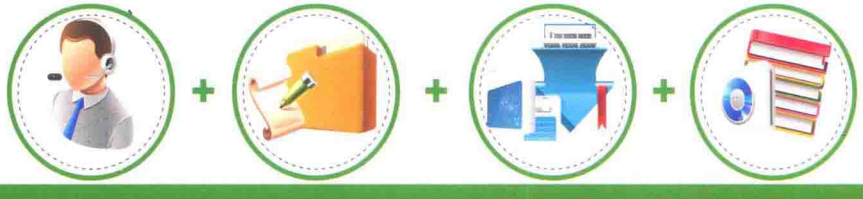
全国高等院校应用型创新规划教材 计算机系列

Python 程序设计实用教程

杨连贺 董禹龙 房 超 主 编
毕璐琪 梁润宇 杨 阳 彭进香 副主编



- 以基础理论—实用技术—实训为主线
- 按照教与学的实际需要取材谋篇
- 精心设置“小型案例实训”，旨在培养学生的实践能力
- 配备免费教学资源——教学课件、程序源代码、授课计划及习题答案等

清华大学出版社

全国高等院校应用型创新规划教材·计算机系列

Python 程序设计实用教程

杨连贺　董禹龙　房　超　主　编

毕璐琪　梁润宇　杨　阳　彭进香　副主编

清华大学出版社
北京

内 容 简 介

Python 是一门简单易学、功能强大的编程语言，它内建了高效的数据结构，能够用简单而又高效的方式编程。它优雅的语法和动态的类型，再结合它的解释性，使其成为在大多数平台下编写脚本或开发应用程序的理想语言。

本书系统而全面地介绍了 Python 语言的全部内容，既能为初学者夯实基础，又适合程序员提升技能。考虑到近几年数据挖掘技术和网络编程技术的发展，本书加入了 Python 语言在科学计算、网络编程、并发技术和数据可视化方面的内容。与一般的 Python 语言教材相比，本书增加了许多实际案例的应用，可以让读者更好地将 Python 基础知识应用到实际工作中。书中的每道例题，均以屏幕截图的方式原滋原味地给出运行结果，便于读者分析程序。

本书可作为高等院校各专业的 Python 语言教材，亦可作为软件开发人员的参考资料，还可作为读者自学 Python 语言的参考书。

图书在版编目(CIP)数据

Python 程序设计实用教程/杨连贺，董禹龙，房超主编. —北京：清华大学出版社，2018
(全国高等院校应用型创新规划教材 • 计算机系列)
ISBN 978-7-302-50047-6

Ⅰ. ①P… Ⅱ. ①杨… ②董… ③房… Ⅲ. ①软件工具—程序设计—高等学校—教材 Ⅳ. ①TP311.561

中国版本图书馆 CIP 数据核字(2018)第 086665 号

责任编辑： 汤涌涛
封面设计： 杨玉兰
责任校对： 宋延清
责任印制： 丛怀宇
出版发行： 清华大学出版社
网　　址：http://www.tup.com.cn, http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969, c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015, zhiliang@tup.tsinghua.edu.cn
课件下载：http://www.tup.com.cn, 010-62791865
印 装 者： 北京国马印刷厂
经　　销： 全国新华书店
开　　本： 185mm×260mm　　**印　张：** 23.5　　**字　数：** 572 千字
版　　次： 2018 年 6 月第 1 版　　**印　次：** 2018 年 6 月第 1 次印刷
印　　数： 1～2500
定　　价： 56.00 元

产品编号：074820-01

前　言

根据 TIOBE 网站的最新排名，Python 已超越 C#，与 Java、C、C++一起，成为全球前四大流行语言。IEEE 发布的 2017 年编程语言排行榜则将 Python 排在榜首。

Python 也是美国大学选用最多的语言，著名的哈佛大学、麻省理工学院、加州大学伯克利分校、卡耐基·梅隆大学等，已将 Python 语言作为计算机专业和非计算机专业的入门语言。Python 崇尚简、短、精、小，其应用几乎无限制，各方面地位超然。Python 在软件质量控制、提升开发效率/可移植性、组件集成、丰富的库支持等方面，均处于先进地位。更重要的是，Python 简单易学、免费开源、可移植、可扩展、可嵌入。此外，Python 还支持面向对象，而且它的面向对象甚至比 Java 和 C#.NET 更彻底。

