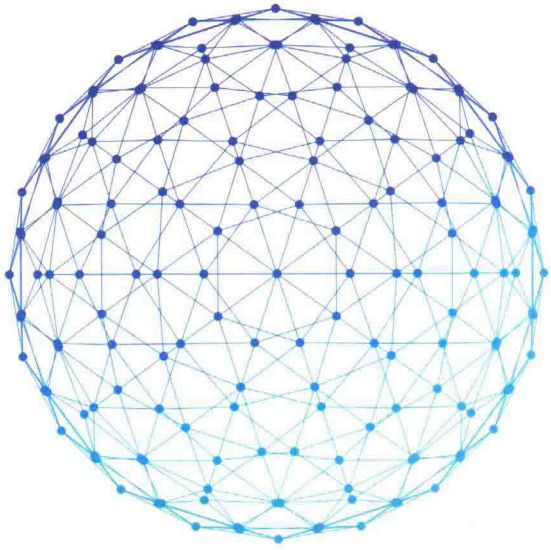
数据中国“百校工程”项目系列教材
数据科学与大数据技术专业系列规划教材

瑞翼教育

Python
编程基础与应用

韦德泉 许桂秋 ◉ 主编
方曙东 莫毅 曹红根 钱鸣 史春雷 张钦礼 胡楠 朱长水 ◉ 副主编



BIG DATA
Technology

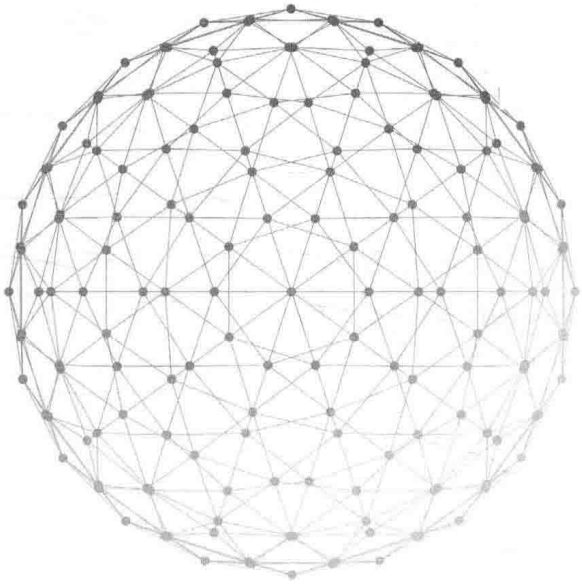
中国工信出版集团
人民邮电出版社
POSTS & TELECOM PRESS

数据中国“百校工程”项目系列教材
数据科学与大数据技术专业系列规划教材

瑞翼教育

Python
编程基础与应用

韦德泉 许桂秋◉主编
方曙东 莫毅 曹红根 钱鸣 史春雷 张钦礼 胡楠 朱长水◉副主编



BIG DATA
Technology

人 民 邮 电 出 版 社
北 京

图书在版编目（CIP）数据

Python编程基础与应用 / 韦德泉，许桂秋主编. --
北京 : 人民邮电出版社，2019.3(2019.3重印)
数据科学与大数据技术专业系列规划教材
ISBN 978-7-115-50346-6

Ⅰ. ①P… Ⅱ. ①韦… ②许… Ⅲ. ①软件工具－程序设计－教材 Ⅳ. ①TP311.56

中国版本图书馆CIP数据核字(2019)第025882号

内 容 提 要

本书从实用的角度出发，采用理论与实践相结合的方式，介绍 Python 程序设计的基础知识，力求培养读者使用 Python 语言解决问题的能力。全书内容包括 Python 程序设计导论、Python 程序设计初步、循环程序设计、函数和递归、Python 数据结构、Python 面向对象程序设计、Python 多线程程序设计。

本书作为 Python 语言的入门教材，目的不在于覆盖 Python 语言的所有知识点，而是介绍 Python 语言的主要语法结构，使读者能够掌握 Python 语言的核心内容，能够在未来的工作中运用 Python 语言编写实用的程序。为了增强实践的效果，本书由浅入深地引入了 5 个综合性的案例，帮助读者理解各种语法知识，并让读者体会如何在实际编程中灵活运用所学知识和技能。

