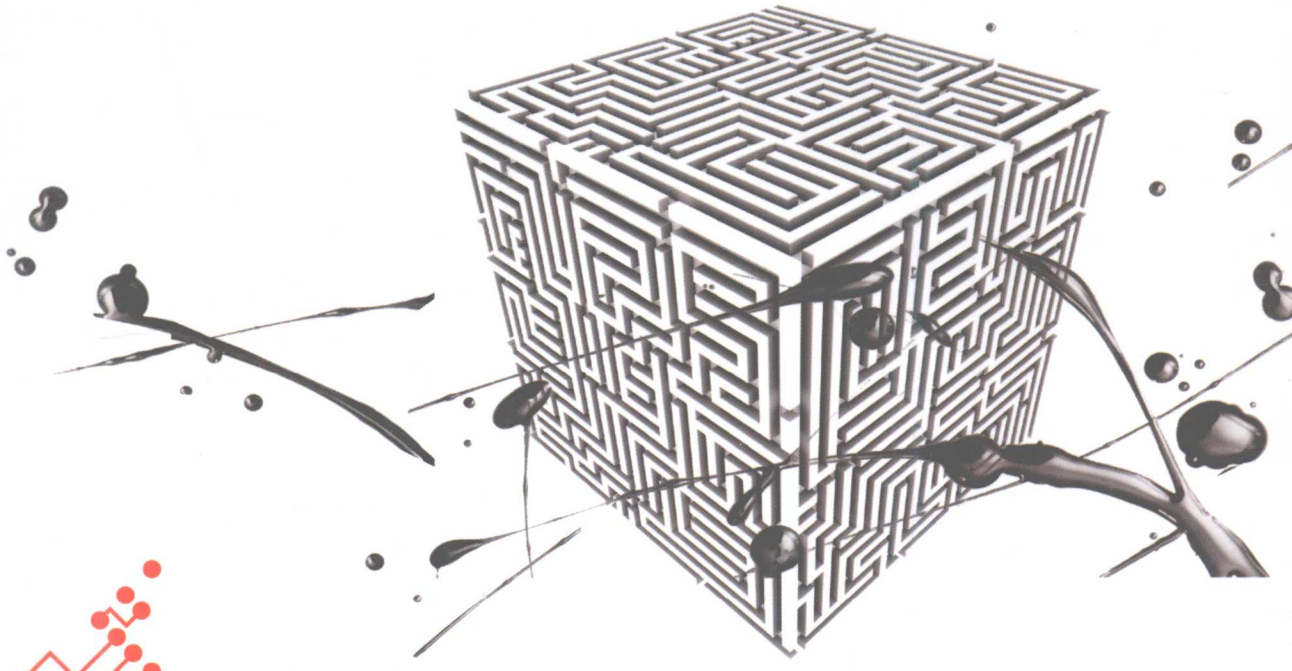


普通高等院校计算机基础教育“十三五”规划教材

Python

程序设计教程

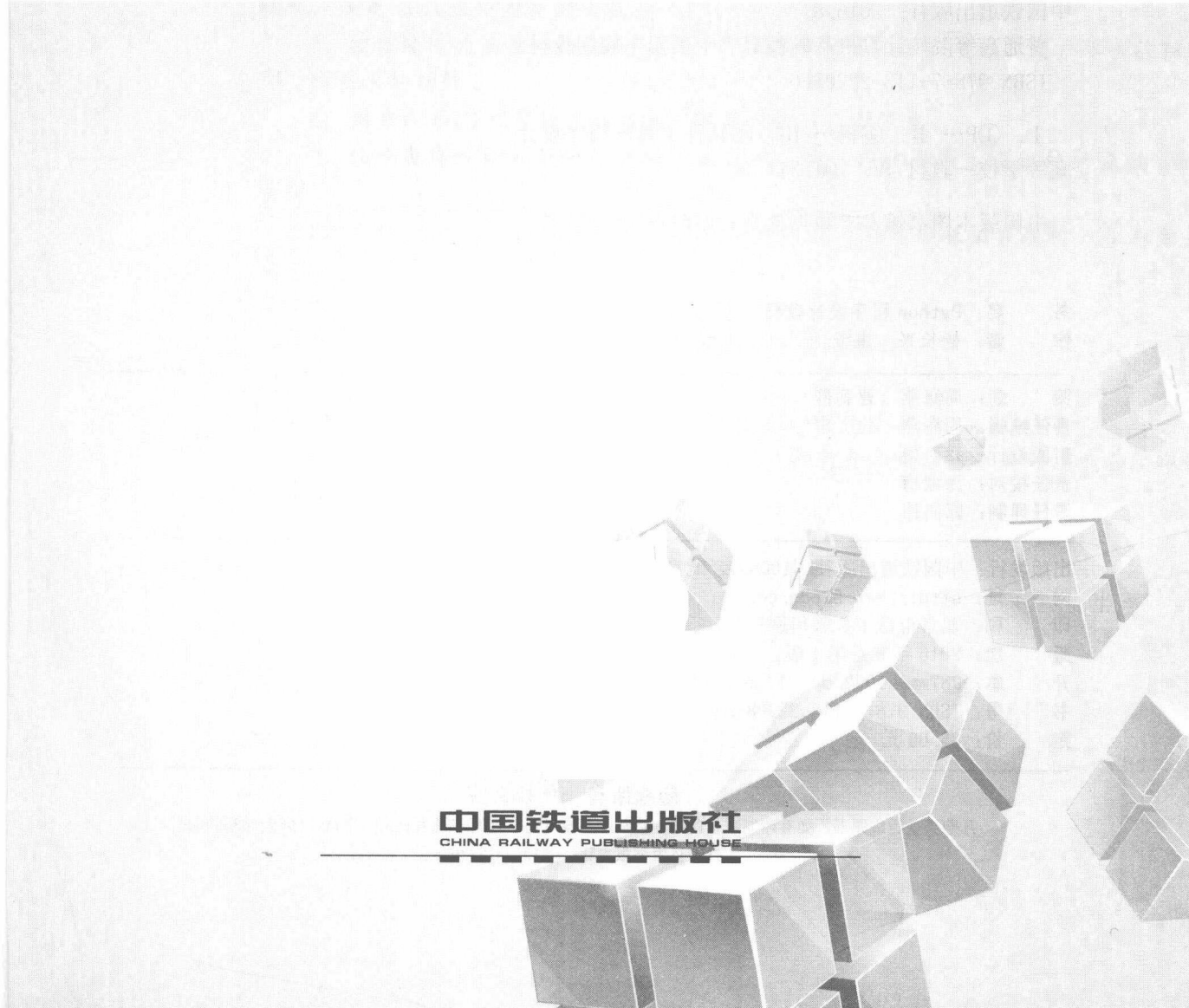
杨长兴　主编

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

普通高等院校计算机基础教育“十三五”规划教材

Python 程序设计教程

杨长兴　主编



中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

内 容 简 介

本书以零基础为起点介绍 Python 程序设计方法。各章内容由浅入深、相互衔接、前后呼应、循序渐进。为了提高读者对程序设计思想方法的理解，本书将程序设计语言模型与人类自然语言模型进行了比较，使读者对程序设计语言模型及其内容的理解有了完整的参照对象。全书各章节选用大量程序设计语言经典实例来讲解基本概念和程序设计方法，同时配有大量习题供读者练习。

本书共 12 章，主要内容包括程序设计语言绪论、对象与类型、运算符与表达式、程序控制结构、函数、列表与元组、字典与集合、文件与目录、模块、错误与异常、面向对象编程、图形用户界面编程。

本书语言表达严谨，文字流畅，内容通俗易懂、重点突出、实例丰富，适合作为高等院校各专业程序设计语言课程的教材，还可作为全国计算机二级考试的参考用书。

图书在版编目（CIP）数据

Python 程序设计教程 / 杨长兴主编. — 北京 : 中国铁道出版社, 2016.8
普通高等院校计算机基础教育“十三五”规划教材
ISBN 978-7-113-22208-6

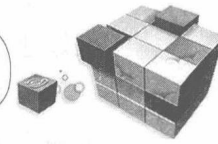
Ⅰ. ①P… Ⅱ. ①杨… Ⅲ. ①软件工具－程序设计－高等学校－教材 Ⅳ. ①TP311.56

