高等院校计算机应用系列教材

Python 编程基础

袁连海　刘华春　姚　捃　编著

清華大學出版社
北　京

内 容 简 介

本书以通俗易懂的语言、翔实生动的示例全面介绍 Python 语言程序设计的基础知识和编程技术。本书分为 8 章，内容涵盖了 Python 语言概述、Python 语法基础、Python 语言控制结构、函数和代码复用、组合数据类型、文件和数据格式化、Python 程序设计方法、Python 计算生态等。本书每章最后配有练习，可以辅助读者学习 Python。

本书结构清晰、讲解详尽，主要面向初学者，既可作为计算机等级考试培训班和高等院校相关专业的教材，也可作为程序设计爱好者的参考书。

本书对应的电子课件、习题答案和实例源文件可以到 http://www.tupwk.com.cn/downpage 网站下载，也可以扫描前言中的二维码获取。

图书在版编目（CIP）数据

Python 编程基础 / 袁连海, 刘华春, 姚琚编著.
北京 : 清华大学出版社, 2025. 1. -- (高等院校计算机应用系列教材). -- ISBN 978-7-302-67730-7

Ⅰ. TP312.8

中国国家版本馆 CIP 数据核字第 2024NQ9326 号

责任编辑：胡辰浩
封面设计：高娟妮
版式设计：恒复文化
责任校对：成风进
责任印制：刘　菲

出版发行：清华大学出版社
网　　址：https://www.tup.com.cn，https://www.wqxuetang.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-83470000　　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

印 装 者：北京同文印刷有限责任公司
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张：**16.75　　**字　　数：**429 千字
版　　次：2025 年 1 月第 1 版　　**印　　次：**2025 年 1 月第 1 次印刷
定　　价：79.00 元

产品编号：095660-01

前　言

Python 以其简洁的语法和强大的功能，已经成为全球最受欢迎的编程语言之一。随着大数据和人工智能的发展，Python 语言在数据分析和人工智能领域的优势越来越明显，无论是编程领域的初学者，还是希望提高编程技能的编程爱好者，这本教材都将为他们开启一扇通往 Python 编程世界的大门。

本书参照全国计算机等级考试(Python 程序设计)科目大纲编写，从 Python 语言的基本概念和基础语法出发，逐步介绍 Python 编程的基本技术，并将这些技术应用于解决实际问题。书中内容包括 Python 语言概述、Python 语法基础、Python 语言控制结构、函数和代码复用、组合数据类型、文件和数据格式化、Python 程序设计方法、Python 计算生态等。本书在介绍 Python 编程的各种语法知识时，运用了大量的程序实例，注重培养读者解决实际问题的能力。

本书内容丰富、结构合理、思路清晰、语言通俗易懂、示例翔实。每一章内容均结合 Python 关键技术和难点，穿插了大量典型示例。每章最后都安排了有针对性的思考和练习题，思考题可以帮助读者巩固所学的基本概念，练习题有助于培养读者的实际动手能力，并增强对基本概念的理解和实际应用能力。

本书主要面向程序设计初学者，既适合作为 Python 程序设计培训班和高等院校相关专业的教材，也可作为应用程序开发人员的参考书。

在本书的编写过程中，我们得到了许多同行和学生的宝贵意见。特别感谢胡辰浩编辑对本教材内容的审阅和建议。没有他们的支持，这本教材不可能如此完善。除封面署名的作者外，参加本书编写的人员还有李思莉、王小莉、余伟、宋扬等人。由于作者水平有限，本书可能存在疏漏之处，欢迎广大读者批评指正。我们的邮箱是 992116@qq.com，电话是 010-62796045。

本书对应的电子课件、习题答案和实例源文件可以到 http://www.tupwk.com.cn/downpage 网站下载，也可以扫描下方的二维码获取。

作　者

2024 年 7 月

目　录

第1章

Python语言概述

Python 语言由荷兰数学和计算机科学研究学会的吉多·范罗苏姆于 1989 年设计，最初将其作为 ABC 语言的替代品。Python 语言提供高效的数据结构，能简单有效地进行面向对象编程。Python 的动态类型以及解释型语言的本质，使其成为多数平台上写脚本和快速开发应用的编程语言。随着版本的不断更新和新功能的增强，Python 逐渐被用于大型项目的开发。

Python 与其他编程语言相比，更容易入门，适合程序爱好者学习编程。Python 解释器易于扩展，用户可以使用 C 语言、C++语言(或者其他可以通过 C 语言调用的编程语言)扩展新的功能和数据类型。此外，Python 还具有丰富的库，提供了适用于各个平台的源码或机器码。

本章学习目标

- 了解程序设计和算法的重要性。
- 掌握算法的基本概念和特点。
- 了解 Python 语言的特点、发展、应用、版本区别及文件类型。
- 理解 Python 程序的运行方式、开发环境和运行环境配置。
- 掌握 Python 集成开发环境(IDLE)。
- 掌握使用 IDLE 创建简单程序，并调试运行。
- 掌握 Python 第三方库的安装方法。
- 掌握 Python 帮助文档的使用方法。

1.1 程序设计和算法

计算机已经应用到人们日常生活的各个方面，计算机由程序进行控制，而程序是用某种程序设计语言来编写的。程序设计语言是一种用于人与计算机交互(交流)的语言，亦称编程语言，是程序设计的具体实现方式，程序设计语言与自然语言相比，具有更简单、更严谨、更精确等特点。

程序设计语言分为机器语言、汇编语言和高级语言，汇编语言与机器语言属于低级语言，主要用来编写底层的程序。高级语言种类繁多，常见的有 C、C++、Java 以及 Python 语言等，这些语言各自具有自身的优点和缺点，适合不同类型的软件开发。例如 C 语言具有简洁紧凑、使用方便灵活、丰富的运算符和数据结构、结构化的控制语句、语法限制不太严格、能实现较底层的功能，以及生成目标代码质量高和程序可移植性好等优点。因此，C 语言比较适合计算机类专业的学生学习编程，而

对于初次接触编程的非计算机专业的学生而言，Python 无疑是最为简洁、最容易上手的编程语言。

程序是计算机指令的某种组合，程序控制计算机的工作流程，完成一定的逻辑功能，以实现某种任务。程序设计是为特定问题提供解决方案的过程，是软件构造活动中的重要组成部分。程序设计通常被称为编程，已成为当今社会需求量最大的职业技能之一。很多岗位都将被计算机程序接管，程序设计将是未来的重要生存技能。现在，很多中小学生都开始尝试学习编写程序。

关于程序设计，著名的计算机科学家和图灵奖获得者沃斯曾经提出一个经典公式：

程序设计=数据结构+算法

其中数据结构是数据的描述和组织形式，算法是对特定问题求解步骤的一种描述，是独立存在的一种解决问题的方法和思想。对同一个问题，可以有不同的解题方法和步骤，也就有不同的算法。用实际生活中具体事例举例，程序设计相当于做一道菜，数据结构相当于食材和调料，算法则相当于菜谱。例如，要做一道经典川菜回锅肉(程序设计或者编程)，除了需要五花肉、豆瓣酱、蒜苗、食盐等食材(数据结构)，还需要按照回锅肉的做法(算法)进行操作，如先将五花肉煮熟并切片，然后按先后顺序放入菜籽油、豆瓣酱、蒜苗、食盐等。算法具有以下几个特点：

- 仅有有限的操作步骤，即“有穷性”(无死循环)。
- 算法的每一个步骤应当是确定的，即无“二义性”。
- 有适当的输入，即有确定的条件。
- 有输出结果，没有输出结果的算法是无意义的。
- 算法中的每一个步骤都应当有效执行(无死循环)。