本书可作为高校 Python 程序设计课程的教材，也可供对 Python 感兴趣的读者阅读参考。

◆ 主　　编　韦德泉　许桂秋
副 主 编　方曙东　莫　毅　曹红根　钱　鸣
　　史春雷　张钦礼　胡　楠　朱长水
责任编辑　邹文波
责任印制　陈　犇

◆ 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
固安县铭成印刷有限公司印刷

◆ 开本：787×1092　1/16
印张：10.5　　2019 年 3 月第 1 版
字数：376 千字　　2019 年 3 月河北第 2 次印刷

定价：39.80 元

读者服务热线：(010)81055256　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

信息技术的高速发展，引发了近几年的大数据和人工智能浪潮。信息技术人员作为时代的“弄潮儿”，在对这些波澜壮阔的景象感到兴奋的同时，又深刻感受到技术的飞速变化所带来的巨大压力。

然而，万变不离其宗。拨开各种新奇技术的层层迷雾，我们还是能够看到其中稳定的内核，那就是计算机的基本原理。计算机归根结底是一种存储数据和执行计算的工具。如果人们想要运用这种工具来改善生产和生活，就需要编写程序控制计算机的运行。因此，程序设计便成为了信息技术从业者的一项重要的技能。

世界上的程序设计语言成千上万，1989 年诞生的 Python 语言凭借其易学易用、简洁清晰的特点，覆盖的人群和领域逐年扩张。目前，Python 语言已经稳居各种编程语言排行榜的前列，成为了世界上使用最广泛的编程语言之一。这个现状给了人们充足的理由和动力去学习和使用 Python 语言。

本书作为 Python 语言的入门教材，能够帮助希望成为信息时代冲浪者的读者，从一个外行跨入信息技术的大门。全书采用理论与实践相结合的方式，循序渐进地介绍 Python 程序设计的知识与思路；由浅入深地引入综合性的实践案例，引导读者运用所学知识解决现实中的问题。

全书共 7 章，阐述了 Python 语言的核心语法结构，以及运用这些语法结构编写程序解决问题的思路。书中还包含了 5 个综合性的案例，其难易程度和侧重点与书中各章节的知识相对应。各章具体内容如下。

第 1 章概述 Python 语言的基本情况和编程环境，并详细介绍搭建编程环境的步骤，确保读者能够正确地配置开发环境，为后面的学习做好准备。

第 2 章初步介绍 Python 语言的主要语法结构，使读者对 Python 语言有一个全面的了解，本章还设计了第 1 个实践案例，可实现简单的投掷骰子的功能。

第 3 章着重阐述如何运用 Python 语言解决迭代类型的问题，讲述 Python 语言处理迭代问题的语法，列出一些常见的迭代问题以及解决方案，并介绍了第 2 个实践案例——编程实现一个猜数字的小游戏。

第 4 章深入讲解 Python 语言的函数功能，并且讲述递归程序设计的思路。

第 5 章讲解 Python 内置的、用途广泛的数据结构，并介绍了数据抽象的概念

以及如何使用 Python 表达这种概念。第 5 章设置了第 3 个实践案例，该案例综合运用了前面章节所讲述的 Python 知识，使用 Python 实现链表并表示树结构。

第 6 章介绍面向对象程序设计的基本知识和思路，本章最后设计了第 4 和第 5 个实践案例，通过综合运用前 6 章讲解的 Python 的各种核心语法，实现计算器和一种简单的编程语言解释器，完成了复杂程序的设计和组织工作。

第 7 章介绍了多线程的基本知识和 Python 多线程程序设计的基本方法。

本书可以作为高等学校计算机和信息管理等相关专业程序设计入门课程的教材。使用本书教学建议安排课时为 64 课时，教师可根据学生的接受能力以及高校的培养方案选择教学内容。

