高等院校计算机应用系列教材

Python 编程基础与应用

汪治华　张　虎　崔　艳　王艳玲　杨娜娜　编著

清華大學出版社

北　京

内 容 简 介

“只有胸怀全局，才能在思考问题时高瞻远瞩。”熟悉一门编程语言的全貌，才能举重若轻、得心应手地运用其解决编程问题。本书以项目案例为驱动，旨在帮助读者轻松掌握 Python 语言对象体系和编程计算生态的有关知识，并培养读者运用计算思维和软件工程思维进行程序设计的能力。全书内容共分为 14 章。第 1 章从 Python 开发环境的搭建、直观的 turtle 对象绘图程序入手，介绍了 Python 语言描述的对象模型、软件对象的工作方式。第 2～13 章的内容包括：Python 基础，数据类型，运算符，流程控制，组合数据类型，函数，类与对象，异常、调试与测试，文件与数据格式化，标准库应用编程，第三方库应用编程，虚拟环境与程序打包发布。这部分内容完整地讲解了 Python 语言对象体系和编程计算生态的知识点，有利于读者在头脑中搭建起 Python 语言全景的知识框架体系。同时，用计算思维过程方法分析人机大战猜拳游戏程序开发步骤，分别以案例形式讲解了游戏项目问题分解、模式识别、归纳抽象、数据描述、算法设计、流程图设计、面向过程程序开发、面向对象程序开发，将计算思维融入案例开发的步骤之中，有利于读者快速掌握计算思维并实现程序设计。第 14 章介绍了软件工程思维方法，以中国茶叶知识数据爬虫为例，按照软件工程流程，完整地讲解了爬虫的开发过程，有利于读者快速掌握基于 Python 语言的软件工程思维并实现程序设计。

本书不仅适合所有对 Python 语言感兴趣的读者阅读，还适合作为高等院校各专业 Python 语言课程教材和社会培训机构的教材。

图书在版编目(CIP)数据

Python 编程基础与应用 / 汪治华等编著. —北京：清华大学出版社，2024.2
高等院校计算机应用系列教材
ISBN 978-7-302-65434-6

I. ①P… II. ①汪… III. ①软件工具—程序设计—高等学校—教材 IV. ①TP311.561

中国国家版本馆 CIP 数据核字(2024)第 043300 号

责任编辑：刘金喜
封面设计：高娟妮
版式设计：芃博文化
责任校对：成风进
责任印制：刘海龙

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：三河市天利华印刷装订有限公司
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张：**20.75　　**字　　数：**531 千字
版　　次：2024 年 4 月第 1 版　　**印　　次：**2024 年 4 月第 1 次印刷
定　　价：69.80 元

产品编号：103052-01

前　言

数字时代已来临，人工智能时代的大幕也已揭开。人类已处于算力时代，算力是社会的基础生产力。高效地利用算力解决问题已成为衡量社会先进性的重要指标，也是发挥个人和团队工作能力的基础。若要有效地利用算力，就必须具备计算思维能力。计算思维是人类继理论思维、实验思维之后兴起的第三种思维方式。

计算思维(computational thinking)是周以真(Jeannette Marie Wing)教授于 2006 年首次提出的概念。计算思维是运用计算机科学的基础概念进行问题求解、系统设计及人类行为理解等涵盖计算机科学方方面面的一系列思维活动。计算思维是与形式化问题及其解决方案相关的思维过程，其解决问题的表现形式应该能有效地被信息处理代理执行。计算思维建立在计算过程的能力和限制之上，由人与机器执行。计算方法和模型使得人们敢于去处理那些原本无法由任何个人独自完成的问题求解和系统设计。

编程语言的运算符表示对数据进行加工处理的方法。通过运算符进行运算能得到确定结果的问题，都可以通过计算机强大算力的计算得到解决。若要利用计算机算力解决问题，就要用计算机能理解的符号或模型把问题描述出来。如何描述问题？需要利用我们人类大脑的复杂思维活动来描述，这种思维活动称为计算思维。计算思维是建立在算力基础上的一种思维方式。在算力基础上解决实际问题是较为普遍的事情，因此，计算思维是人人都需要具备的一种思维能力。