Python 是高“性价比”的语言。它合理地结合了高性能与低成本(代码量小、维护成本低、编程效率高)的特色，致力于用最简洁、最简短的代码完成任务。

完成同样的业务逻辑时，在其他编程语言中可能需要编写大量的代码，而在 Python 中只需要调用内建函数或内建对象的方法即可实现，甚至可以直接调用第三方扩展库来完成。一般情况下，Python 的代码量仅仅是 Java 的 1/5，足见 Python 编程的高效。

Python 是应用“无限制”的语言。它被广泛应用于后端开发、游戏开发、网站开发、科学计算、大数据分析、云计算、图形开发等领域。美国中央情报局 CIA 网站、世界上最大的视频网站 YouTube、国内最大的问答社区“知乎”等，都是用 Python 开发的，搜狐、金山、腾讯、盛大、网易、百度、阿里、淘宝、土豆、新浪、果壳等著名的 IT 公司都在使用 Python 完成各种各样的任务。

Python 是一种代表“简单主义”思想的语言。它的设计哲学是优雅、明确、简单。阅读一个良好的 Python 程序，感觉就像是在阅读英语，尽管这个英语的要求非常严格！Python 的这种伪代码本质，是它最大的优点之一。

Python 是“高层次”的语言。它内建优异的数据结构，很容易表达各种常见的数据结构，不再需要定义指针、分配内存，编程也简单了许多，也无须考虑程序对内存的使用等底层细节，把许多机器层面上的细节隐藏起来，凸显出逻辑层面的编程思考。

Python 是免费、开源、跨平台的高级动态编程语言。它支持命令式编程、函数式编程，全面支持面向对象编程；它语法简洁、清晰，拥有功能丰富而强大的标准库和大量的第三方扩展库；它使用户能够专注于解决问题，而不是去搞明白语言本身，这是它开发效率高的根本原因。

由此可见，用“出类拔萃”来形容 Python 并不为过。Python 以如此众多的优势，吸引着无数程序员投身于其中。网上的一句流行语颇耐人寻味：“人生苦短，我用 Python”。

在国外，“Python 热”正在逐步升温，涉及方方面面的领域；在国内，越来越多的大学已将 Python 列入本科生的必修课程或选修课程，越来越多的 IT 企业将开发语言瞄向了 Python。可以预见的是，国内的“Python 热”即将掀起，本书的出版迎合了这一趋势。

本书的内容组织说明如下。

为了拓展应用范围，充分利用现有资源，对于 Python 程序员而言，熟练运用第三方扩展库是非常重要的。使用成熟的扩展库，可以帮助我们快速地实现业务逻辑，达到事半功倍的效果。但是，第三方扩展库的理解和运用，无疑要建立在对 Python 基础知识和基本数据结构熟练掌握的基础上。因此，本书兼顾“基础”与“应用”两个方面，前 6 章把重点放在基础上，通过大量的经典例题，讲解 Python 语言的核心内容；后 6 章则把重点放在应用上，通过大量的案例，介绍 Python 在实际开发中的应用。关于不同应用领域的第三方扩展库，读者可以参考附录 B，并结合自己的专业领域查阅相关文档。

本书共分 12 章，主要内容组织如下。

第 1 章：Python 程序设计入门。介绍什么是 Python，学习 Python 的原因，Python 的发展历史，多种平台下 Python 环境的搭建，使用集成开发环境 IDLE 来帮助学习 Python，Python 常用的开发工具，最后给出本书的第一个 Python 程序。

第 2 章：Python 语言基础。讲解 Python 的语法和句法，Python 的数据类型，Python 的常量与变量，Python 的运算符与优先级，Python 的数值类型，Python 的字符串类型，Python 的高级数据类型(列表、元组、字典、集合)，最后介绍正则表达式及其应用。

第 3 章：Python 流程控制。讲解 if 语句和 for 语句的基本格式、执行规则、嵌套用法，range()函数在循环中的使用方法，while 语句的基本格式、执行规则、嵌套用法，最后介绍 break、continue、pass 等关键字在循环中的使用方法。