算法的表示可以有多种形式，常用的有自然语言、流程图、伪代码、N-S 流程图等。自然语言就是人们日常使用的语言，如汉语、英语等。用自然语言描述算法通俗易懂，但由于自然语言表示的含义往往不太严格，需要根据上下文才能判断其准确含义，因此描述文字冗长，容易引起“歧义”。例如，川菜经典名菜回锅肉制作流程(算法)的自然语言描述如下：(1)五花肉冷水下锅，加花椒粒、姜片、料酒、煮熟至筷子能扎透。(2)捞出五花肉切成薄片备用。(3)用中小火将五花肉片煸出油，加一勺豆瓣酱、一点白糖和少许豆豉。 (4)倒入蒜苗炒至断生。除了很简单的问题以外，一般不用自然语言来描述算法。

流程图用一系列的图像、流程线和文字描述算法的基本操作和控制流程，又称算法流程图或程序流程图。标准流程图所用的符号包括起止框、判断框、处理框、输入输出框、注释框、流程线和连接点等，如图 1-1 所示。其中，“起止框”表示算法的开始和结束；“输入输出框”表示算法输入输出操作，框内填写需输入或输出的各项；“处理框”表示算法中的各种处理操作，框内填写处理说明或算式；“判断框”表示算法中的条件判断操作，框内填写判断条件；“注释框”表示算法中某操作的说明信息，框内填写文字说明；“流程线”表示算法的执行方向；“连接点”的作用是指示流程图中不同步骤或子流程之间的连接和流向。

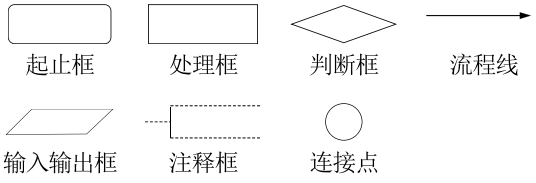


图 1-1　流程图符号

【例 1-1】计算 1+2+3+…+*n* 的和，*n* 从键盘输入，程序的流程图如图 1-2 所示。

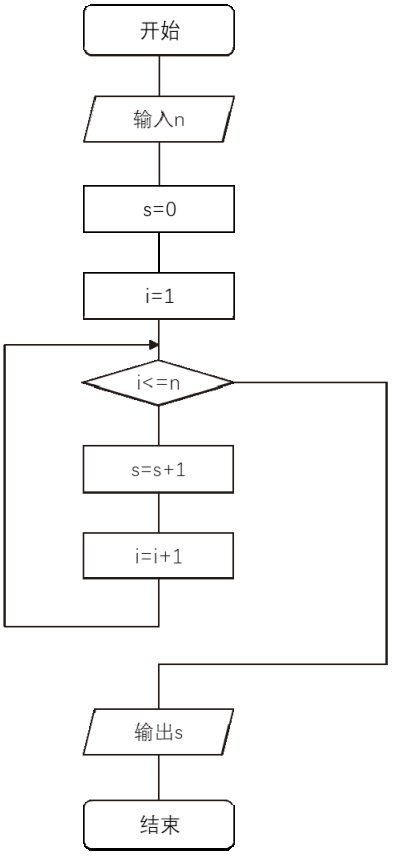


图 1-2　程序流程图

伪代码是一种非正式的，类似于英语结构的语言，用于描述算法的逻辑，介于自然语言与编程语言之间。使用伪代码的目的是使被描述的算法可以容易地以任何一种编程语言(如 Python、C、Java 等)实现。因此，伪代码必须结构清晰、代码简单、可读性好，并且类似自然语言。以下是例 1-1 的伪代码表示：

```
开始(begin)
输入(Input)n
0→s
1→i
当(while) i<=n  执行(do)
s+i→s
i+1→i
循环到此结束(end do)
输出(Print) s
算法结束(end)
```

算法也可以用 N-S 流程图(盒图)表示。这种流程图最早由 I.Nassi 和 B.Shneiderman 在 1973 年发表的题为“结构化程序设计的流程图技术”的文章中提出，因此也称为 N-S 图。N-S 图含有三种基本的控制结构，可以用来构建符合结构化程序设计原则的程序逻辑，如图 1-3 所示。

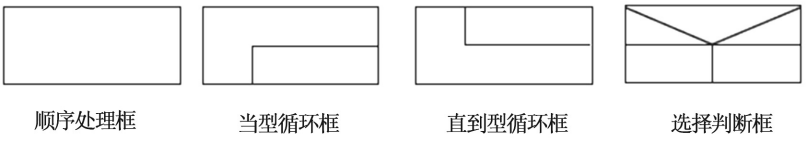


图 1-3　N-S 流程图的基本结构

在程序设计中，还有一种 IPO 程序设计方法(或者 IPO 描述)，指的是在软件设计和开发过程中，遵循的一种基于输入(Input)、处理(Processing)和输出(Output)的基本设计思路。

- 输入(Input)数据：输入是一个程序的开始。程序要处理的数据有多种来源，形成了多种输入方式，包括文件输入、网络输入、控制台输入、交互界面输入、随机数据输入、内部参数输入等。
- 处理(Processing)数据：处理是程序对输入数据进行计算产生输出结果的过程。计算问题的处理方法统称为“算法”，它是程序最重要的组成部分。可以说，算法是一个程序的灵魂。
- 输出(Output)数据：输出是程序展示运算成果的方式。程序的输出方式包括控制台输出、图形输出、文件输出、网络输出、操作系统内部变量输出等。

IPO 描述主要用于区分程序的输入输出关系，重点在于结构划分，流程图侧重于描述算法的具体流程关系，流程图的结构化关系相比自然语言描述更进一步，有助于阐述算法的具体操作过程。算法可以用某种编程语言来实现，对于一个计算问题，可以用 IPO 描述、流程图描述或者直接以代码方式描述。

程序由若干条语句组成，这些语句按照顺序一条一条地执行，这种顺序结构是简洁的。但在现实世界解决问题的过程中，不可避免地遇到需要进行选择或需要循环工作的情况。这时，程序执行的顺序需要发生变化，而不是从前向后逐一执行。因此，程序中除了顺序结构以外，通常还有选择结构和循环结构。三种基本的算法结构如下：

- 顺序结构
- 选择结构(分支结构)
- 循环结构(重复结构)

为了支持这些控制结构，任何编程语言都需要提供丰富、灵活的控制语句。从结构化程序设计的观点看，所有程序都可以使用三种结构，即顺序结构、选择结构和循环结构实现。在默认的情况下，计算机总是按照语句的顺序依次执行，除非特别指明。为使程序更加清晰、易调试和修改，并且减少错误的发生，结构化编程倡导尽量少用或不使用类似于 goto 的跳转语句。

顺序结构的程序设计是最简单的，只要按照解决问题的顺序写出相应的语句即可，其执行顺序是自上而下，依次执行，图 1-4 展示了顺序结构图示。

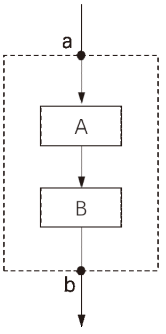


图 1-4　顺序结构

例如，a 和 b 的初始值分别为 a = 1 和 b =2，现在要交换 a 和 b 的值，这个问题类似于交换两个杯子的水，这需要用到第三个杯子。假设第三个杯子是 t，那么正确的程序为：

```
t = a;
a = b;
b = t;
```

执行结果是 a = 2，b = 1。如果改变其顺序，写为：

```
a = b;
b = a;
```

则执行结果就变成 a 和 b 的值都是 2 了，不能达到预期的交换两个变量的目的(初学者最容易犯这种错误)。

顺序结构可以独立使用构成一个简单的程序，常见的输入、计算、输出的程序就是顺序结构的典型例子，例如计算圆的面积：输入圆的半径 r，计算 s = 3.14159×r×r，然后输出圆的面积 s。然而，大多数情况下顺序结构都作为程序的一部分，与其他结构一起构成一个复杂的程序(例如分支结构中的复合语句、循环结构中的循环体等)。