中国版本图书馆 CIP 数据核字(2016)第 189620 号

书　　名：Python 程序设计教程
作　　者：杨长兴　主编

策　　划：周海燕　曹莉群　　读者热线：（010）63550836
责任编辑：周海燕　包　宁
封面设计：乔　楚
责任校对：汤淑梅
责任印制：郭向伟

出版发行：中国铁道出版社（100054，北京市西城区右安门西街 8 号）
网　　址：http://www.51eds.com
印　　刷：北京市昌平百善印刷厂
版　　次：2016 年 8 月第 1 版　　2016 年 8 月第 1 次印刷
开　　本：787mm×1092mm　1/16　　印张：12.75　　字数：280 千
书　　号：ISBN 978-7-113-22208-6
定　　价：35.00 元



前　言

目前，在教育部高等学校计算机基础课程教学指导委员会的指导下，计算机基础课程教学改革工作在不断推进深入。程序设计语言课程是大学生必须掌握的计算机基础课程，大学生们通过这门课程的学习，应该掌握程序设计的基本方法，具备用程序解决问题的能力。如何选择某种程序设计语言作为高等学校大学生程序设计课程的语言环境，是各校计算机基础教育工作者研究的课题之一。Python 语言作为一门开源语言，已被许多学校引入教学过程。它是面向对象和过程的程序设计语言，具有无界整数数据类型及丰富的数据结构、可移植性强、语言简洁、程序可读性强等特点。根据实际教学经验，在程序设计课程教学改革研究时，我们选用 Python 语言作为程序设计课程的语言环境。对本书内容的选择，我们力求面向读者，以程序设计零基础为起点，全面介绍了包括面向过程和面向对象的 Python 程序设计方法。让读者首先接受面向对象的程序设计的思想方法，并理解面向对象的程序设计是需要以面向过程的程序设计方法作为基础的。

全书共分为 12 章，第 1 章介绍程序设计语言入门与 Python 语言开发环境；第 2 章介绍对象与类型；第 3 章介绍运算符与表达式；第 4 章介绍程序控制结构；第 5 章介绍函数；第 6 章介绍列表与元组；第 7 章介绍字典与集合；第 8 章介绍文件与目录；第 9 章介绍模块；第 10 章介绍错误与异常；第 11 章介绍面向对象编程；第 12 章介绍图形用户界面编程。

本书编者长期从事程序设计课程的教学工作，并利用各种语言开发工具开发了许多软件项目，具有丰富的教学经验和较强的科学研究能力。编者本着加强基础、注重实践、强调思想方法的教学、突出应用能力和创新能力培养的原则，力求使本书有较强的可读性、适用性和先进性。我们的教学理念是：教学是教思想、教方法，真正做到“授人以鱼，不如授人以渔”。为了加强读者对程序设计思想方法的理解，本书将程序设计语言模型与人类自然语言模型相比较，让读者对程序设计语言模型及其内容的理解有了完整的参照对象。为了提高读者的编程技巧，书中选用了大量的经典例题，这些例题与相应章节的内容是完全吻合的，例题还备有多种可能的解答，以期拓展读者的解题思路。为了便于读者自学，全书在内容组织、编排上注重由浅入深、循序渐进。因此，本书适合作为高等院校各专业程序设计课程的教材，也可作为广大计算机爱好者的自学参考用书。教师选用本书作为大学生程序设计课程的教材时，可根据实际教学课时数调整或取舍内容。

本书所给出的程序示例均在 Python 3.3 环境下进行了调试和运行。为了帮助读者更好地学习 Python，编者在每章后还编写了大量的习题供读者练习。

本书由杨长兴主编，并负责全书的总体策划、统稿和定稿工作。肖峰教授协助主编完成统稿、定稿工作。各章参加编写人员：中南大学杨长兴（第 1 章）；大连医科大学肖峰、河北医科大学李连捷（第 2、3 章）；中山大学刘燕（第 4 章）；北京大学郭永青（第 5 章）；首都医科大学夏翊（第 6 章）；中南大学田琪、李利明、李小兰（第 7、8 章）；复旦大学韩绛青、武警后勤学院孙纳新（第 9、10 章）；中南大学周春艳、刘卫国、朱从旭（第 11 章）；肖峰、中南大学周肆清、罗芳、奎晓燕（第 12 章）。

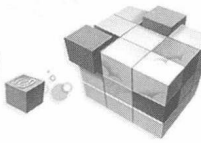
本书的编写得到了清华大学谭浩强教授、吴文虎教授的指导与帮助，在此一并表示衷心感谢。在本书的编写过程中，中南大学邵自然、吕格莉、裘嵘、杨莉军、曹丹等老师参与了大纲的讨论，本书吸收了他们许多宝贵的意见和建议，在此一并表示衷心感谢。编者在编写本书的过程中参考了大量的文献资料，在此也向这些文献资料的作者表示衷心感谢。

