技术单独分割到一章或两章之中。除此之外，我们将学习编写一个程序的代码，但是这个程序往往不是问题解决或软件开发的最难或最有挑战性的部分。

为什么使用 Python

在过去的 20 年里，计算机技术和应用程序已经变得越来越高级，计算机科学课程也是如此，特别是入门阶段的课程。今天的学生在学习了一些编程和解决问题的方法之后，就期望快速转向软件开发、复杂度分析和数据结构等主题，而 20 年前，这些都是纳入到高级课程中学习的。此外，面向对象编程成为主要的范型，这导致教材的编写者将诸如 C++和 Java 这样强大的、工业级的编程语言引入到入门性的课程中。结果，刚开始学习计算机科学的学生，往往被同时要掌握高级概念和编程语言语法这样的综合性的任务搞得不知所措，而并没有体验到用计算机解决问题的成就感和兴奋感。

本书使用 Python 编程语言作为讲解计算机科学的第二门课程，这种方法更具有可操作性，而且对于学生和讲师似乎更有吸引力。Python 具有以下几个教学优点：

- Python 拥有简单的、符合惯例的语法。Python 语句和那些伪代码算法很接近，并且 Python 表达式使用代数中所能找到的惯用的表示法。因此，读者花很少的时间就能够掌握 Python 的语法，而把较多的时间用于解决有趣的问题。
- Python 具有安全的语意。含义违反语言定义的任何表达式或语句，都会导致错误消息。
- Python 的扩展性很好。初学者可以很容易地用 Python 编写简单的程序。Python 还包括了现代编程语言的所有高级功能，例如，支持数据结构和面向对象的软件开发，当这些技术变得必要的时候，就可以使用它们。
- Python 有很强的可交互性。用户可以在解释器的提示符窗口中输入表达式，以验证实验性的代码，并且会立即接受到反馈。也可以组合较长的代码段，并且将它们保存到脚本文件中，以作为模块或独立的应用程序加载。
- Python 具备通用的用途。在今天的环境中，这意味着，该语言包括可用于最新应用程序的资源，包括媒体计算和 Web 服务等。
- Python 是免费的，并且广泛用于工业界。你可以下载 Python 以便在各种设备上运行。Python 的用户群体很大，从事 Python 编程的专业人士很容易找到好工作。

总结这些优点，无论对于初学者还是对于专家来说，Python 都是用于表达计算思想的合适的、灵活的工具。如果你在第一年里很好地学习了这些思路，那么在学习以后课程的时候，要快速转换到其他的语言，应该也不会有什么问题。更为重要的是，你在计算机上花费的时间将会很少，而可以将更多的时间用于思考解决有趣的问题。

本书的组织结构

本书按照易于学习的方式来组织，只有在需要的时候，才会介绍每一种新的概念。

第 1 章给出了 Python 编程功能的概览，这是想要开始使用 Python 学习编程和解决问题所需的基础。本章的内容按照易于学习的方式来组织，如果你拥有 Python 编程经验，可以快速跳过这些内容；如果你是 Python 新手，则可以学习得深入一点以快速掌握这门语言。

本书剩下的部分，从第 2 章到第 12 章，介绍了典型的 CS2 课程的主要话题，特别是抽象数据类型的规范、实现和应用，并且给出了作为主要的工具和关注点的集合类型。在这个过程中，你将完全地学习面向对象编程技术和良好的软件设计的要素。其他重要的 CS2 主题包括数据的递归过程、搜索和排序算法，以及在软件开发中使用的工具，例如复杂度分析和用来进行文档设计的图形化表示方法（UML）。

第 2 章介绍了抽象数据类型的概念，并且针对集合 ADT 的各种分类给出了一个概览。

第 3 章和第 4 章介绍了用于实现大多数集合的数据结构，以及分析它们的性能代价的工具。第 3 章介绍了使用大 O 表示法来进行复杂度分析，还给出了足够的资料，通过使用搜索和排序算法作为示例，使你能够对算法和数据结构的运行时间和内存使用进行简单的分

析。第 4 章详细介绍了如何处理数组和线性链表结构，这是用于实现大多数集合的具体的数据结构。我们将学习支持数组和链表结构的底层的计算机内存模式，以及它们所需的时间/空间代价。