选择结构又称为分支结构，图 1-5 展示了选择结构的示意图。在编写程序时，根据条件 p 是否为“真”，决定执行不同的分支。例如，条件为“真”时执行语句 A；条件为“假”时执行语句 B。利用选择结构，程序员可以根据不同情况选择执行不同的语句。

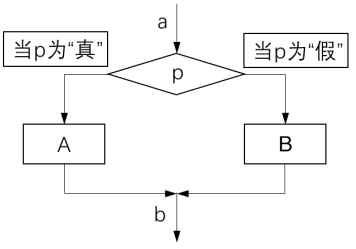


图 1-5　选择结构

循环结构包括当型循环(While 型循环)和直到型循环(Until 型循环)。图 1-6 显示了循环结构

的程序执行流程。当型循环是首先判断循环条件 p1，如果条件 p1 为“真”，则执行语句 A；如果条件 p1 为“假”，则不执行语句 A。直到型循环首先执行语句 A，再判断循环条件 p2，若条件 p2 为“假”，则执行语句 A，直到条件 p2 为“真”，结束循环。

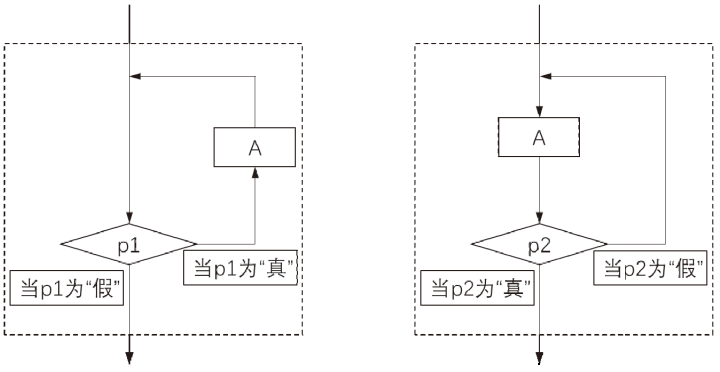


图 1-6　循环结构

以上介绍的三种结构有几个共同点：(1)只有一个入口和只有一个出口；(2)结构内每一部分都有机会被执行到；(3)结构内不存在“死循环”。

1.2 程序设计语言

编写程序时，需要使用某种程序设计语言。程序设计语言包括机器语言、汇编语言和高级语言。

机器语言是计算机可直接执行的二进制代码，例如：11010010 00111011。这样的二进制代码不易被人类理解和记忆，不同的处理器有自己的指令集，通过指令实现对计算机的操作和控制。由于机器语言不便于人们记忆和使用，为了提高编程效率，引入了汇编语言。

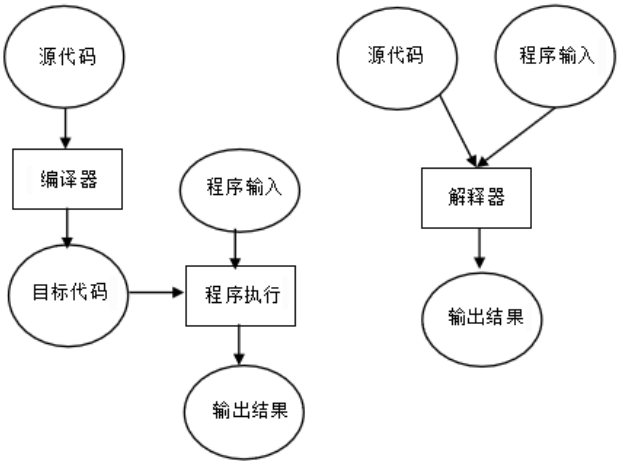
汇编语言是一种低级语言，它使用易于理解和记忆的助记符(如 MOV、ADD 等)来代表特定的机器语言指令。这些指令直接对应于计算机硬件的操作，如加法、减法、数据传输等。汇编语言的指令集通常与特定的处理器或平台相关，因此不同平台之间的汇编代码不可直接移植。汇编语言的指令集非常丰富，涵盖了计算机程序设计的各个方面。由于汇编语言直接操作硬件，因此编写汇编语言程序需要对计算机体系结构有深入的理解。此外，汇编语言程序的执行效率非常高，但在编写复杂程序时，汇编语言不如高级语言方便和易于维护。

高级语言是采用某种编程语言编写的计算机程序，其语法和结构设计使得程序对人类更易阅读和理解，常见的高级语言包括 Java、C、C++、C#、PHP 和 Python 等。这些语言各有特点，适用于不同的应用领域。例如，C 语言是现代高级语言的鼻祖，它强调结构化、模块化和高效率，通常用于系统软件和嵌入式系统开发。Java 语言因其跨平台特性和强大的面向对象设计功能，在企业级软件开发、安卓移动开发和 Web 应用开发等领域广泛应用。Python 则以其简洁而强大的语法和丰富的第三方库支持，适用于人工智能、机器学习、图像处理、科学计算、数据分析和数据可视化等领域。

计算机执行采用高级语言编写的源程序的方式有两种：编译和解释。编译是指编译器将源

程序一次性翻译成目标机器的可执行程序，生成的可执行文件可以在兼容的操作系统上独立运行，类似于把一种语言翻译成另一种语言后再使用。解释则是指解释器在每次程序运行时逐行或逐段地解释源程序，并即时执行，不生成独立的可执行文件，类似于实时翻译并执行。程序设计语言可以分为编译型语言和解释型语言。编译型语言由编译器对源程序文件进行编译和连接，生成可执行的二进制文件，在操作系统中可以单独运行这个可执行的文件，具有运行速度快、代码效率高、编译后的程序不可修改、保密性较好的优点。同时，也具有代码需要经过编译方可运行、可移植性较差的缺点。C 和 C++就是典型的编译型语言。

解释型语言是在运行时逐行或逐段地将源代码翻译成机器语言并执行，而不是一次性将整个程序翻译成机器语言。因此，解释型语言的运行速度通常较慢。例如 Python、Java、C#虽然具有不同的执行方式，但它们都可以被归类为解释型语言。虽然 Java 程序在运行之前也有一个编译过程，但是并不是将程序编译成机器语言，而是将它编译成字节码(可以理解为一个中间语言)。在运行的时候，由 Java 虚拟机将字节码翻译成机器语言。另外，脚本语言一般都有相应的脚本引擎解释执行，一般需要解释器才能运行。例如 JavaScript、ASP、PHP、PERL 等都是脚本语言(它们依赖于脚本引擎的解释执行，而不需要预先编译成独立的可执行文件)。解释型语言的优点是可移植性较好，只要有解释环境，程序就可在不同的操作系统上运行。解释型语言的缺点是运行需要解释环境，运行速度比编译型程序要慢，占用资源较多，代码效率相对较低(因为没有经过编译的优化过程)。图 1-7 展示了编译型语言和解释型语言的不同处理方式。



(a) 编译型语言　　(b) 解释型语言

图 1-7　编译和解释方式

1.3　Python 语言的特点和执行方式

计算机目前已广泛应用于人类日常生活的各个场景。计算机由程序控制，而程序则通过编程语言编写。对于初次接触编程的程序员而言，Python 无疑是最为简洁、易上手的编程语言。

1989 年圣诞节期间，吉多 • 范罗苏姆受到他曾参与设计的 ABC 语言的启发，想要开发一个新的脚本解释程序来继承 ABC 语言，于是 Python 诞生了。Python 被设计为一种解释型、面向对象、动态数据类型的高级程序设计语言。