由于编者水平所限，书中疏漏及不妥之处在所难免，敬请读者不吝赐教。

编　者

2016 年 6 月

目　录



第1章 程序设计语言绪论

计算机程序设计语言通常是指高级程序设计语言，包括本书将要介绍的 Python 语言，之所以说它是高级程序设计语言，是因为它是按照人类的理解方式设计，人类可以编写、阅读理解这种程序。理解计算机程序的主体不仅仅是人类，还有一类主体是计算机，只有计算机理解并执行程序的功能，才能解决程序所需要完成的功能。计算机必须通过某种转换将计算机程序转换为计算机可直接识别的代码才能执行程序的功能，完成工作任务。其实，计算机程序设计语言类似于人类的自然语言，二者之间有着相似甚至相同的语言模型。

通过本章的学习，掌握计算机程序设计语言模型、程序编译与解释的概念；掌握 Python 程序设计语言开发环境和应用程序开发过程；编写程序的基本步骤、算法与流程图。

计算机程序是用于解决实际问题的。学习 Python 程序设计语言的目的，就是要学会使用 Python 语言编写出适合自己实际需要的程序。程序包括数据和施加于数据上的操作两方面的内容。数据是程序处理的对象，操作步骤反映了程序的功能细节，全部操作步骤的集合则是程序表达的功能。不同类型的数据有不同的操作方式和取值范围，程序设计需要考虑数据的表示以及操作步骤（即算法）。Python 语言具有丰富的数据类型和相关运算，这是它有别于其他程序设计语言的最大特点之一，它有其他程序设计语言所不具备的众多的数据类型，特别是其整数类型，对于精确运算（上百位或更多位数值）是其独有的。本章首先介绍程序设计的基本概念、Python 程序结构、Python 程序的执行方式以及开发环境配置。

1.1 计算机程序设计语言概述

计算机程序设计语言是人类在计算机上解决实际问题的一种编码规则工具。当一个求解问题能够用数学模型表达时，人们会考虑用某种程序设计语言将该问题的数学模型表示成计算机可以接受的程序形式，再由计算机自动处理这个程序，生成人们所需要的结果。

程序设计语言随着计算机科学的发展而发展，它由最早的机器语言形式逐步发展成为现在的接近人类自然语言的形式。

20 世纪 50 年代的程序设计是使用机器语言或汇编语言编写的，用这样的程序设计语言设计的程序相当烦琐、复杂，不同机器使用的机器语言或汇编语言几乎完全不

同。能够使用这类语言编写程序的人群极其有限，也就限制了这类计算机程序设计语言的普及和推广，必然影响计算机的普及和应用。

20 世纪 50 年代中期研制出来的 FORTRAN 语言是计算机程序设计语言历史上的第一个高级程序设计语言。它在数值计算领域首次将程序设计语言以接近人类自然语言的形式呈现在人们面前，它引入了许多目前仍在使用的程序设计概念，如变量、数组、分支、循环等。20 世纪 50 年代后期研制的 ALGOL 语言进一步发展了高级程序设计语言，提出了块结构的程序设计概念。即一个问题的求解程序可以由多个程序块组成，块与块之间相对独立，不同块内的变量可以同名，但互不影响。

到了 20 世纪 60 年代后期，人们设计出来的程序越来越庞大，随之而来的问题是程序越庞大，程序的可靠性越差，错误越多，并且难以维护。程序设计人员难以控制程序的运行，这就是当时的“软件危机”问题。为了解决“软件危机”问题，荷兰科学家 E.W.Dijkstra 在 1969 年首次提出了结构化程序设计的概念，这种思想强调从程序结构和风格上研究程序设计方法。后来，瑞士科学家 Niklans Wirth 的“算法+数据结构=程序”思想进一步发展了结构化程序设计方法，将一个大型的程序分解成多个相互独立的部分（称为模块）。模块化能够有效分解大型、复杂的问题，同时每个模块相互独立，提高了程序的维护效率。这就是面向过程的结构化程序设计思想。所谓面向过程的结构化程序设计思想是人们在求解问题时，不仅要提出求解的问题，还要精确地给出求解问题的过程（将问题的求解过程分解成多个、多级相互独立的小模块）。20 世纪 70 年代初面世的 C 语言就是典型的、面向过程的结构化程序设计语言。