第 4 章：函数模块。讲解 Python 代码编写规范和风格，函数的定义与调用，函数参数的传递，Python 变量作用域，函数与递归，迭代器与生成器，Python 自定义模块，输入输出语句的基本格式及执行规则，匿名函数的定义与使用。

第 5 章：文件与异常处理。介绍文件和文件对象，讲解基于 os 模块的文件操作方法，基于 shutil 模块的文件操作方法，文本文件、CSV 文件、Excel 文件的基本操作，HTML、XML 文档的基本操作，最后介绍 Python 的异常处理机制及 Python 程序的调试方法。

第 6 章：面向对象编程。介绍面向对象技术，讲解类与对象的定义和使用，类的属性与方法，类的作用域与命名空间，类的单继承和多继承，最后以数个典型实例讲解面向对象程序设计的应用。

第 7 章：数据库编程。讲解数据库技术基础，SQLite 和 MySQL 数据库的数据类型、基本操作，使用 Python 操作 SQLite 和 MySQL 数据库的方法。

第 8 章：Web 开发。讲解 Web 应用的工作方式，MVC 设计模式，CGI 通用网关接口，使用模板快速生成 Web 页面。

第 9 章：使用 Python 进行数据分析。讲解使用 Python 进行数据挖掘的原因，介绍 NumPy 库、SciPy 库、Matplotlib 库和 Pandas 库，最后通过数理统计中的数据离散度分析和数据挖掘中的离群点分析等典型案例，介绍 Python 在数据可视化方面的应用。

第 10 章：GUI 编程和用户界面。讲解 GUI 界面的概念，Tkinter 模块及其各种组件，网格布局管理器，最后介绍 GUI 编程。

第 11 章：多进程与多线程。介绍多进程与多线程的概念，讲解多进程与多线程的区别，进程间通信技术，进程池，最后介绍 thread 锁。

第 12 章：网络编程。讲解计算机网络基础知识，Socket 通信技术，urllib 库及其使用，端口扫描器，最后以一个简单的网络爬虫为例，对前几章的知识进行综合应用。

本书最大的特点是内容紧凑、案例丰富、学以致用；程序输出原滋原味，既有正确输出的结果，又有错误输出的提示，让读者既能从“正”的方面学到经验，又能从“负”的方面吸取教训，使经验与教训兼而得之。全书总体内容按照先基础、后应用的顺序安排。前 6 章为基础篇，其内容循序渐进；后 6 章为应用篇，其内容自成体系；每个知识点按照先讲解知识、后给出案例的顺序编写；每个软件都配有安装过程截图，每道例题都配有运行结果截图，一目了然。

本书作者具有近 30 年的程序设计教学经验，讲授过多门编程语言课程，并编写过大量的应用程序，青年时期曾参加过市级讲课大赛并取得优异成绩，特别是在美国访学期间，用 Python 语言开发过较大规模的软件。在内容的组织和安排上，本书结合了作者多年教学与科研中积累的经验，并巧妙地将其糅合到相应的章节中。

本书以目前流行的 Python 3 为基础，适当兼顾 Python 2.x，既讲解 Python 的基础知识，又适当介绍 Python 在各个方面的应用，因而，可以满足不同层次读者的需要。

本书可以作为高等院校计算机或非计算机专业程序设计语言公共课或选修课教材，基础教学建议选取前 6 章内容，推荐 36 学时；“基础+应用”教学建议按“6+n”方式选取教学内容，后面 6 章可根据专业需要择其一二，或全部选用，推荐 42~64 学时。建议采用边讲边练的教学模式。本书可以作为具有一定 Python 基础的读者进一步学习的资料，可供参加各类计算机考试的人员学习和参考，也可以作为从事数据分析、网络运维、数据库开发、Web 开发、界面设计、软件开发等工作的工程师的参考资料。对于打算利用业余时间快乐地学习一门编程语言并编写一些小程序来自我娱乐的读者，本书是首选的学习资料。本书亦适合对编程有着浓厚兴趣的中小学生作为课外阅读资料。