Python 语法受到了多种语言的影响，其中最主要的是 ABC 语言，它强调了简洁性和易用性。此外，Python 也从 C 语言中借鉴了一些语法思想，尤其是在底层实现和系统交互方面。自诞生以来，Python 已经具有了类(class)、函数(function)、异常处理(exceptional handling)机制，并内置了包括列表(list)和字典(dict)在内的核心数据类型以及以模块为基础的拓展系统。吉多·范罗苏姆于 1989 年定下目标之后便投身于 Python 语言的设计之中，但 Python 的第一个公开版本直到 1991 年才发行，此版本使用 C 语言实现，能调用 C 语言的库文件。2000 年 10 月，Python 2.0 发布，Python 转为完全开源的开发方式。2008 年 12 月，Python 3.0 版本发布，并被作为 Python 语言持续维护的主要系列。

2010 年，Python 2.x 系列发布了最后一个版本，其主版本号为 2.7。同时，Python 的维护者们声称不在 Python 2.x 系列中继续对主版本号升级，Python 2.x 系列逐渐退出了主流支持和广泛使用。

Python 的设计哲学是优雅、明确、简单，它的语法清晰、干净、易读、易于维护，使得编程过程更为简单直接，特别适合初学者。Python 能够让新手专注于编程逻辑，而不被复杂的语法细节所困扰。对于想快速就职的读者而言，学习 Python 无疑是一条捷径。根据各大机构对主流编程语言的排名统计，Python 常年位居前三。

相比于其他编程语言，Python 语言具有以下特点。

- 简单易学：Python 语言是儿童编程入门的语言，其大量应用在科学研究、工程计算、数据分析、数据可视化以及机器学习和人工智能领域。Python 是一种代表简单主义思想的语言，它编写的程序读起来就感觉像是在读英语段落一样流畅。此外，Python 的语法简洁，使得编程人员可以专注于解决问题，而不是语言本身的语法。
- 解释型语言：Python 属于解释型语言，而 C 语言是编译型语言。
- 语法优美：Python 语言是高级语言，它的代码接近人类语言，只要掌握由英语单词表示的助记符，就能大致读懂 Python 代码。
- 免费和开源：Python 是 FLOSS(自由/开放源码软件)之一，用户可以自由地下载、复制、阅读、修改代码。Python 的开源性使其不断改进，并由一群致力于优化 Python 的人推动其发展。
- 跨平台特性：Python 语言编写的程序可以不加修改地在任何平台中运行。Python 程序能够被移植到许多平台上，它无需修改便可以在众多平台上运行，这些平台包括 Linux、Windows 以及 Google 基于 Linux 开发的 Android 平台。
- 面向对象：Python 既支持面向过程编程，也支持面向对象编程。在面向过程的语言中，程序是由封装了可重用代码的函数构成的。在面向对象的语言中，程序是由数据和功能组合而成的对象构建起来的。与其他主要的语言(如 C++和 Java)相比，Python 以一种非常强大且简单的方式实现面向对象编程。
- 扩展性良好：Python 不仅可以引入.py 文件，还可以通过接口和库函数调用由其他高级语言(如 C 语言、C++、Java 等)编写的代码。
- 丰富的库：Python 的标准库特别庞大，涵盖了正则表达式、线程、数据库、网页浏览器、单元测试、GUI(图形用户界面)等各种功能。除了这些标准库之外，Python 中还提供了许多高质量的库，世界各地的程序员通过开源社区又贡献了十几万个几乎覆盖各个应用领域的第三方函数库。

- 通用灵活：Python 是一门通用编程语言，适用于科学计算、数据处理、数据可视化、数据分析、游戏开发、人工智能、机器学习、自然语言处理等各个领域。
- 模式多样：Python 既支持面向对象编程，又支持面向过程编程。
- 良好的中文支持：Python 3.x 解释器采用 UTF-8 编码，支持多种语言字符，包括英文、中文、韩文、法文等。

Python 语言虽然有其优点，但也存在一些缺点。相比于编译型语言(如 C 语言)，Python 的执行效率较低(在实际应用中，Python 的开发速度和易用性往往能够弥补其相对较低的运行效率)。另外，由于 Python 3.x 和 Python 2.x 在语法和特性上有所不同，因此并非所有使用 Python 2 编写的程序都能无须修改就在 Python 3 解释器上运行。

根据执行方式不同，编程语言通常分为静态语言和脚本语言两类。静态语言(如 C、C++和 Java)是使用编译执行的编程语言。脚本语言是使用解释执行的编程语言(如 Python、JavaScript 和 PHP)。

Python 是一门跨平台的脚本语言。脚本程序(用脚本语言开发的程序)以纯文本保存，其执行是由其所对应的解释器解释执行的。Python 解释器用于解释 Python 语句和程序，主要包括 Cpython、Jython、IronPython 和 PyPy。Python 3 源文件采用 UTF-8 编码。

Python 在许多方面得到广泛应用，例如实现 Web 爬虫和搜索引擎中的很多组件都是采用 Python 语言编写的；美国宇航局的 NASA 在它的几个系统中既用 Python 开发，又将其作为脚本语言；全球最大的视频分享网站，其视频分享服务中的大部分使用 Python 编写；许多国内知名网站的前台、后台及服务器的管理平台都使用 Python 编写。Python 是一种广泛应用在多个领域的编程语言，其主要应用领域如下。

- 数据处理和数据分析：Python 提供了丰富的库和工具，如 numpy 库、pandas 库、matplotlib 库和 scipy 库，用于数据处理、可视化、统计分析和机器学习等任务。
- 人工智能和机器学习：Python 在人工智能和机器学习领域得到了广泛应用，目前流行的机器学习库和框架，如 TensorFlow、PyTorch 和 Scikit-learn，都提供了 Python 的接口和支持。
- 网络开发：Python 可以用于构建 Web 应用程序、API 和服务器端开发。流行的 Web 框架(如 Django 和 Flask)，使得使用 Python 进行 Web 开发变得简单和高效。
- 自动化和脚本编程：Python 是一种强大的脚本语言，可用于自动化任务、批处理和系统管理，其简洁性和易读性可以使编写和维护脚本变得更加容易。
- 科学计算和工程：得益于强大的数值计算库(如 numpy 库和 scipy 库)和可视化库(如 matplotlib)，Python 在科学计算和工程领域被广泛应用。
- 游戏开发：Python 可以用于游戏开发，尤其是 2D 游戏。Pygame 是一个流行的 Python 游戏开发库，提供了各种游戏开发所需的功能。
- 网络爬虫：Python 的简洁性和强大的第三方库(如 beautifulsoup 和 scrapy)，使其成为构建网络爬虫和数据抓取应用程序的理想选择。
- 软件测试和自动化：Python 提供了许多测试框架和工具(如 Pytest 和 Selenium)用于软件测试和自动化测试。
- 量化金融和算法交易：Python 在量化金融和算法交易领域具有重要地位。它提供了用于金融数据分析、模型开发和交易执行的库和工具。

此外，Python 还广泛应用于教育、科学研究、图像处理、文本处理、物联网和大数据处理

等领域。因此，不同专业的学生都应学习 Python 语言。

Python 语言文件类型主要分为 3 种：

- 源代码文件(source file)；
- 字节码文件(byte-code file)；
- 优化字节码文件(optimized byte-code file)。

这些文件可以直接运行，不需要编译或连接。

源代码文件包括扩展名为.py 的源代码文件和扩展名为.pyw 的源代码文件，前者由解释器负责解释执行，可在控制台下运行；后者用于图形用户界面(GUI)程序。源代码文件可以用文本编辑器(如记事本)打开并编辑。

字节码文件是 Python 源文件经过编译之后生成的，扩展名为.pyc。字节码文件不能用文本编辑器打开。pyc 文件与平台无关，可以运行在 Windows、UNIX 和 Linux 等操作系统上。通过运行脚本可以将.py 文件编译成.pyc 文件。