面向过程的结构化程序设计是从求解问题的功能入手，按照工程的标准和严格的规范将求解问题分解为若干功能模块，求解问题是实现模块功能的函数和过程的集合。由于用户的需求和硬件、软件技术的不断发展变化，按照功能划分将求解问题分解出来的模块必然是易变和不稳定的。这样开发出来的模块可重用性不高。20 世纪 80 年代提出的面向对象的程序设计方法即是为了解决面向过程的结构化程序设计所不能解决的代码重用问题。面向对象的程序设计方法是从所处理的数据入手，以数据为中心而不是以求解问题的功能为中心来描述求解问题。它把编程问题视为一个数据集合，数据相对于功能而言，具有更好的稳定性。这就是“对象+对象+……=程序”的理论。面向对象程序设计与面向过程结构化程序设计相比，最大的区别就在于：前者关心的是所要处理的数据，而后者关心的是求解问题的功能。面向对象程序设计方法很好地解决了“软件危机”问题。

面向对象程序设计语言有两类：一类是完全面向对象的语言，另一类是兼顾面向过程和面向对象的混合式语言。

面向对象程序设计语言其实是以面向过程结构化程序设计语言为基础的。面向对象程序设计语言在构建应用程序框架、输入/输出界面等方面由系统做了大量的基础工作，应用程序设计人员只需要关注应用问题的解决；而面向过程结构化程序设计语言程序人员需要解决应用程序框架、输入/输出界面、应用问题的解决过程，并且面向过程结构化程序设计语言的程序代码与数据相互独立。

无论是面向对象程序设计语言还是面向过程结构化程序设计语言，从解决应用问

题的角度来说，它们都与人类自然语言有着极其相似的语言模型，从设计语言、使用语言上，都有共同的语言模型。图 1-1 所示为程序设计语言模型图。

图 1-1 程序设计语言模型图

读者可以了解一下一般程序设计语言的模型。通过图 1-1 进行理解：学习某种程序设计语言，主要是学习（见图 1-1）根据词法规则用某种语言字符集中的字符构造单词；根据语法规则用单词构造语句；根据逻辑规则（任务内在的联系）用语句构成程序。读者可以根据图 1-1 自学其他程序设计语言。实际上，任何语言都遵从这种模式，包括自然语言（英语、汉语等）。对于自然语言，只是字符集中的字符多，构词规则复杂，语法规则更复杂，由若干语句组成的集合称为文章或文章段落而已。其实计算机程序设计语言就是从自然语言模型中简化出来的。理解了这个道理，对于学习程序设计语言是很有帮助的。

对于第一次学习某种计算机语言的读者来说，图 1-1 实际上是以学习人类自然语言为模型（参照对象），给出了学习某种计算机语言的模型。有了这个参照对象，读者自然知道从什么地方开始学习程序设计语言了，重点解决什么问题。

1.2 程序的编译与解释

程序开发人员编写的高级语言程序应该让他人读懂，更重要的是使计算机（硬件）理解和执行。执行的过程就是解决问题的过程。

高级语言程序按照执行方式分为静态语言和脚本语言。静态语言程序采用编译方法执行，脚本语言采用解释方法执行。

编译是将高级语言程序（称为源程序）通过编译程序（针对某种静态语言的系统程序）转换为目标代码（又称目标程序，还不是最终的计算机可执行的代码，代码的文件保存形式一般是.OBJ）的过程。执行编译的程序称为编译程序或编译器。为了让计算机直接执行（完成）程序的功能，还需通过连接程序将目标代码转换为执行代码（执行程序），执行代码的文件保存形式一般是.EXE。计算机直接执行的程序就是.EXE 文件。图 1-2 给出了程序的编译、连接、执行过程。