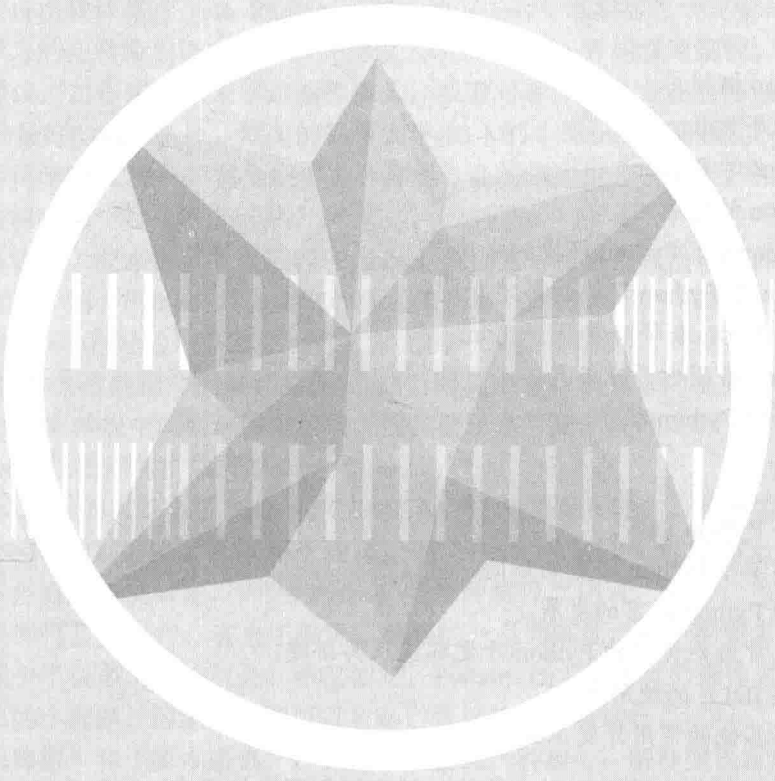
本书由天津工业大学杨连贺、董禹龙、房超主编，该校毕璐琪、梁润宇及天津市电子计算机研究所杨阳、湖南应用技术学院彭进香为副主编。限于作者的经验和水平，书中的错误与不足之处在所难免，希望得到专家和读者的批评指正。

本书编写过程中，天津工业大学计算机科学与软件学院硕士研究生张海潮和焦翠姣在程序调试方面做了很多工作，在此一并向她们表示衷心的感谢。

作　者

2018 年 5 月于天津工业大学

目　录



第 1 章

Python 程序设计入门

本章要点

(1) Python 概述。
(2) 什么是 Python?
(3) 为什么学 Python?
(4) Python 的发展。
(5) Windows 下 Python 环境的搭建。
(6) Linux 下 Python 环境的搭建。
(7) 使用 IDLE 来帮助学习 Python。
(8) Python 常用的开发工具。
(9) 第一个 Python 程序。

学习目标

(1) 了解 Python 语言。
(2) 理解学习 Python 语言的目的。
(3) 了解 Python 语言的发展。
(4) 熟悉常用平台下的 Python 开发环境及其搭建。
(5) 掌握 IDLE 的使用方法。
(6) 了解其他的常用开发工具。

本章向读者引入在国内方兴未艾的一门高级程序设计语言——Python。常用的编程语言达数十种之多，其功能各有千秋，应用领域也大相径庭。Python 之所以能够从众多的高级语言中脱颖而出，是因为 Python 是一种代表简单主义思想的语言，但集功能广泛与强大于一身。同样的功能，只用很少的代码就可以实现，编写的程序清晰易懂，优雅美观。用“高效开发+简单易学”来形容 Python 是恰如其分的。

1.1 Python 概述

Python 是一种简单易学、功能强大的编程语言，它继承了传统编译语言的强大性和通用性，具有高层次的数据结构，支持面向对象的编程方法。

Python 优雅的语法、动态的类型，连同它天然的解释性，使其成为在大多数平台下进行许多领域快速应用开发的理想语言。

1.1.1 什么是 Python

Python 是一门跨平台的、开源的、免费的、解释型高级动态编程语言，是由荷兰人 Guido van Rossum 在 1989 年始创的，其名字来源于一个喜剧。Python 的第一个公开发行版是在 1991 年初发布的，历经 20 余年的发展，目前最高版本为 3.6。根据 TIOBE 的最新排名，Python 已超越 C#，与 Java、C、C++一起，成为全球前四大最流行的语言。