优化字节码文件是经过优化代码生成的，扩展名为.pyo 的文件。该类文件不能用文本编辑器打开或者编辑。Python 3.5 版本开始采用 pyc 文件存储优化和非优化代码，不再支持 pyo 文件。

1.4 Python 的安装与配置

1.4.1 安装 Python 解释器

Python 官方网站的下载网址是：https://www.python.org/downloads/，如图 1-8 所示。用户可以登录官方网站下载相应的 Python 版本。在学习时，建议使用最新的稳定版本，不必追求过高的版本。

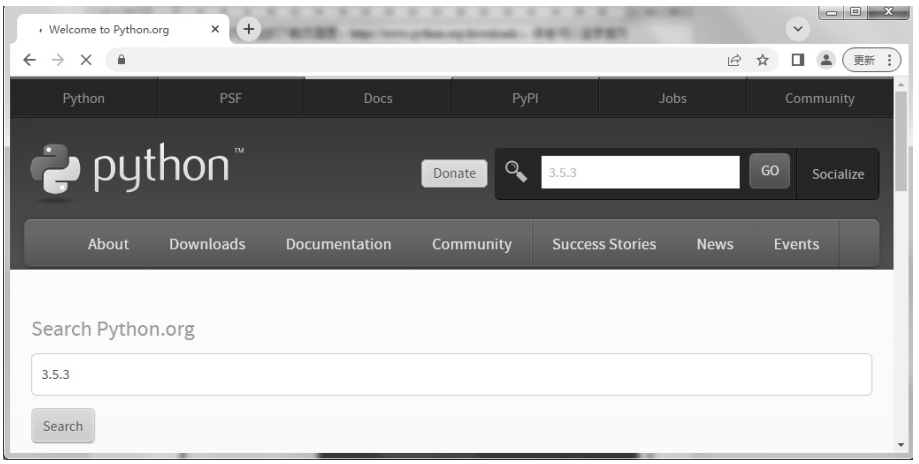


图 1-8　Python 官方下载页面

Python 解释器有许多版本，主要的 Python 标准库更新只针对 Python 3.x 系列，许多企业也正从 Python 2.x 向 Python 3.x 过渡。因此，对于初学 Python 的用户而言，使用 Python 3.x 无疑是一个不错的选择。国家计算机等级考试目前使用的版本是 Python 3.5.3。因此本书以此版本作为安装的解释器，以便帮助用户重点掌握语言的特性。

进入 Windows 版本软件下载页面后，根据操作系统版本选择相应软件包。本书使用的是 Windows 10 操作系统(64 位)，因此应选择 Download Windows x86-64 executable installer 下载 Python 3.5.3 的安装包，如图 1-9 所示。

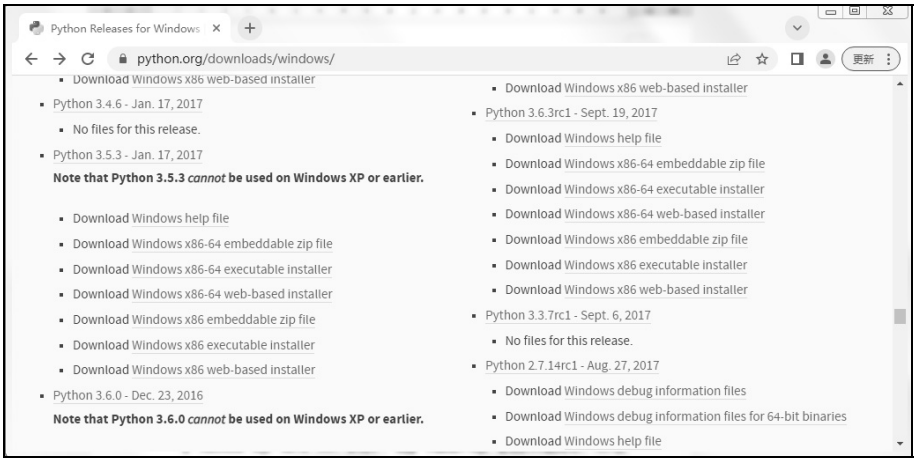


图 1-9　Windows 版本 Python 下载页面

完成 python-3.5.3.exe 安装文件的下载后，双击该文件打开图 1-10 所示安装窗口，选中 Add Pyhon 3.5 to PATH 复选框，选择 Customize installation 选项， 进入图 1-11 所示安装选项界面(这里采用默认设置)。

图 1-10　Python 解释器安装窗口

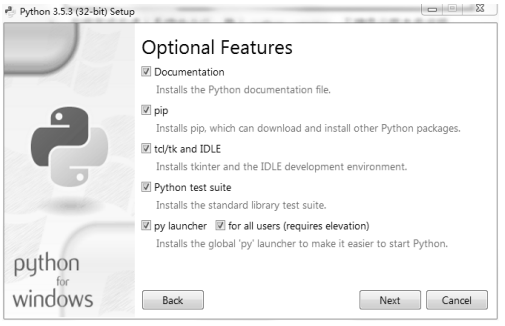


图 1-11　安装选项界面

在图 1-11 所示的界面中单击 Next 按钮进入高级选项界面。选中图 1-12 所示的复选框后，单击 Install 按钮。安装成功后将打开图 1-13 所示的界面。

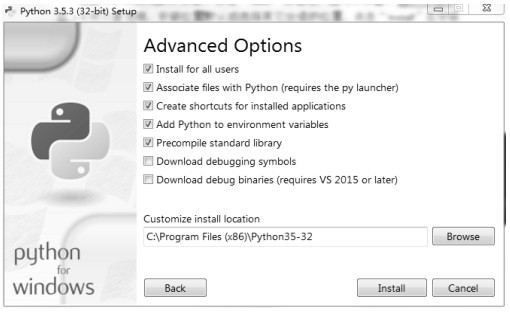


图 1-12　高级选项界面

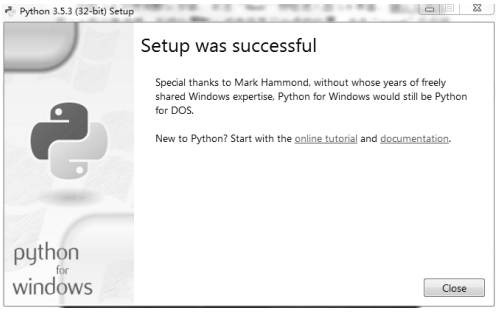


图 1-13　安装成功界面

Python 解释器安装完成后在 Windows 系统的“开始”菜单栏中将显示 Python 3.5 文件夹，其中包含已经安装的 Python 组件，如图 1-14 所示。具体组件描述如下。

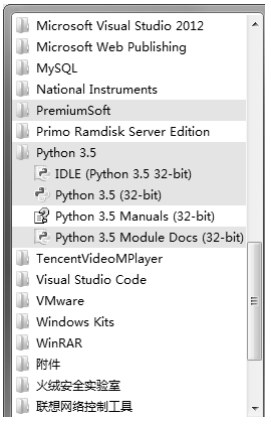


图 1-14　安装的 Python 组件

- IDLE(Python 3.5 32-bit)：集成开发环境，可以在此环境中运行交互式命令或者编写 Python 程序。
- Python 3.5(32-bit)：在命令窗口中打开 Python 解释器。
- Python 3.5 Manuals(32-bit)：Python 语言手册。
- Python 3.5 Modules Docs(32-bit)：模块文档。

1.4.2　运行 Python 程序

Python 程序的运行方式有两种：交互式和文件式。交互式运行方式指 Python 解释器逐行接收 Python 代码并即时响应；文件式也称批量式，指先将 Python 代码保存在文件中，再启动 Python 解释器批量解释文件中的代码。需要注意的是，无论是交互式还是文件式，Python 程序都是通过解释器解释运行的，不需要像 C 语言那样经过编译环节。