解释是将高级语言程序通过解释程序（针对某种脚本语言的系统程序）转换为可执行代码并同时逐条执行的过程。执行解释的程序称为解释器。图 1-3 给出了程序的解释过程。

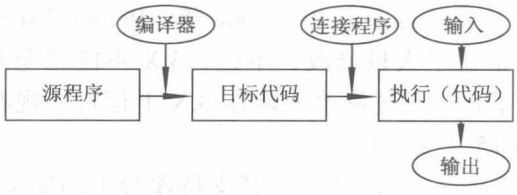


图 1-2 程序的编译、连接、执行过程

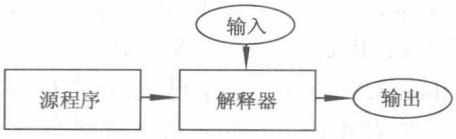


图 1-3 程序的解释过程

编译与解释的区别在于编译是一次性的工作，一旦程序被编译，不再需要编译程序和源程序代码了，而解释则在程序的每次执行过程中都需要解释程序和源程序代码。

编译过程是一次性的，因此，编译过程的执行速度并不重要，重要的是目标程序的质量，目标程序代码量与执行速度直接决定了后面生成的执行程序的代码量与执行速度，所以，目标程序的质量才是编译过程的关键。为此，现在的编译程序在不断地优化，目的是提高执行效率。而在解释程序中，因为优化技术会消耗运行时间，使得整个程序的执行速度会受到影响，不能过多地集成优化技术。解释执行方法尽管牺牲了一定的执行速度，但可以支持跨平台（硬件或操作系统）、对保留和维护源程序代码十分方便，适合非实时等运行场合。理论上说，编译后程序比解释后程序执行速度要快。

现在随着编译器和解释器工具的进步，解释器中也吸收了编译器的功能，在程序执行的过程中，解释器也会产生一个完整的目标代码，这种新型解释器会对现代脚本语言执行性能的提高起到重要作用。Python 语言就是一个典型的脚本语言，采用解释执行的方式，但它的解释器中包含了编译器的功能。

采用编译方法的好处：

（1）对于相同的源程序代码，编译所产生的目标程序代码执行速度快。

（2）编译所产生的目标程序代码可以脱离编译器独立运行。

采用解释方法的好处：

（1）程序调试执行时，程序纠错、维护方便、灵活。而编译后程序如果有错，需要修改程序后再次编译、连接。

（2）源程序虽然不能脱离解释器独立运行，但源程序代码可以在不同操作系统上运行，可移植性好，这是编译方法没有的特点。

1.3 Python 语言

1.3.1 Python 语言及其特点

Python 语言诞生于 1990 年左右，由荷兰人 Guido van Rossum 设计并领导开发。该语言命名为 Python 源于 Guido 的兴趣，当初只是为了自娱自乐尝试编写一种替代 ABC 这些编程语言的脚本语言，没想到受到大家的喜欢，一直发展至今，后来引入了对多平台的支持。2000 年，Python 2.0 的正式发布标志着 Python 语言正式进入了广泛应用的时代。直至今日，许多的标准库、应用程序都是基于 Python 2.X 系列解释器的，2.X 系列中的 2.7.6 版还有不少的使用者，而且在不断更新中。2008 年，Python 3.0 正式发布，Python 3.X 在语法层面和解释器内部做了大量修改，不过，3.X 不能完全兼容 2.X，因此，基于 2.X 的库函数及应用程序必须经过修改才能在 3.X 上运行。现在 Python 3.X 系列的版本已发展到 3.5.1 版（2015 年 12 月）。

笔者相信，Python 语言最终会有一个完美的 3.X 版，并且，其支持库会更加完美、丰富。本书中的实例均采用 3.3 版。

Python语言是一个脚本语言。将其作为大学生的第一门程序设计语言课程，在国内还处于尝试阶段，这样做的合理性、可行性还值得研究。大家会从不断开发Python语言程序的过程中，体会到Python语言的特点，找到其答案。

Python语言的特点：

（1）Python语言简洁，只有少量的语法约束。也许正是这种特点，让许多人容易上手，很快会找到解决问题的方法。语言简洁、语法约束少，编写程序时接近人类自然语言的形式。不会像其他语言那样，有一个小小的语法错误，程序就不能运行，让程序编写人员长时间纠结在语法排错上。

（2）Python语言通过强制缩进保证程序的可读性。这是通过语法规则来保证程序的可读性，其他程序设计语言没有这一条。实际上，程序具有良好的可读性是十分重要的，首先能保证程序能让别人或自己看懂、理解，其次是能够从某种程度上确保程序的正确编写。