第 5 章和第 6 章将关注点转移到了面向对象设计。这些原理用来组织一个高品质的集合类框架，后面的各章将会详细介绍该框架。

第 5 章关注接口和实现之间的重要差异。我们开发了一个包集合的单个接口和几个实现，来作为第一个示例。本章强调了要在一个接口中包含的通用方法，以允许不同类型的集合在应用程序中协作。例如，创建迭代器的方法，该迭代器允许使用一个简单的循环来遍历任何的集合。本章介绍的其他的主题包括多态和信息隐藏，它直接源自于接口和实现之间的差异。

第 6 章展示了类继承如何能够减少面向对象的软件系统中的冗余代码的数量。这里介绍继承、方法调用的动态绑定以及抽象类等相关概念，这些概念还将会在整个剩下的各章中使用。

学习了这些概念和原理，我们就准备好了了解其他主要的集合 ADT 了，它们构成了第 7 章到第 12 章的主题。

第 7 章到第 9 章介绍了线性集合，包括栈、队列和列表。每一个集合都首先从用户的视角来看，用户只会关注所选实现的接口和性能特征。每一个集合都通过一个或多个应用来介绍。最后，本章还开发了几个实现，并且对它们的性能代价进行了分析。

第 10 章到第 12 章介绍了高级数据结构以及一些相关算法，以便过渡到计算机科学随后的课程中。第 10 章介绍了各种树结构，包括二叉搜索树、堆和表达式树。第 11 章使用哈希策略，介绍了无序集合、包、集和字典的实现。第 12 章介绍了图和图处理算法。

正如前面所提到的，本书的独特之处在于展示了集合类型的一个专业框架。你将会探讨每一个集合在整个架构中的位置，而不止是认识一系列似乎不相关的集合。这个方法允许你查看集合类型有什么共同性，以及每一个集合有什么独特性。同时，你还将接触到继承和类层级的实际使用，这是面向对象软件设计的主题，但很少会在这个层级的课程中讲解和阐述。

本书特点

本书通过使用众多的示例和图表仔细地阐述和介绍了概念。新的概念总是应用于完整的程序之中，以展示它们是如何解决问题的。本章还尽早地并且一贯性地强调了编写干净、可读的文档的好习惯。

本书包含了几个重要的特点：

- 案例学习——这部分展示了完整的 Python 程序，范围从简单的程序到实用的程序。为了强调软件开发生命周期的重要性和有用性，案例学习按照用户需求、分析、设计、实现、对测试的建议的框架来讨论，在每个阶段都有精确定义的任务。一些案例学习还在每章末尾的编程项目中进行了扩展。
- 每章小结——每章最后都有对该章介绍的主要概念的一个概览。

- 关键术语——当书中引入一个新的术语的时候，用粗体显示。
- 练习——大部分章节中都有练习，通过询问该节内容相关的基本问题来巩固读者的学习。每章最后都有复习题。
- 编程项目——每章最后都有一组不同难度的编程项目。
- 附录——附录部分包含了本书中使用的集合框架的相关信息。

意见反馈

我们尝试编写高质量的图书，但是，还是可能会有一些错误，请通过 lambertk@wlu.edu 及时反馈这些错误。http://home.wlu.edu/~lambertk/python/列出了已发现的一些错误，以及关于本书的一些信息。

配套文件下载

可以从 www.epubit.com.cn 下载本书的配套文件。
选用本书作为教材的老师，可通过 contact@epubit.com.cn 申请教学 PPT。

目　录

第 1 章　Python 编程基础

本章给出了 Python 编程的一个快速概览。本章专门帮助那些 Python 初学者和不熟悉 Python 的人尽快上手，但本章并不打算完全地介绍计算机科学或 Python 编程语言。要了解 Python 编程的更多细节，请阅读我的另一本图书《Fundamentals of Python：First Programs》（Course Technology/Cengage Learning, 2012）。要了解 Python 编程语言的相关文档，请访问 www.python.org。

如果你已经安装了 Python，请在命令提示符窗口运行 `python` 或 `python3` 命令来检查 Python 的版本号（Linux 和 Mac 用户首先要打开终端窗口，Windows 用户首先要打开一个 DOS 窗口）。最好使用 Python 的最新版本。请查看 www.python.org，尽可能下载和安装最新的版本。要运行本书中的程序，你需要 Python 3.0 或更高的版本。