特别提示：由于软件在不断升级或网站界面有更新，读者打开的软件下载界面与看到的软件版本可能与本书不一致，没有关系，软件的下载与安装方法是类似的。

由于编者水平有限，编写时间仓促，书中难免出现一些疏漏和不足之处，恳请广大读者批评指正。

编　者

2019 年 1 月

目　录

第1章 Python 程序设计导论

在编程的世界里，没有哪一种语言更好，只有哪一种语言更合适。我们提倡“存在即合理”的理念。当前热门的编程语言都有其存在的道理，它们都有各自擅长的领域和特性。因此，我们无法去衡量哪一门语言是最好的，只能根据具体的应用场景选择最合适的编程语言。

Python 是一门跨平台、开源、免费的解释性高级动态编程语言，其语法精简，安装容易，可扩展性强，越来越受到人们的关注和青睐。本章将要探讨以下 4 个方面的内容。

（1）计算机系统中硬件和软件的作用。

（2）计算机编程语言的形式和功能。

（3）几种 Python 语言的开发环境。

1.1 计算机与程序

计算机是根据指令操作数据的设备。从功能的角度来看，计算机是对数据的操作，表现为数据计算、输入/输出处理和结果存储等；从可编程的角度来看，计算机就是根据一系列指令自动地、可预测地、准确地完成操作者意图的工具。

1.1.1 计算机的基本组成

计算机和网络是信息技术的核心，利用计算机可以高效地处理和加工信息，随着计算机技术的发展，计算机的功能越来越强大，不但能够处理数值信息，而且还能处理各种文字、图形、

图像、动画、声音等非数值信息。下面介绍计算机的组成与工作原理、硬件系统和软件系统相关的知识。

1. 计算机的组成与工作原理

计算机的组成指的是计算机系统结构的逻辑实现，包括计算机内数据信号和控制信号的流向及逻辑设计等。冯·诺依曼于 1945 年提出了存储程序的设计思想，直到今天，计算机仍然采用冯·诺依曼结构。冯·诺依曼将计算机分成五大基本部分：输入设备、存储设备、运算器、控制器和输出设备。计算机的工作原理如图 1-1 所示。

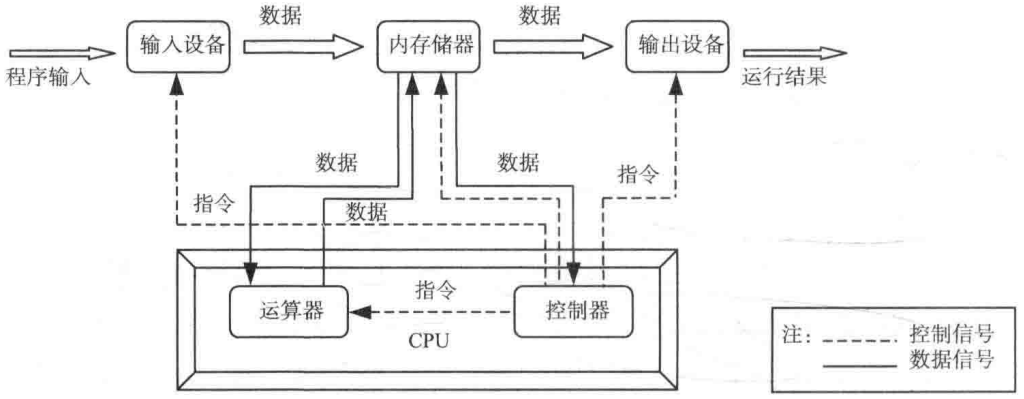


图 1-1　计算机的工作原理

2. 计算机的硬件系统

计算机的硬件系统指构成计算机的所有物理部件的集合。从外观上看，由主机、输入和输出设备组成。根据冯·诺依曼原理，可将计算机分成输入设备、存储设备、运算器、控制器和输出设备。

（1）输入设备（Input Devices）：输入设备是计算机的重要组成部分，输入设备与输出设备合称为外部设备，简称外设。输入设备的作用是将程序、原始数据、文字、字符、控制命令或现场采集的数据等信息输入计算机。常见的输入设备有键盘、鼠标、手写板、触摸屏、扫描仪（光电输入机）、磁带机、磁盘机、光盘机等，如图 1-2 所示。