（3）Python语言具有丰富的数据结构（类型）。Python语言在多数程序设计语言的基础上，增加了列表、字典、元组、集合等数据结构，同时，对数字类型数据在表达范围和表达方法上进行扩充、修改补充，从而使数字的计算不受所属类型的存储位数的限制，可以精确地计算出任意位数的数据。例如，可以用简单的表达：2**100，精确地计算其结果：1 267 650 600 228 229 401 496 703 205 376。

（4）Python语言具有可移植性。尽管Python语言是脚本语言，但它可以同时被编译和解释执行。Python语言的标准实现是由可移植的ANSI C编写的，可以在目前所有主流平台上编译和解释执行。除语言解释外，Python语言发行时自带的标准库和模块在实现上都尽可能地考虑到了跨平台的移植性。此外，Python语言的源程序自动编译成可移植的字节码，这些字节码在已安装兼容版本Python语言平台上运行的结果是相同的。

（5）Python语言支持面向过程，同时支持面向对象，支持灵活的编程模式。

（6）Python语言的使用与分发是完全免费的，与其他开源软件一样。任何人可以从Internet上免费获取Python语言的系统源代码，可以复制，可以将其嵌入某系统随产品一起发布，没有任何限制。

Python语言的特点何止这些。读者需要在后续章节中不断学习，从实例中品味Python语言的强大功能和特点。

1.3.2 第一个Python语言程序示例

先让读者看一个用Python语言书写的求圆的面积的通用程序。

例 计算圆的面积。程序代码如下：

```
# -*- coding: GB2312 -*-
# ex1-1.py 计算圆的面积

def area(r):
    s = 3.14159*r*r
    return s

print (area(10))
```

这个程序的第一行是声明程序中使用了中文代码，没有这样的声明时，程序中是不能使用中文代码的。第一、二行其实是程序的注释行。第四至第六行是定义一个根据圆的半径计算其面积的函数。第八行用一个 print()函数输出一个半径为 10 的圆的面积。

这个程序可用任何编辑器编辑，以.py 的扩展名保存。在 Python 语言的解释器下运行，在解释器的交互窗口中会输出结果 314.159。

1.3.3 Python 语言程序的书写规范

现在以上一小节的第一个程序实例为例子说明 Python 程序的书写规范问题。

（1）Python 程序源代码最大的特点是：用缩进表示程序代码的层次。如例 1.1 中的第四至第六行，第四行是函数的头，其下面两行是函数体，从层次结构上讲，函数体比函数头要低一个层次，所以第五、六行缩进。缩进用 4 个空格表示（这是最流行的 Python 程序源代码缩进方式），也可以用制表符表示，但不要将二者混用。这种以强制缩进方式描述程序的层次结构对阅读程序是有好处的。

（2）一行代码的长度不超过 80 字符。如果实际代码超过 80 字符，通常使用圆括号、方括号和花括号折叠长行，也可以使用反斜杠延续行。例如：

```
if width == 0 and height == 0 and \
color == 'red' and emphasis == 'strong' or \
highlight > 100:
```

（3）注释问题。注释以“#”和一个空格开始，行内注释是和语句在同一行的注释，行内注释应该谨慎使用，行内注释应该至少用两个空格和语句分开，它们应该以“#”和单个空格开始。

（4）空格问题。在书写赋值语句或表达式时，建议在赋值运算符（=）、比较（==, <, >, !=, <>, <=, >=, in, not in, is, is not）、布尔运算（and, or, not）等运算符两边各置一个空格。例如：

```
x = x*2 - 1
c = (a+b) * (a-b)
```

（5）空行问题。用两行空行分隔顶层函数和类的定义，类内方法、函数的定义用单个空行分隔。

（6）关于标识符的约定。标识符用于命名变量、函数、类名、模块名等对象。标识符可以包含字母、数字和下画线（_），但必须以非数字字符开始。虽然对标识符的定义有完备的词法规则，但在编程时，还要遵守一些约定。像 if、else、for 等这样的单词是保留字，不能再用作标识符；以单、双下画线开始或结束的标识符通常有特殊意义，一般不用作标识符。

1.4 配置 Python 语言的开发环境

本节以 Windows 操作系统为基础，介绍如何安装配置 Python 语言的开发环境。也就是下载、安装、启动运行 Python 语言的解释器。

首先，从 Python 网站（http://www.python.org/download/）下载 Python 语言的基本开发和运行环境程序。对于 Windows 操作系统环境，选择名为“python-3.3.3.amd64