也许 Guido van Rossum 最初并没有想到今天 Python 会在各种行业中获得如此广泛的

使用。著名的自由软件作者 Eric Raymond 在“如何成为一名黑客”一文中，将 Python 列为黑客应当学习的四种编程语言之一，并建议人们从 Python 开始学习编程。这的确是一个非常中肯的建议，对于那些从未学过编程的人，或者对非计算机专业的编程学习者而言，Python 不失为最好的选择之一。欧美的很多大学(如 MIT、Stanford 等)都以 Python 作为入门语言，之后再学习 C/C++，甚至很多非计算机专业的学生也开设此课。相信在不久的将来，国内高校将兴起一股“Python 热”。

有些人喜欢用“胶水语言”来形容 Python，是因为它可以很轻松地把许多其他语言编写的模块结合在一起。Python 具有丰富和强大的基本类库，能够把用其他语言制作的各种模块(尤其是 C/C++)很轻松地联结在一起。常见的一种应用情形是，使用 Python 快速生成程序的原型(有时甚至是程序的最终界面)，然后对其中有特别要求的部分，用更合适的语言改写，比如 3D 游戏中的图形渲染模块，性能要求非常高，就可以用 C/C++重写，而后封装为 Python 可以调用的扩展类库。

很多初学 Java 的人都会被 Java 的 CLASSPATH 搞得晕头转向，花上大半天才搞清楚原来是 CLASSPATH 搞错了，以至于连自己的“Hello World!”程序都无法运行。而用 Python，就不会出现此类问题，因为 Python 是一种解释型语言，同时，也是一种脚本语言，写好代码即可直接运行，省去了编译、链接的一系列麻烦，对于需要多动手实践的初学者而言，减少了很多出错的机会。不仅如此，Python 还支持交互的操作方式，如果只是运行一段简单的小程序，连编辑器都可以省略，直接输入即可运行。

谈及“解释型”和“脚本语言”，人们常常会有一种担心：解释型语言通常很慢。

的确，从运行速度来讲，解释型语言通常会慢一些，但 Python 的速度却比人们想象的快很多。虽然 Python 是一种解释型语言，但实际上也可以编译(就像编译 Java 程序一样)，即把 Python 程序编译为一种特殊的 ByteCode。程序运行时，执行的实际上是 ByteCode，省去了对程序文本的分析解释，速度自然得到了显著的提升。在用 Java 编程时，人们往往崇尚一种 Pure Java 方式，除了虚拟机外，一切代码都用 Java 编写，无论是基本的数据结构还是图形用户界面，而 Pure Java 的 Swing 却成为无数 Java 应用开发者的噩梦。

Python 与之不同，它崇尚的是实用，它的整体环境是用 C 编写的，其中的很多基本功能和扩展模块均用 C/C++编写，执行这一部分代码时，其速度就是 C 的速度。用 Python 编写的普通桌面应用程序，其启动、运行的速度与用 C 书写的程序相差无几。除此之外，通过一些第三方软件包，用 Python 编写的源代码还可以以类似于 JIT 的方式运行。此举可大大提高 Python 代码的运行速度，针对不同类型的代码，运行速度将有 2~100 倍的提升。

Python 是一种结构清晰的编程语言，使用缩进的方式来表示程序的嵌套关系，可谓是一种创举。此举把过去软性的编程风格升级为硬性的语法规定，再也不需要在不同的风格之间进行选择。与 Perl 语言不同，Python 中没有各种隐晦的缩写，也不需要强记各种符号的含义。用 Python 书写的程序很容易读懂，这是不少人的共识。虽然 Python 是一种面向对象的编程语言，但它的面向对象却不像 C++那样强调概念，而是更注重实用。说到底，Python 不是为了体现对概念的完整支持而把语言搞得很复杂，而是用最简单的方法，让编程者能够享受到面向对象带来的好处，这正是 Python 能够吸引众多支持者的原因之一。