图 1-2　输入设备

（2）存储器（Memory）：存储器的功能是存储程序、数据和各种信号、命令等信息，并在需要时提供这些信息。存储容量的基本单位是字节（Byte），一个字节由八位二进制数（Bit）组成。为了表示方便，存储单位还有千字节（KB）、兆字节（MB）、吉字节（GB），它们之间的换算关系为 $1KB=2^{10}B=1024B$；$1MB=2^{10}KB=1024KB$；$1GB=2^{10}MB=1024MB$。存储器分为内存和外存。

① 内存：用于存储程序和数据，又可分为只读存储器（Read Only Memory，ROM）和随机存储器（Random Access Memory，RAM），二者的区别如表 1-1 所示。

表 1-1　　ROM 与 RAM 的区别

类别	对信息的修改	断电后的信息情况	用途
ROM	只读	不丢失	永久存放特殊专用信息
RAM	可读、可写	全部丢失	存放临时程序和数据

② 外存：可用于长期存储程序和数据，容量大。软盘、硬盘、光盘、U 盘等都是外部存储器。硬盘是一种硬质圆形磁表面存储媒体，不但存储量大，而且读写速度快，是目前计算机主要的存储设备。

（3）运算器（Arithmetic Unit）：运算器的功能是对数据进行各种算术运算和逻辑运算，即对数据进行加工处理，是计算机实施算术运算和逻辑判断的主要部件。

（4）控制器（Controller）：其功能是对程序规定的控制信息进行解释，根据其要求进行控制，实现调度程序、数据、地址，协调计算机各部分的工作及内存与外设的访问等功能，是指挥、控制计算机运行的中心。其作用是：从存储器中取出信息进行分析，根据指令向计算机各个部分发出各种控制信息，使计算机按要求自动、协调地完成任务。中央处理器（Central Processing Unit，CPU）是运算器和控制器的合称，是微型计算机的核心，人们习惯上用 CPU 型号来表示计算机的档次，例如，286、386、486、Pentium、PⅡ、PⅢ、P4 等。

（5）输出设备（Output Devices）：输出设备与输入设备同样是计算机的重要组成部分，它可以输出计算机的中间结果或最后结果、机内的各种数据符号及文字或各种控制信号等信息。常用的输出设备有显示器、打印机、绘图仪等，如图 1-3 所示。

显示器　　打印机　　绘图仪

图 1-3　输出设备

3. 计算机软件系统

软件（Software）是一系列按照特定顺序组织的计算机数据和指令的集合，是程序、数据和有关文档资料的总称。软件通常被划分为系统软件和应用软件两类。系统软件为计算机使用提供最基本的功能，但是并不针对某一特定应用领域。而应用软件则恰好相反，不同的应用软件根据用户和所服务的领域提供不同的功能。其中，系统软件根据功能又可分为操作系统（OS）、各种程序语言的编译和解释软件、数据库管理系统。

（1）操作系统。操作系统是系统软件中最基础的部分，是用户和裸机之间的接口，其作用是管理计算机的软硬件资源，使用户更方便地使用计算机，以提高计算机的利用率。常见的操作系统有 Windows、Linux。

（2）各种程序语言的编译和解释软件。编译和解释软件的作用是将人类可读可理解的源代码文本，转换为计算机可以运行的机器指令格式。正是由于编译和解释软件的存在，我们才可以使用 Python 或者 Java 这样的高级计算机语言编写程序。

（3）数据库管理系统。数据库是按一定的数据方式组织起来的数据的集合。数据库管理系统的作用就是管理数据库。它一般具有建立、编辑、维护和访问数据库的功能，并提供数据独立、完整及安全的保障。现在市场上常见的数据库管理系统有 MySQL、SQL Server、Oracle 等。