图 1-15 所示为在命令提示符下交互式运行 Python 程序的界面。可以通过打开 Windows 命令提示符窗口(依次选择“开始”|“所有程序”|“附件”|“命令提示符”命令打开命令提示符窗口)并输入 Python 命令来启动 Python 解释器。Python 解释器或控制台都可以使用相同的方式交互式运行 Python 程序。例如，在控制台中进入 Python 环境后，在提示符“>>>”后输入以下代码：

```
print("hello world.")
```

解释器将对上述命令解释，并输出结果“hello world.”。

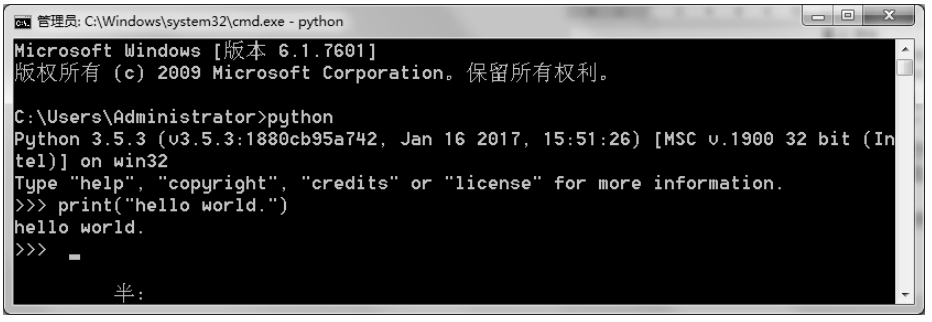


图 1-15　在命令提示符下交互式运行 Python 程序

细心的用户可能已经注意到，Windows 命令提示符是“>”；而 Python 解释器的提示符是“>>>”。

文件方式执行是指创建一个 Python 文件(扩展名为.py)，在其中编写 Python 代码并保存。然后，在该 Python 文件所在文件夹的空白区域按住 Shift 键并同时右击鼠标，在弹出的快捷菜单中选择“在此处打开命令窗口”命令，以打开命令窗口，在命令提示符“>”后输入命令“python hello.py”来运行 Python 程序。

Python 解释器内置了 Python 的官方开发工具——IDLE。IDLE 具备集成开发环境(IDE)的基本功能，开发人员通常根据个人需求或偏好选择使用其他的开发工具。在开始菜单的“Python 3.5”下选择 IDLE 命令，将会打开如图 1-16 所示的集成开发环境。在 IDLE 中，既可以交互式运行程序，也可以编写代码并保存成文件运行。

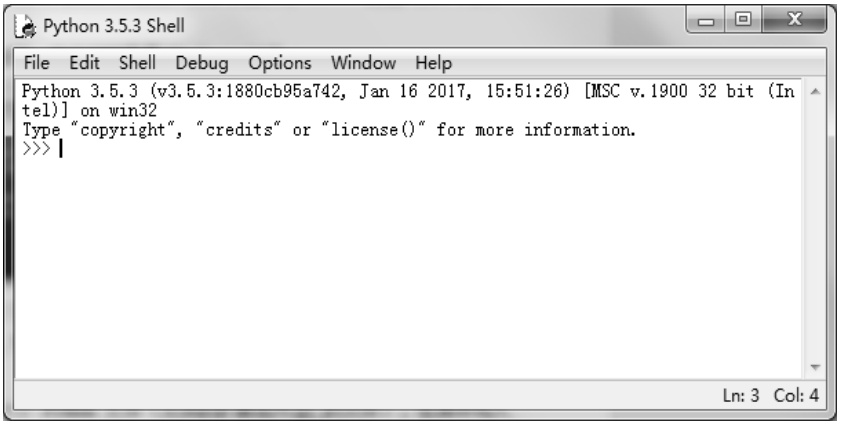


图 1-16　Python 集成开发环境

下面来编写第一个 Python 程序，与所有其他编程语言一样，第一个程序是输出“hello world.”。在图 1-16 所示窗口中，选择 File | New File 命令打开如图 1-17 所示的窗口，在这里，可以输入 Python 源程序代码，也可以进行修改等操作。这个过程类似于使用文本编辑器写文字，只不过书写 Python 源程序有一定的规范(本书第 2 章将详细介绍)。当前阶段，只需要输入如图 1-17 所示的一条语句即可，无须深究这条语句究竟是做什么的。选择 File 菜单下的 Save As 命令，将源程序文件保存在计算机中的某个位置。注意为文件命名，例如图 1-18 中将文件命名为“hello.py”。

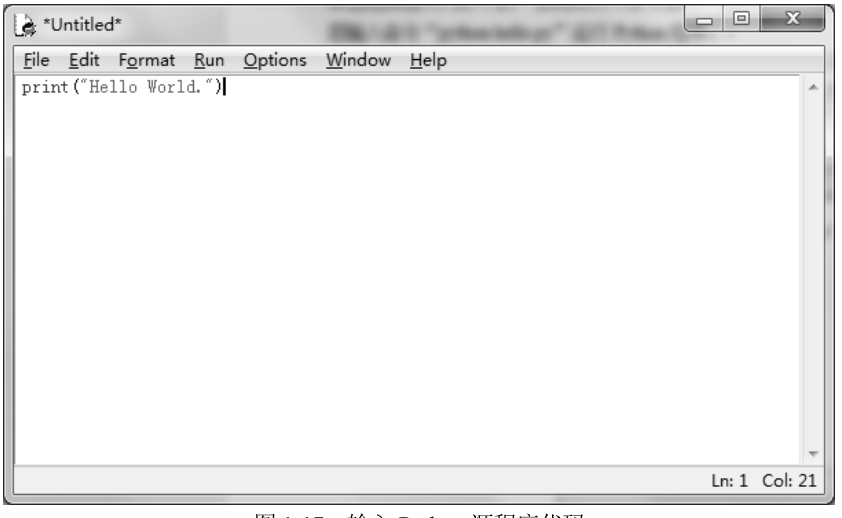


图 1-17　输入 Python 源程序代码

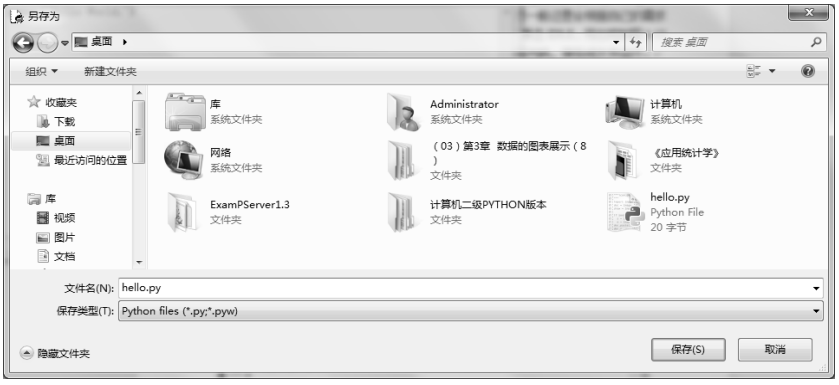


图 1-18　保存 Python 源程序文件

要运行编写好的 Python 源程序文件，可以在 IDLE 中打开源程序文件，然后选择 Run | Run Module 命令运行程序。另外，也可以按 F5 按键运行源程序文件。运行结果如图 1-19 所示。