Python 是一种功能丰富的语言，它拥有一个强大的基本类库，同时拥有数量众多的第三方扩展库，这使得 Python 程序员无须去羡慕 Java 的 JDK。Python 为程序员提供的基本

功能十分丰富，使得人们写程序时无需一切从底层做起。

C 语言用来编写操作系统等贴近硬件的系统软件，所以，C 语言适合开发那些追求运行速度、充分发挥硬件性能的程序，而 Python 是用来编写应用程序的高级编程语言。许多大型网站就是用 Python 开发的，例如 YouTube、Instagram，还有国内的豆瓣等。很多大型的组织机构，包括 Google、Yahoo 等，甚至 NASA(美国航空航天局)，都大量地使用 Python。除此之外，还有搜狐、金山、腾讯、盛大、网易、百度、阿里、淘宝、土豆、新浪、果壳等公司，都在使用 Python 完成各种各样的任务。

Guido van Rossum 给 Python 的定位是“优雅、明确、简单”，所以 Python 程序看上去总是简单易懂。初学者学 Python，不但入门容易，而且将来深入下去，可以编写那些非常非常复杂的程序。

总地来说，Python 的哲学就是简单优雅，尽量写容易看明白的代码和少的代码。

1.1.2 为什么学 Python

之所以越来越多的人选择学习 Python，是因为 Python 有着与众不同的特性。

(1) 易用性与高速度的完美结合。

在为数众多的高级语言中，在易用性和速度上结合得最完美的非 Python 莫属。通过丧失一点点经常可以忽略不计的运行速度而获得更高的编程效率，这是众多的编程者选择 Python 的原因。

(2) 自动的垃圾回收机制。

Python 的自动垃圾回收机制是高级编程语言的一种基本特性，用支持这一功能的语言编程，程序员通常无须关心内存泄漏问题，而用 C/C++书写程序时，这却是最重要的需要认真考虑而又很容易出错的问题之一。

(3) 内建的优异数据结构。

数据结构是程序的重要组成部分。在用 C 编程时，对于链表、树、图这些数据结构，需要用指针仔细表达，而这些问题在 Python 中简单了很多。在 Python 中，最基本的数据结构是列表、元组和字典，用它们表达各种常见的数据结构可谓轻而易举。由于不再需要定义指针、分配内存，编程也简单了许多。

(4) 简易的 CORBA 绑定。

公用对象请求代理体系结构(Common Object Request Broker Architecture，CORBA)是一种高级的软件体系结构，它是与语言无关、与平台无关的。

C++、Java 等语言都有 CORBA 绑定，但与它们相比，Python 的 CORBA 绑定却容易了很多，因为在程序员看来，一个 CORBA 的类和 Python 的类用起来以及实现起来并没有什么差别。没有复杂体系结构的困扰，用 Python 编写 CORBA 程序也变得容易了。

(5) 成熟的跨平台技术。

随着 Linux 的不断成熟，越来越多的人转到 Linux 平台下工作，软件的开发者自然就希望自己编写的软件可以在所有平台下运行。Java 的“一次编写处处运行”口号曾使它成为跨平台开发工具的典范，但其运行速度却不被人们看好。

Python 不仅支持各种 Linux/Unix 系统，还支持 Windows，甚至在 Palm 上都可以运行 Python 程序。Python 不仅支持老一些的 TK，还支持新的 GTK+、QT 以及 wxWidget，而这些都可以在多个平台下工作。通过它们，程序员就可以编写出漂亮的跨平台图形用户界面(Graphical User Interface，GUI)程序。

(6) 高可扩展性与“乘坐快车”。

如果希望一段代码可以很快地执行，或者不希望公开一个算法，则可以使用 C/C++编写这段程序，然后在 Python 中调用，从而实现对 Python 程序的扩展。换言之，程序员可以用 C/C++为 Python 编写各种各样的模块，这不仅可以让程序员以 Python 的方式使用系统的各种服务和用 C/C++编写的优秀函数库与类库，还可以大幅度提高 Python 程序的运行速度。用 C/C++编写 Python 的模块并不复杂，而且为了简化这一工作，人们还制作了不少工具，用来协助这一工作。正因如此，现在各种常用的函数库和类库都有 Python 语言的绑定，用 Python 可以做到的事情越来越多了。