1.1.2 什么是程序

程序是一个特定的指令序列，它告诉计算机要做哪些事，按什么步骤去做。指令是一组用二进制数表示的命令语言，用来表示计算机所能完成的基本操作。程序是为求解某个特定问题而设计的指令序列。程序中的每条指令规定计算机完成一组基本操作。如果把计算机完成一次任务的过程比作乐队的一次演奏，那么控制器就好比是一位指挥，计算机的其他功能部件就好比是各种乐器与演员，而程序就好比是乐谱。计算机的工作过程就是执行程序的过程，或者说，控制器是根据程序的规定对计算机实施控制的。

1.1.3 计算机如何执行程序

程序只有在计算机上运行起来，才能真正产生效果。一个程序从产生到运行，会经历几个典型的步骤：编辑→编译→链接。只有顺利通过这些步骤的程序才能够在计算机上运行起来。

1. 编辑

这个阶段用于产生人工可读可理解的程序源代码。在这个阶段，程序员使用编辑器软件，根据程序设计语言的语法编写源代码。当前主流的程序设计语言有 Java、Python、C、C++等。

2. 编译

人们能够读懂程序源代码，但是计算机并不能直接运行这些源代码。计算机能够认识和执行的是二进制的计算机指令。因此源代码必须转换为二进制指令。这个转换过程被称为编译过程，编译阶段的产物可以由计算机运行，从而能够进行运算或者操作计算机硬件。例如，解决一个数学问题，或者在显示器上播放视频。

3. 连接

现代的计算机程序非常复杂，通常由多个人或者多个团队分工协作完成。每个人/团队负责编写相对独立的一部分程序源代码，并独立编译得到对应的计算机指令。显然，需要某种机制将所有人的工作拼装组合，得到完整的计算机程序。这种机制被称为连接。它的作用是将每个人的独立编译结果，也就是一个程序的部分二进制指令，汇总起来构造出完整的能够最终在计算机上运行起来的程序文件。

经过上述步骤生成的程序文件，如何在计算机上运行呢？计算机有负责存放数据与指令的装置，称为内存；还有能够进行算术和逻辑运算的装置，称为中央处理单元，即 CPU。运行一个程序的时候，首先将该程序加载到内存中，然后将内存中的程序指令按顺序放入 CPU 中执行。CPU 根据具体的每一条指令，或者进行某种运算，或者指挥计算机上的其他硬件工作。这样，一个程序的所有指令执行完毕，就完成了该程序的运行。根据程序中所包含的不同指令，不同的程序就完成了不同的工作。

1.2　Python 语言

Python 是一种计算机程序设计语言。目前有很多种编程语言，比如，比较难学的 C 语言、非常流行的 Java 语言、适合网页编程的 JavaScript 脚本语言等。那么，如何定位 Python 语言？

用任何编程语言来开发程序，都是为了让计算机完成一定的工作，如上传或下载文件，编写一个文档等，而计算机的 CPU 只是负责辨识机器指令，所以，虽然不同的编程语言差异极大，最后都要翻译成 CPU 可以执行的机器指令。而不同的编程语言，即便是做同一项工作，编写的代码量的差距也很大。

比如，完成同一个任务，使用 C 语言要写 1000 行代码，使用 Java 只需写 100 行，而使用 Python 可能只需写 20 行。因此，Python 是一种相当简洁的高级语言。

对于初学者而言，Python 语言是非常简单易用的，连包括 Google 在内的许多大公司都在大规模使用 Python。

使用 Python 可以完成许多日常任务。例如，可以制作网站，很多著名的网站包括 YouTube 就是用 Python 语言开发的；可以做网络游戏的后台，很多在线游戏的后台都是使用 Python 开发的。当然，Python 语言也有不适用的领域，如开发操作系统、手机应用、3D 游戏等。