计算思维对其他学科的研究产生了深刻的影响。例如，计算生物学正在改变着生物学家的思考方式；量子计算正在改变着物理学家的思考方式。计算思维也渗透到了普通人的生活之中，掌握计算思维已经成为现代人应具备的基本技能。计算思维是各专业学生都应掌握的思维方式，可以将其应用于专业研究和专业创新中。掌握计算思维，有助于人们更好地从事医学、法律、商业、政治工作，以及其他任何类型的科学和工程，甚至艺术工作。

Python 语言以其显著的优点成为人们广泛接受的编程语言，是各行各业利用计算机解决问题的重要工具，是一种通用的现代计算语言。Python 语言以其强大的计算生态，得到了各个领域的广泛应用，几乎可以说形成了“Python 万能工具，全民编程”的时代。Python 更是数据分析和人工智能领域的首选语言，利用 Python 进行科学计算的研究机构日益增多，一些知名大学已经采用 Python 来教授程序设计课程。例如，哈佛大学的计算机课程 CS50、卡内基梅隆大学的编程基础、麻省理工学院的计算机科学及编程导论课程等都使用 Python 语言。

Python 是一个用于解决问题的强大利器。若要快速掌握这个利器，就需要快速了解其所有性能，然后择其为我所用之处，修炼绝技，使其发挥最佳效能。本书以全景的方式展示了 Python 语言的知识点，旨在帮助读者快速入门。建议读者采用框架式的学习方法，快速地熟悉 Python 的数据类型、运算符、程序控制、输入输出、模块导入 5 个方面的知识，在头脑中建立起 Python 语言知识框架体系，其他知识点在应用编程中再逐步深入。这样可以达到快速掌握 Python 语言工具并将其应用于实际情境的目标。

在 Python 中，元类对象创建类对象，类对象创建实例对象，实例对象实现具体的工作。Python 面向对象建模的计算思维与自然思维方式一致，语法也接近自然语言，用 Python 语言可以很容易地描述现实世界的对象。通过 Python 学习和掌握计算思维是最佳途径。计算思维分为问题分解、模式识别、归纳抽象和算法设计 4 个步骤。本书以人机大战猜拳游戏项目开发过程为主线，以用到的解决问题的 Python 编程知识点进行章节划分，详细讲解了解决问题的办法。第 3 章数据类型中讲解了用 Python 数据结构描述人机大战猜拳游戏的案例；第 4 章运算符中讲解了用计算思维过程分析人机大战猜拳游戏并设计算法的案例；第 5 章流程控制中讲解了绘制人机大战猜拳游戏算法流程图的案例；第 6 章组合数据类型中讲解了利用列表、字典等组合数据类型设计人机大战猜拳游戏程序的案例；第 7 章函数中讲解了面向过程设计人机大战猜拳游戏程序的案例；第 8 章类与对象中讲解了面向对象设计人机大战猜拳游戏程序的案例。通过案例完整地演绎 Python 计算思维解决问题的步骤，有利于读者快速习得运用计算思维分析问题并编程实现的能力。

运用 Python 语言编程解决问题，终归属于软件工程范畴，因此，应该具备工程化的构建软件项目的能力。本书第 14 章首先介绍了软件工程思维及软件开发流程，软件开发流程可以分为需求分析、方案制定、设计描述、制造编程、检验部署 5 个阶段；然后按照这 5 个阶段讲解爬虫项目开发过程，将软件工程思维融入项目开发中，有利于读者快速掌握软件工程思维和软件项目开发方法。

本书特色体现在以下三方面。

(1) 入门即知全貌。本书全面介绍了 Python 语言的知识点，每个知识点均有练习代码、实训案例，有利于读者快速、全面地掌握 Python 语言工具。

(2) 入门即用对象。在 Python 中，一切皆对象，通过 Python 著名的 turtle 对象绘图库，读者可以直观地学习和使用 Python 对象，然后设计对象，从而有助于将 Python 面向对象计算思维快速融入自然思维。

(3) 入门即会工程。每个程序员都是艺术家，编程既是创造性的活动，也是工程性的活动，软件质量必须靠工程化的技术来保障。把软件工程思维、方法融入实际项目开发中，有利于读者快速掌握软件工程思维和软件项目开发方法。