Python 从一开始就特别关注可扩展性。Python 可以在多个层次上扩展。从高层上，可以直接引入.py 文件，在底层则可以引用 C 语言的库。Python 程序员可以快速地使用 Python 写.py 文件作为扩展模块。但当运行速度作为重要的考虑因素时，Python 程序员可以深入底层，写 C 程序，编译为.so 文件后引入到 Python 中使用，就如同人们乘坐快车一样。Python 恰似使用钢来构建房子，规定好大的框架之后，程序员可以在此框架下相当自由地扩展或更改。

(7) 隐藏细节，凸显逻辑。

Python 将许多机器层面上的细节隐藏，交给编译器处理，并凸显出逻辑层面的编程思考。Python 程序员可以花更多的时间用于思考程序的逻辑，而不是具体的实现细节。这一特征吸引了广大的程序员。

Python 有如此众多的优点，读者是否会觉得 Python 是一把万能钥匙呢？答案自然是否定的。这是因为，Python 虽然功能强大，但它却不是万能的。不同的语言有不同的应用范围，想找一把“万能钥匙”是不太可能的。C 和汇编语言适合编写系统软件，如果用它们来编写企业应用程序，恐怕很难得心应手。因此，聪明的程序员总是选用合适的工具去开发软件。如果要编写操作系统或驱动程序，Python 很显然是做不到的。要写软件，没有哪个工具是万能的，现在之所以有那么多的编程语言，就是因为不同的语言适合做不同的事情。因此，选择适合自己的语言才是最重要的。

常言道：“好钢要用在刀刃上。”要想用有限的时间完成尽量多的任务，就要把各种无关的问题抛弃，而 Python 恰恰提供了这种方法，这也是我们要学习 Python 的原因。

1.1.3　Python 的发展

1989 年，为了打发圣诞节假期，Guido 开始写 Python 语言的编译器。Python 这个名字，来自 Guido 所挚爱的电视剧 Monty Python's Flying Circus。他希望这个新的称作 Python 的语言能符合他的理想：创造一种介于 C 和 Shell 之间、功能全面、易学易用、可拓展的语言。

1991 年，第一个 Python 编译器诞生，它是用 C 语言实现的，并能够调用 C 语言的库

文件。Python 从一开始就具有了类、函数、异常处理，并且拥有包含表和字典在内的核心数据结构，以及以模块为基础的拓展系统。

1994 年 1 月，Python 1.0 推出，其中增加了 lambda、map、filter 和 reduce。

1999 年，Python 的 Web 框架之祖——Zope 1 发布。

2000 年 10 月 16 日，Python 2.0 问世，加入了内存回收机制，构成了现在 Python 语言框架的基础。

2004 年 11 月 30 日，Python 2.4 版出现；同年，目前最流行的 Web 框架 Django 诞生。

2006 年和 2008 年先后推出的 2.5 和 2.6 为过渡版本(2008 年已推出 3.0 版本)，2010 年推出的 2.7 版为 2.x 的最后一版。

2014 年 11 月，Python 2.7 将在 2020 年停止支持的消息被发布，并且不会再发布 2.8 版本，同时建议用户尽可能地迁移到 3.4+。

Python 最初发布时，在设计上有一些缺陷，比如 Unicode 标准晚于 Python 出现，所以一直以来，对 Unicode 的支持并不完全，而 ASCII 码支持的字符很有限。早期版本对中文的支持不好，Python 3 相对于 Python 早期的版本是一个较大的升级，Python 3 在设计的时候没有考虑向下兼容，所以很多早期版本的 Python 程序无法在 Python 3 下运行。为了照顾早期的版本，推出了过渡版本 2.6，该版基本上使用了 Python 2.x 的语法和库，同时考虑了向 Python 3.0 的迁移，允许使用部分 Python 3.0 的语法与函数。2010 年继续推出了兼容版本 2.7，大量 Python 3 的特性被反向迁移到了 Python 2.7。2.7 比 2.6 进步非常多，同时拥有大量的 Python 3 中的特性和库，并且照顾了原有的 Python 开发人群。