1.1　基本程序要素

和所有的现代编程语言一样，Python 也有大量的功能和构造。然而，Python 是少数的几种基本程序要素相当简单的语言之一。本节将开始介绍使用 Python 编程的一些基础知识。

1.1.1　程序和模块

Python 程序包含一个或多个模块（module）。模块只不过是 Python 代码的一个文件，其中可以包含语句、函数定义和类定义。简短的 Python 程序也称为脚本（script），可以包含在一个模块之中。而较长的或较为复杂的程序，通常包含一个主模块和一个或多个支持模块。主模块包含了程序执行的起点，支持模块则包含了函数和类的定义。

1.1.2　Python 程序示例：猜数字

接下来，我们来看一个完整的 Python 程序，它是一个和用户玩猜数字的游戏的程序。计算机要求用户输入数值范围的最小值和最大值。计算机随后“思考”出在这个范围之内的一个随机数，并且重复地要求用户猜测这个数，直到用户猜对了。在用户每次进行猜测之后，计算机都会给出一个提示，并且会在这个过程的最后显示出总的猜测次数。这个程序包含了几种类型的 Python 语句，例如，输入语句、输出语句、赋值语句、循环和条件语句，我们将在本章的后面介绍它们。这个程序还包含了一个函数的定义。

如下是该程序的代码，代码位于文件 `numberguess.py` 之中：

```
"""
Author: Ken Lambert
```

```
Plays a game of guess the number with the user.
"""

import random

def main():
    """Inputs the bounds of the range of numbers
    and lets the user guess the computer's number until
    the guess is correct."""
    smaller = int(input("Enter the smaller number: "))
    larger = int(input("Enter the larger number: "))
    myNumber = random.randint(smaller, larger)
    count = 0
    while True:
        count += 1
        userNumber = int(input("Enter your guess: "))
        if userNumber < myNumber:
            print("Too small")
        elif userNumber > myNumber:
            print("Too large")
        else:
            print("You've got it in", count, "tries!")
            break

if __name__ == "__main__":
    main()
```

如下是一个用户和该程序的交互记录：

```
Enter the smaller number: 1
Enter the larger number: 32
Enter your guess: 16
Too small
Enter your guess: 24
Too large
Enter your guess: 20
You've got it in 3 tries!
```

1.1.3 编辑、编译并运行 Python 程序

你可以在一个终端窗口中输入一条命令，来运行整个 Python 程序。例如，运行本书中给出的大多数示例。要运行 numberguess.py 文件中包含的程序，在大多数的终端窗口中，只要输入如下的命令就可以了：

```
python3 numberguess.py
```

要创建或编辑 Python 模块，尝试使用 Python 的 IDLE（Integrated DeveLopment Environment，集成开发环境）。要启动 IDLE，在一个终端提示符后输入 idle 或 idle3，或者单击其图标。也可以通过在某个 Python 源代码文件（以 .py 为扩展名的任何文件）上单击，或者通过在一个文件上点击鼠标右键并选择“Open or Edit with IDLE”，来启动 IDLE。确保将你的系统设置为在加载这种类型的文件的时候会打开 IDLE。

IDLE 提供了一个 shell 窗口，可以交互式地运行 Python 表达式和语句。使用 IDLE，可以在编辑器窗口和 shell 窗口之间来回切换，以开发和运行完整的程序。IDLE 还能够格式化代码，并为代码提供颜色区分。

当你用 IDLE 打开已有的 Python 文件的时候，文件会出现在编辑器窗口中，并且 shell 会在一个单独的窗口弹出。要运行一个程序，将光标移动到编辑器窗口中，并且按下 F5 键。Python 会在编辑器窗口中编译代码，并且在 shell 窗口中运行它。

当运行包含几个 Python 模块的一个程序的时候，针对每个模块（除了主模块之外）编译后的代码，会被保存到一个二进制代码文件中（以 .pyc 为扩展名的一个文件）。Python 加载这些文件，以便运行随后的程序（如果没有对相应的 .py 文件做出修改的话）。