1.2.1 Python 语言简介

Python 是著名的程序员 Guido van Rossum 在 1989 年圣诞节期间，为了打发无聊时光而编写的一种编程语言。

现在，全世界差不多有 600 多种编程语言，但流行的编程语言只有 20 多种。最近 30 年常用编程语言的排名变化如图 1-4 所示。

Programming Language	2018	2013	2008	2003	1998	1993	1988
Java	1	2	1	1	17	-	-
C	2	1	2	2	1	1	1
C++	3	4	3	3	2	2	4
Python	4	7	6	11	24	13	-
C#	5	5	7	8	-	-	-
Visual Basic .NET	6	11	-	-	-	-	-
PHP	7	6	4	5	-	-	-
JavaScript	8	9	8	7	21	-	-
Ruby	9	10	9	18	-	-	-
R	10	23	48	-	-	-	-
Objective-C	14	3	40	50	-	-	-
Perl	16	8	5	4	3	9	22
Ada	29	19	18	15	12	5	3
Lisp	30	12	16	13	8	6	2
Fortran	31	24	21	12	6	3	15

图 1-4　近 30 年常用编程语言的排名变化图

总之，这些编程语言各有千秋。C 语言是可以用来编写操作系统的贴近硬件的语言，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。

1. Python 语言的发展

Python 语言也是从诸多其他语言发展而来的，包括 ABC、Modula-3、C、C++、Algol-68、SmallTalk、UNIX shell 和其他的脚本语言等。

与 Perl 语言一样，Python 源代码同样遵循 GPL（GNU General Public License）协议。

现在 Python 由一个核心开发团队在维护，Guido van Rossum 仍然在团队中发挥着至关重要的作用。

2. Python 的特点

（1）易于学习。Python 有相对较少的关键字，结构简单，语法定义明确，学习起来容易上手。

（2）易于阅读。Python 代码定义得很清晰。

（3）易于维护。Python 成功的一个很重要的原因在于它的源代码相当容易维护。

（4）拥有广泛的标准库。Python 最大的优势之一是其具有丰富的库，且可跨平台使用，在 UNIX、Windows 和 Macintosh 等不同系统中的兼容性很好。

（5）支持互动模式。互动模式支持用户从终端输入执行代码并获得结果。用户利用互动模式可进行测试和调试代码。

（6）可移植强。基于其开放源代码的特性，Python 已经被移植（也就是使其工作）到许多平台。

（7）可扩展性强。如果用户需要一段运行很快的关键代码，或者是想要编写一些不愿开放的算法，则可以使用 C 或 C++完成那部分程序，然后在 Python 程序中调用它们。

（8）支持数据库。Python 提供所有主要的商业数据库的接口。

（9）支持 GUI 编程。Python 下的 GUI 编程代码可以创建和移植到许多系统中调用。

（10）可嵌入。用户可以将 Python 代码嵌入到 C/C++程序，让程序的使用者获得“脚本化”的能力。

3. Python 语言的优点

（1）提供丰富的基础代码库。当使用一种语言开始做软件开发时，除了编写核心代码外，还需要很多基本的已经写好的现成的代码，来帮助加快开发进度。Python 就为我们提供了非常完善的基础代码库，覆盖了网络、文件、GUI、数据库、文本等大量的编程内容，被形象地称作“内置电池（Batteries Included）”。用 Python 开发，许多功能不必从零编写，直接使用现成的即可。

（2）具有丰富的第三方库。除了内置的库外，Python 还有大量的第三方库，也就是别人开发的，可供用户直接使用的库。当然，如果你开发的代码通过很好的封装，也可以作为第三方库给别人使用。

（3）应用范围广。许多大型网站就是用 Python 开发的，如 YouTube、国内的豆瓣等。很多大公司，包括 Google、Yahoo 等，甚至 NASA（美国航空航天局）都大量地使用 Python。

4. Python 语言的缺点

任何编程语言都有缺点，Python 也不例外。

（1）运行速度慢。与 C 程序相比，Python 的运行速度非常慢，因为 Python 是解释型语言，代码在执行时会一行一行地翻译成 CPU 能理解的机器码，这个翻译过程非常耗时，所以很慢。而 C 程序则是运行前直接编译成 CPU 能执行的机器码，所以运行速度非常快。

但是大量的应用程序不需要这么快的运行速度，因为用户根本感觉不出来。例如，开发一个下载 MP3 的网络应用程序，若 C 程序的运行时间需要 0.001 秒，Python 程序的运行时间需要 0.1