前面刚刚提到，Python 2.7 是 2.x 系列的最后一个版本，已经停止开发，不再增加新的功能，而且将于 2020 年终止支持，这意味着所有最新的标准库更新改进，只会在 3.x 版本里出现。Guido 决定清理 Python 2.x，并且不再兼容旧版本。

2.0 版到 3.0 版最大的一个改变，就是使用 Unicode 作为默认编码。Python 2.x 中直接写中文会报错，Python 3 中可以直接写中文了。从开源项目来看，支持 Python 3 的比例已经显著提高，知名的项目一般都支持 Python 2.7 和 Python 3+。

Python 3 比 Python 2 更规范、统一，去掉了不必要的关键字。Python 3.x 还在持续改进，截至本书定稿，Python 的最高版本是 2017 年 7 月 17 日推出的 3.6.2 版。我们推荐大家使用 Python 3.x。

1.2 Python 开发环境的搭建

Python 可在多种平台下运行，但考虑到目前应用较多的是 Windows 平台和 Linux 平台，故本节分别介绍这两种平台下 Python 开发环境的搭建。

1.2.1 Windows 下 Python 开发环境的搭建

(1) 下载 Python 安装包。

首先登录 Python 官方网站 http://www.python.org/download/，点击 Download 后，选择适合自己版本的安装包，并下载之。

(2) 安装 Python。

双击下载的安装包，一路单击 Next，最后单击 Finish。

(3) 配置环境变量。

Python 安装完毕后，还需要把 Python 的安装目录添加到系统的 PATH 环境变量中。

(4) 测试。

在 DOS 命令提示窗口中输入“python”，若显示图 1-1 所示的界面，则说明 Python 已经安装成功。

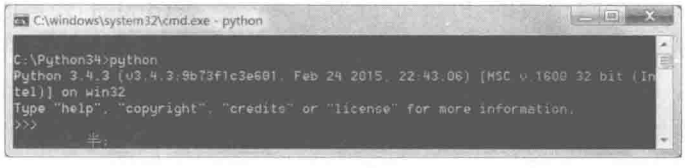


图 1-1　Python 安装成功的显示界面

(5) 演示 Python 命令。

按照很多资料介绍的，输入的第一个命令是“print ‘Hello world!’”，如图 1-2 所示。

结果显示“SyntaxError”(句法错误)。为什么会出错呢？原因在于，很多资料都是以 Python 2.x 为背景介绍的，而现在我们安装的是 3.x 版本，3.x 要求采用的写法是 print (‘Hello World!’)，即把 print 视为一个函数，而不是命令(考虑到习惯，本书仍称其为 print 命令)，因而需要使用括号，并把字符串放在引号中(单引号、双引号均可，但首尾必须相同)。改正后的运行结果如图 1-3 所示。

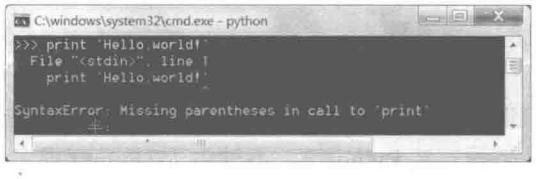


图 1-2　想要显示“Hello world!”

图 1-3　命令行上显示“Hello world!”

(6) 集成开发环境的使用。

经过以上测试可知，Python 环境已安装完毕，但如何开发软件呢，难道用这种命令行方式开发？当然不是，因为有众多的集成开发环境(Integration Development Environment，IDE)可供使用。

在 Windows 下安装 Python 的同时，也默认安装了其自带的集成开发环境 IDLE，其启动方法是：开始→所有程序→Python 3.4→IDLE，启动后的界面如图 1-4 所示。

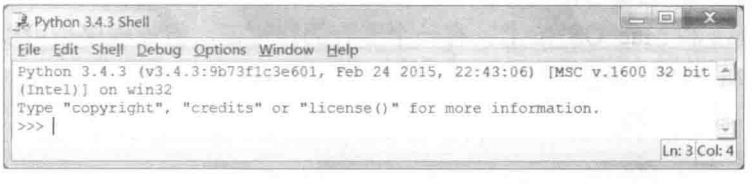


图 1-4　Python 的集成开发环境 IDLE 界面