如果一个 Python 程序挂起了或者没有正常结束的话，可以按下 Ctrl+C 键或关闭 shell 窗口来退出。

1.1.4 程序注释

程序注释是 Python 编译器会忽略的文本，但是，作为文档，它对于程序的阅读者很有价值。Python 中的单行注释以一个#符号开头，并且到当前行的末尾结束。它的颜色是红色的（尽管由于本书是黑白印刷的，你并看不出来这一点）。如下所示：

```
# This is an end-of-line comment.
```

一个多行注释是用 3 个单引号或 3 个双引号括起来的一个字符串。这样的注释也叫作文档字符串（docstring），表明它们可以记录程序中的主要结构。前面给出的猜数字程序就包含了两个文档字符串。第一个文档字符串在程序文件的顶部，充当了整个 numberguess 模块的注释。第二个文档字符串在主函数的头部之下，描述了这个函数做什么事情。尽管文档字符串看上去很简短，但是它们能在 Python shell 中为程序员提供帮助，因此扮演了一个重要的角色。

1.1.5 词法元素

语言中的词法元素是用于构造语句一类的单词和符号。和所有的高级编程语言一样，Python 的一些基本符号是关键字，例如 if、while 和 def。还包括了其他的一些词法元素，例如标识符（名称）、字面值（数字、字符串和其他内建的数据结构）、运算符和分隔符（引号、逗号、圆括号、方括号和花括号）。

1.1.6 拼写和命名惯例

Python 关键字和名称都是区分大小写的。因此，while 是一个关键字，而 While 则是程序员定义的一个名称。Python 关键字是以小写字母的方式拼写的，并且在 IDLE 窗口中以橙色表示。

所有的 Python 名称，其颜色都是黑色的，除非将其作为函数、类或方法名来引用，在这种情况下，它们显示为蓝色。名称可以以一个字母或一个下划线（'_'）开头，后面跟着任意多个字母、下划线或者数字。

在本书中，模块、变量、函数和方法的名称，都是以小写字母的形式来拼写的。模块名只有一种例外，就是当一个模块名包含了一个或多个嵌入的名称的时候，嵌入的名称都是

大写的。类名也遵从相同的命名惯例，但是，类名的首字母是大写的。当一个变量名是一个常量的时候，其所有的字母都是大写的，并且会有一个下划线用来隔开任何嵌入的名称。表 1.1 给出了这些命名惯例的示例。

表 1.1 Python 命名惯例的示例

名称类型	示例
变量	`salary, hoursWorked, isAbsent`
常量	`ABSOLUTE_ZERO, INTEREST_RATE`
函数或方法	`printResults, cubeRoot, isEmpty`
类	`BankAccount, SortedSet`

要使用能够描述其在程序中的角色的名称。通常，变量名应该是名词或形容词（如果它们表示布尔值的话，就是形容词），而函数和方法名应该是动词（如果它们表示动作）或者名词或形容词（如果它们表示返回的值）。

1.1.7 语法元素

一种语言的语法元素是由词法元素组成的语句（表达式、语句和其他结构）的类型。和大多数的高级语言不同，Python 使用空白（空格、制表符和换行）来标记多种类型的语句的语法。这意味着，在 Python 代码中，缩进和换行是很重要的。像 IDLE 这样的智能编辑器，能够帮助正确地缩进。程序员不需要操心用分号隔开句子，或者用花括号来标记语句块。在本书所有的 Python 代码中，我使用 4 个空格宽度的缩进。

1.1.8 字面值

数字（整数或浮点数）的写法和在其他编程语言中的写法都是一样的。布尔值 `True` 和 `False` 都是关键字。一些数据结构，例如字符串、元组、列表和字典，也拥有字面值，稍后我们将会看到这一点。

1.1.9 字符串字面值

可以用单引号、双引号，或者成对的三个双引号或三个单引号将字符串括起来。最后的这种表示方法，对于包含多行文本的字符串来说，是很有用的。字符值是单字符的字符串。\字符用于将非图形化的字符（例如，换行\n 和制表符\t，或者\字符本身）进行转义。下面的代码段及其输出，展示了各种可能性：

```
print("Using double quotes")
print('Using single quotes')
print("Mentioning the word 'Python' by quoting it")
print("Embedding a\nline break with \\n")
print("""Embedding a
line break with triple quotes""")
```