秒，但由于网络更慢，用户还需要等待 1 秒，用户基本上感觉不到 1.001 秒和 1.1 秒的区别。

（2）代码不能加密。如果要发布 Python 程序，实际上就是发布源代码。这一点与 C 语言不同。C 语言不用发布源代码，只需要把编译后的机器码（也就是 Windows 上常见的 xxx.exe 文件）发布出去。要从机器码反推出 C 代码是不可能的，所以，凡是编译型的语言，都没有这个问题，而解释型的语言，则必须把源代码发布出去。

5. Python 语言的语法

Python 是一个高层次的、结合了解释性、编译性、互动性和面向对象的脚本语言。Python 的程序具有很强的可读性，它具有比其他语言更有特色的语法结构。

（1）Python 是一种解释型语言。这意味着开发过程中没有了编译这个环节，类似于 PHP 和 Perl 语言。

（2）Python 是交互式语言。这意味着用户可以使用 Python 提示符直接互动执行编写的程序。

（3）Python 是面向对象语言。这意味着 Python 支持面向对象的风格或代码封装在对象的编程技术。

（4）Python 是初学者的语言。Python 对初级程序员而言，是一种伟大的语言，它支持广泛的应用程序开发，从简单的文字处理，到 WWW 浏览器的制作，再到游戏的开发等。

1.2.2 REPL

Python 是一种交互式的语言，这就意味着用户能够在一个软件环境中以一种互动的方式输入 Python 程序代码，该环境实时地给出代码的执行结果。这种软件环境在计算机的术语中被称为 Read–Eval–Print Loop（REPL），又称 shell。REPL 是一种简单、交互式的计算机编程环境，它采用单用户输入，然后对输入内容进行评估，并将结果返回给用户。在 REPL 环境中编写的程序，总是被分段执行。

1. 简介

在 REPL 中，用户键入一个或多条表达式（而不是整个完整的程序单元），然后 REPL 对这些表达式进行评估并显示结果。“Read–Eval–Print Loop”的命名来源于 Lisp 的如下私有功能。

（1）系统的读取函数接受来自用户的表达式，并将其解析为内存中的数据结构。例如，用户可以输入 S 表达式（+ 1 2 3），系统将其解析为包含 4 个数据元素的链表。

（2）eval 函数获取此内部数据结构并对其进行评估。在 Lisp 中，评估一个 S 表达式，是从函数名开始的，其余的部分是函数的参数。所以函数“+”调用参数 1、2、3，最后得出结果 6。

（3）打印功能是通过调用 eval 来产生结果的，并将结果输出给用户。如果是一个复杂的表达式，eval 会输出一个格式化的结果，以便用户理解。但是，在这个例子中，数字 6 不需要打印很

多格式。

然后，开发环境返回读取状态，并创建一个循环，当程序关闭时，该循环终止。

REPL 有助于探索性编辑和调试程序，因为程序员可以在决定为下一次读提供表达式之前检查打印结果。

由于打印函数的格式化文本的输出结果与读取函数所使用的输入格式完全相同，因此大多数结果以可复制的形式被打印并粘贴回 REPL 中。

2. REPL 的功能

REPL 可以实现的典型功能包括以下几个方面。

（1）显示输入和输出的历史数据。

（2）为输入表达式和结果设置变量，这些变量在 REPL 中是可用的。例如，通常情况下，在 Lisp 中，“*”是指最后的结果，“**”和“***”是指之前的结果。

（3）REPL 的级别。在许多 Lisp 系统中，如果在读取、评估或打印表达式时发生错误，系统不会将错误消息抛回到顶层。它反而会在错误上下文中启动一个更深层的新的 REPL，用户可以检查问题，修复并继续。如果在调试 REPL 中发生错误，则再次启动更深层次的另一个 REPL。通常，REPL 提供特殊的调试命令。