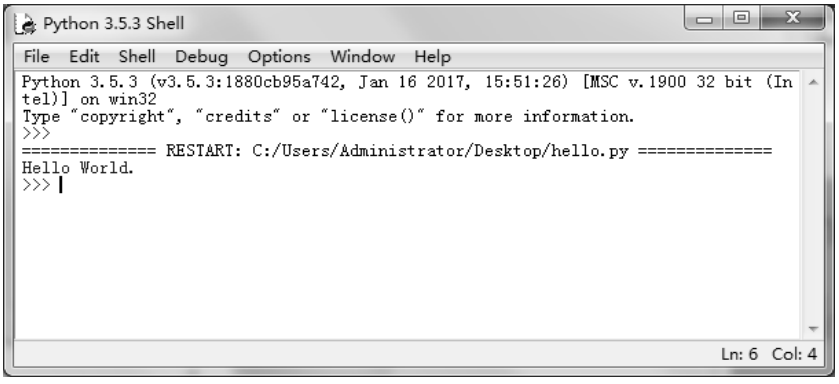


图 1-19　程序运行结果界面

用户可以在 IDLE 交互式窗口提示符下输入以下代码，并查看执行结果。

```
>>> 3+2*5
13
```

“Python 之禅”是隐藏在 Python 语言中的前辈程序员的经验准则，由 Tim Peters 倡导，旨在编写更加优美、简洁、易读、可扩展的程序。这些准则不仅适用于编程领域，也适用于其他领域。

在 IDLE 中执行下列命令，将输出“Python 之禅”(其下的中文翻译引用网络上的翻译，希望用户在编程过程中理解并遵守)。

```
>>> import this
```

The Zen of Python
Python 之禅
Beautiful is better than ugly.
精美优于丑陋。
Explicit is better than implicit.
明确优于含混。
Simple is better than complex.

简明优于繁复。
Complex is better than complicated.
繁复优于难懂。
Flat is better than nested.
平铺直叙优于构架交错。
Sparse is better than dense.
错落有致优于密密麻麻。
Readability counts.
易读性很重要。
Special cases aren't special enough to break the rules. Although practicality beats purity.
在规则面前没有特例，尽管实用性比纯粹性更重要。
Errors should never pass silently. Unless explicitly silenced.
错误不应被轻易忽视，除非明确忽视。
In the face of ambiguity, refuse the temptation to guess.
在模棱两可时，拒绝猜测。
There should be one--and preferably only one--obvious way to do it.
应当有且只有一种明显的方式来做这件事。
Although that way may not be obvious at first unless you're Dutch.
尽管这种方式一开始可能并不明显，除非你是荷兰人(Python 创始人是荷兰人)。
Now is better than never. Although never is often better than *right* now.
现在开始好过从不开始，尽管从不开始好过急于求成。
If the implementation is hard to explain, it's a bad idea.
如果执行方案很难解释，那它一定不是个好主意。
If the implementation is easy to explain, it may be a good idea.
如果执行方案很容易解释，那这或许是个好主意。
Namespaces are one honking great idea -- let's do more of those!
命名空间是个绝妙的主意，让我们多一些这样的想法。

1.4.3　安装 Python 第三方库

Python 语言能够流行的一个重要原因是其包含众多第三方库的支持。除了内置模块以外，有时需要安装额外的第三方库。Python 安装第三方库通常有两种方式：

- 使用 pip 命令行工具在线下载需要的第三方库。
- 手动下载第三方库安装包，然后使用 pip 命令进行安装。

pip 是 Python 的软件包管理命令，是 Python 语言自带的命令行工具，用于安装和管理第三方软件包。使用 pip 工具安装软件包的命令如下所示：

```
pip install some-package-name
```

举例来说，要安装名为 requests 的第三方库(该库用于处理 HTTP 请求)，可以在命令行窗口中执行以下命令(适用于 Windows 系统)：

```
pip install requests
```

这样 requests 第三方库就下载并安装完成了。要在程序中使用 requests 库，只需要在程序代码中添加 import requests 即可。如果执行 pip install some-package-name 命令时遇到找不到软件包的情况，需要手动下载相关的第三方库安装包，然后使用 pip 命令进行安装。

图 1-20 显示系统未安装 jieba 库(jieba 库是支持中文文字处理的第三方库)时，执行 import jieba 将提示没有这个模块。要安装 jieba 库，可以打开命令提示符，输入 pip install jieba 并按 Enter 键。此时，如果计算机已连接互联网，将会自动下载并安装 jieba 库，如图 1-21 所示。

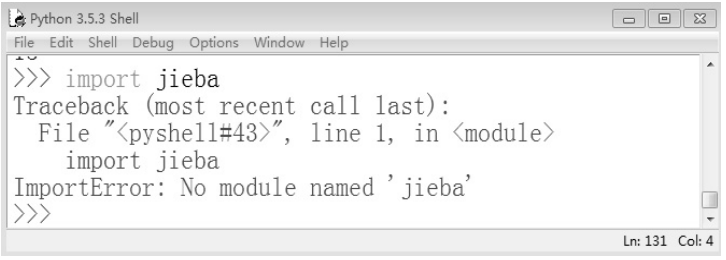


图 1-20　未安装 jieba 库

图 1-21　在命令行窗口中安装 jieba 库

安装完 jieba 库后，在 Python 环境中执行 import jieba 则可以导入并使用 jieba 库。用户可以根据需要安装或卸载特定的库。要卸载某个库可以在命令提示符中使用命令：pip uninstall 库名。例如，输入 pip uninstall jieba，将卸载安装的 jieba 库。

在安装某些库时，如果直接从国外网站下载，则速度较慢。因此，对于想要安装的库，在执行 pip 命令时输入国内镜像站点下载通常可以提供更快的下载速度。

用户可以通过以下命令格式从指定的镜像源网站下载库：pip install -i 镜像源网址 库名称。例如，在命令提示符中执行以下命令表示从镜像站点 https://pypi.tuna.tsinghua.edu.cn/simple 下载 matplotlib 库：

```
pip install -i https://pypi.tuna.tsinghua.edu.cn/simple matplotlib
```

1.4.4　使用 Python 帮助文档

Python 手册是 CHM 格式标准的文档，CHM 是微软新一代的帮助文件格式，利用 HTML 源文件，可以帮助内容以类似数据库的形式编译存储。在 Python 3.5 安装目录下，打开文件资

源管理器并单击“Python 3.5.3 Manuals”，将会打开 Python 帮助文档，如图 1-22 所示。

图 1-22　Python 帮助文档

在图 1-22 左侧选择“搜索”标签页，可以搜索某个关键词，图 1-23 是搜索“turele”关键词后的结果(在图 1-23 中下拉滚动条可以看到一段绘制太阳花的代码)。按照前面介绍的新建 Python 源程序文件的方法新建一个名为 sun.py 的文件，并输入以下代码：

```
from turtle import *
color('red', 'yellow')
begin_fill( )
while True:
    forward(200)
    left(170)
    if abs(pos( )) < 1:
        break
end_fill( )
done( )
```

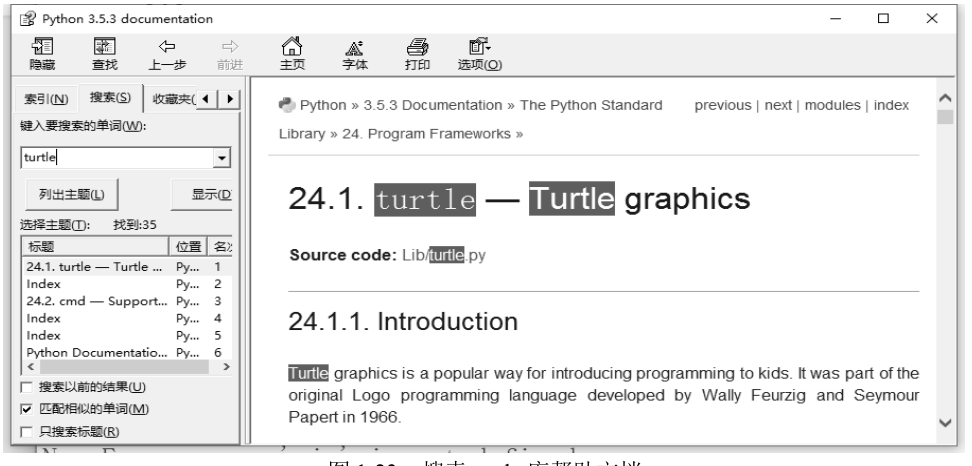


图 1-23　搜索 turtle 库帮助文档

保存文件后，运行 sun.py 文件，将显示如图 1-24 所示的运行结果。

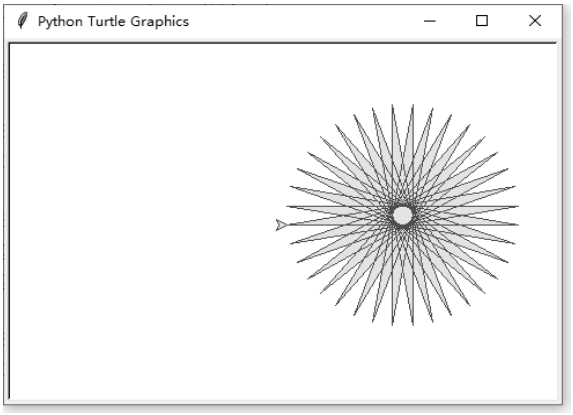


图 1-24　turtle 绘制的图形

除了 Python 手册，用户还可以通过交互式帮助了解 Python 提供的许多标准函数(又称内置函数)。Python 的交互式处理是以人机对话方式获得结果的方法。在命令提示符下输入 Python 命令，然后执行内置函数 help( )可进入交互式帮助系统，如图 1-25 所示。从图 1-25 可以看出，命令提示符从“>>>”变成了“help>”，表明已经进入交互式帮助系统。如果要退出交互式帮助系统，输入 quit 命令即可。

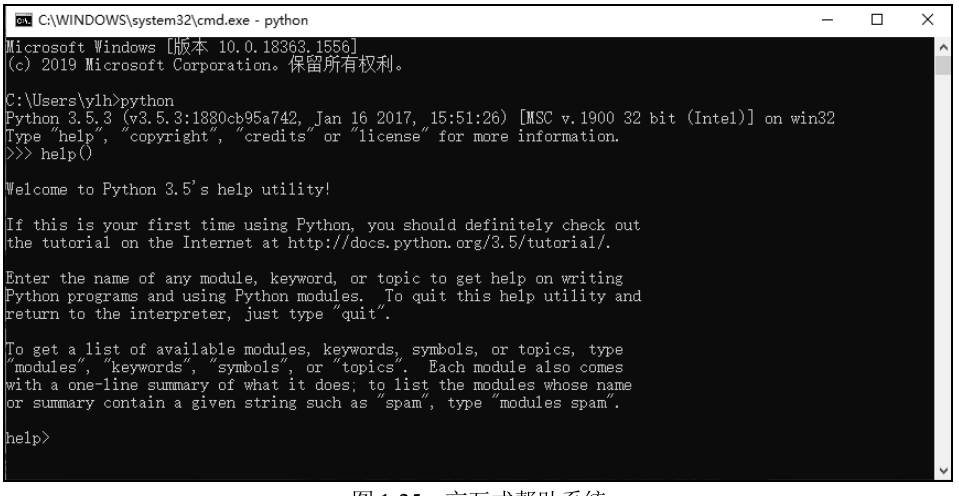


图 1-25　交互式帮助系统

在交互式帮助系统中输入 modules 命令，将显示所有已经安装的模块。要查看某个模块的帮助信息，直接输入模块名称即可。要了解模块中某个函数的信息，可以使用“模块名.函数名”的格式。例如，要查看 random 模块 randint 函数的帮助信息，在交互式帮助系统中输入以下命令，将获得 randint 函数的功能说明。

```
help> random.randint
Help on method randint in random:
random.randint = randint(a, b) method of random.Random instance
    Return random integer in range [a, b], including both end points.
help>
```

1.5 本章小结

本章全面讲述程序设计和算法的基本概念，对算法的特点及描述算法的几种方法进行了介绍。程序的运行包括编译和解释两种，Python 是一种解释型语言。本章还介绍了 Python 语言的特点、发展、应用、版本区别及文件类型(Python 程序的运行方式包括交互式和文件方式)，重点阐述了 IDLE 开发环境和运行环境的配置，并通过 IDLE 创建简单程序，调试并运行 Python 源程序文件。另外，介绍了 pip 工具，它用于安装和卸载第三方库。要学好编程，除自己动手多读、多写程序外，知道如何使用 Python 手册和交互式帮助系统也是必备技能。

1.6 思考和练习

一、判断题

1. 相比 C++程序，Python 程序的代码更加简洁、语法更加优美，但效率较低。　(　　)
2. Python 3.x 版本完全兼容 Python 2.x。　(　　)
3. 模块文件的扩展名必定是.py。　(　　)
4. Python 是编译型语言。　(　　)
5. 算法和实现是一样的。　(　　)

二、填空题

1. Python 是一种面向_____的高级编程语言。
2. Python 可以运行在多种平台，体现了 Python 语言_____的特性。
3. 使用_____关键字可以在当前程序中导入模块。
4. 安装和卸载第三方库的命令是_____。
5. Python 的源代码文件通常以_____作为扩展名。

三、选择题

1. 关于 Python 语言的特点，以下选项描述正确的是(　　)。
   A. Python 语言不支持面向对象　　B. Python 语言是解释型语言
   C. Python 语言是编译型语言　　D. Python 语言是非跨平台语言
2. 下列选项中哪个选项不是 Python 语言的主要应用领域(　　)。
   A. Web 开发　　B. 数据分析　　C. 系统级编程　　D. 桌面应用开发
3. 下列关于 Python 的说法中，错误的是(　　)。
   A. Python 是从 ABC 语言发展起来的　　B. Python 是一门高级计算机语言
   C. Python 只能编写面向对象的程序　　D. Python 程序的效率比 C 程序的效率低
4. Python 是一种什么类型的编程语言(　　)。
   A. 编译型　　B. 解释型　　C. 汇编型　　D. 机器型

5. Python 语言的哪个特性使得它非常适合用于快速开发原型和脚本编写(　　)。

A. 可移植性　　B. 面向对象　　C. 易于学习　　D. 高性能

四、编程题

在 IDLE 开发环境中新建 Python 源程序文件并输入以下程序代码，验证程序运行结果，通过运行以下程序，可以加深用户对 Python 语言的认识。

1. 编写 Python 程序，绘制多个起点相同但大小不同的五角星。

```
import turtle as t
def draw_fiveStars(leng):
    count = 1
    while count <= 5:
        t.forward(leng)                    # 向前走 50
        t.right(144)                       # 向右转 144 度
        count += 1
    leng += 10                             # 设置星星大小
    if leng <= 100:
        draw_fiveStars(leng)
def main( ):
    t.penup( )
    t.backward(100)
    t.pendown( )
    t.pensize(2)
    t.pencolor('red')
    segment = 50
    draw_fiveStars(segment)
    t.exitonclick( )
if __name__ == '__main__':
    main( )
```

2. 编写 Python 程序进行整数求和。输入整数 n，计算 1+2+3+…+n 之和。

```
a = int(input("请输入一个整数："))
s = 0
for i in range(a+1):
    s += i
print("1~%d 的和为：%d"%(a,s))
```

3. 编写 Python 程序进行整数排序。输入三个整数，把这三个数由小到大输出。

```
a = [ ]
for i in range(3):
    x = int(input('请输入整数：'))
    a.append(x)
a.sort( )
for i in a:
    print(i,end=" ")
```

4. 编写 Python 程序，打印九九乘法表。

```
for i in range(1,10):
    for j in range(1,i+1):
        print("%d×%d=%-2d "%(j,i,i*j),end = '')
    print('')
```

5. 编写 Python 程序，打印字符串“Hello，World!”。