（4）错误处理。REPL 提供重启功能。当重启可用时，若发生了一个错误，那么，系统就会回到一个特定的 REPL 层重新开始执行。

1.2.3　Python 脚本

Python 是一款应用非常广泛的脚本程序语言，在生物信息、统计、网页制作、计算等多个领域都体现出了强大的功能。Python 和其他脚本语言（如 R、Perl）一样，都可以直接在命令行里运行脚本程序。下面简单介绍 Python 3.6 环境下脚本的运行。

下面介绍下载与安装运行 Python 的方法。打开 Python 的官方网站，在首页找到下载页面的链接。可以根据本机的操作系统类型下载相应的安装文件，并安装。安装之后会在开始菜单出现对应的 Python 3.6 文件夹和相应的工具选项。如果选择该文件夹下的 IDLE（Python GUI）工具选项（这是一个功能完备的代码编辑器），则允许用户在这个编辑器中编写 Python 代码。而且输入 Python 的关键字后，按下【Tab】键即可自动补全不完整的代码，如图 1-5 所示。如果单击开始菜单下 Python 3.6 文件夹的 Python（Command Line）便可进入执行 Python 脚本的命令行界面，如图 1-6 所示。在这两个界面中均可运行 Python 脚本。

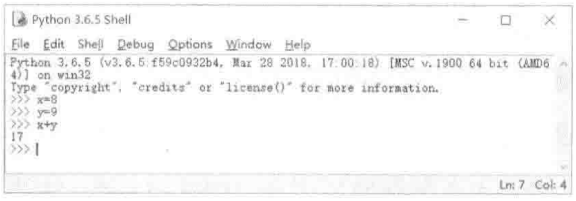


图 1-5　IDIE（Python GUI）代码编辑器

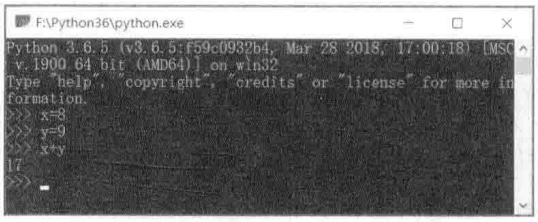


图 1-6　Python 命令行（Command Line）界面

1.3　Python 的开发环境

Python 可应用于多个平台，包括 Linux 和 Mac OSX。用户可以通过终端窗口输入“python”命令来查看本地是否已经安装 Python 以及 Python 的安装版本。下面介绍常用的 Python 开发软件——Anaconda 和 PyCharm 的安装方法。

1.3.1　Anaconda

Anaconda 是一个包与环境的管理器，一个 Python 发行版，以及一个超过 1000 多个开源包的集合。它是免费和易于安装的，并且提供免费的社区支持。

1．首先从 Anaconda 的官方网站下载 Anaconda 安装包。进入官网后，需要根据用户的操作系统（Windows、MacOS 或 Linux）选择与本机系统（是 32bit 还是 64bit）相匹配的 Python 3.0 或以上的下载版本，界面如图 1-7 所示（特别提示：由于软件在不断升级或网站界面有更新，读者打开的下载界面与看到的软件版本可能与本书的不一致，没有关系，软件的下载与安装方法是类似的）。

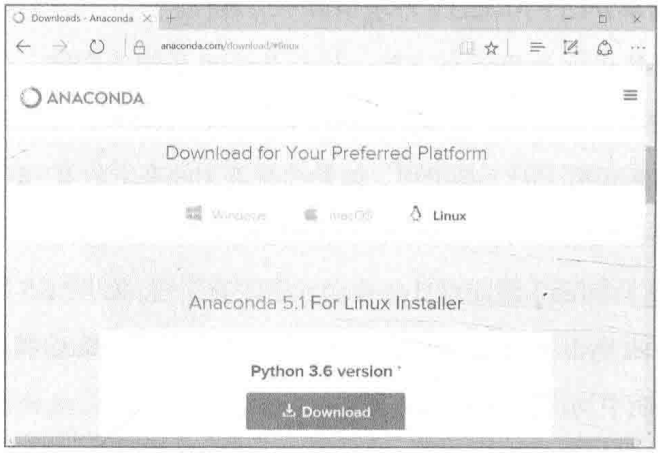


图 1-7　Anaconda 官网下载界面