本书编著团队包括重庆理工大学的汪治华、河南科技大学的张虎、焦作大学的崔艳、黑龙江生态工程职业学院的王艳玲和滨州职业学院的杨娜娜。该团队具有政府部门管理、产业发展规划、科技创业企业孵化、企业经营、产品开发实战经验，以及高等院校科研教学实践经验。本书是编著团队基于多年从事计算机社会服务和高等院校 Python 语言教学实践的经验总结，是在深入理解 Python 语言特点的基础上倾心打造的力作。

在编写本书的过程中，编者参考、引用和改编了国内外出版物中的相关资料及网络资源。同时，还得到了来自社会企业和高等院校专家同仁的关心、指导和大力支持。在此表示诚挚的感谢。

在编写本书的过程中，我们虽竭尽所能地将好的内容呈现给读者，但书中也难免有疏漏和不妥之处，敬请读者批评指正。服务邮箱：476371891@qq.com。

编　者

2023 年 12 月

导　读

什么是程序设计

程序设计是近十几年来蓬勃发展的一门学科。编写程序已不仅仅是计算机相关专业人士的专利，也成为各学科专业人员的基本技能。尤其是 Python 语言的流行，几乎可以说当今世界已是“Python 万能工具，全民编程时代。”

那么，什么是程序设计呢？程序设计是这样一门工程艺术，它将问题求解方案描述成计算机可以执行的形式。首先是问题导向，在任何领域这都是根本，只有找到一个普遍的需求问题，才是一个可用的、受欢迎的、有价值的程序设计的开始。其次是求解方案，这是计算思维要做的工作，包括问题分解、模式识别、归纳抽象、算法设计 4 个步骤，是本书通过人机大战猜拳游戏案例讲解的重要内容。最后是将问题求解方案描述成计算机可以执行的形式，即用编程语言编写计算机可以执行的程序。把问题求解方案转换为程序是一门工程艺术，因为其具有多种方案、算法、代码风格等，是大脑多样性的创造；因为整个过程都必须遵循社会、经济、技术规范，所以是工程。在程序设计过程中要按照软件工程思维工作，包括需求分析、方案规划、设计描述、编程制造、检验部署 5 个阶段，这里的需求分析、方案规划主要是按照社会、经济、技术分析确定需求问题及宏观的规划。计算思维主要运用在软件工程的设计描述阶段。程序设计的内在思维活动是软件工程思维和计算思维，外在形式是特定领域的问题、编程语言工具和最后得到的程序。

Python 程序设计犹如制造多级火箭，基本语法与内建模块是火箭发动机，标准模块是火箭体，第三方模块与自定义模块是火箭增加级，应用程序是用于精准打击、解决问题的火箭弹头。

为什么使用 Python

编写程序有多种编程语言可以实现，如 Python、C、C++、C#、Java、JavaScript 等。为什么选择 Python 呢？其中的原因很多。Python 是一门效率极高的语言，在解决同样的问题时，需要编写的代码行更少。Python 语法简洁，代码简单、清晰、优美，更易于阅读、调试和扩展。Python 拥有广泛的应用领域，如爬虫程序、系统运维工具、科学运算、人工智能和大数据、云计算、金融分析与量化交易等。Python 还广泛应用于科学领域的学术研究和应用研究。Python 具有全球最大的编程计算生态，有丰富的标准库和第三方库。Python 可以灵活地进行组件集成，代码可以调用 C 和 C++的库，可以被 C 和 C++的程序调用，可以与 Java 组件集成，可以与 COM 和.NET 等框架进行通信。Python 的软件质量很高，并且能够提高开发者的效率。Python 具有

良好的一致性，即使代码并非本人所写，也保证了其易于理解。

编者选择 Python 有两个重要的原因：一是 Python 简单直接，直奔问题而去，其易用性和强大内置工具使编程成为一种乐趣而不是琐碎的重复劳动，解决问题的效率极高；二是 Python 具有高品质的社区，社区内有各行各业充满激情的人。大多数程序员都需要向解决过类似问题的人寻求建议，经验丰富的程序员也不例外。当需要有人帮助解决问题时，有一个好社区很重要。大多数学习计算机编程的人最好从 Python 开始，对于把 Python 作为第一门语言学习的人来说，Python 社区无疑是坚强的后盾。

Python 是一门出色的语言，值得我们去学习。

如何快速学习本书

第一步，到 Python 官方网站下载安装 Python 解释器，了解命令行输入指令方式，熟悉 IDLE 的 shell 运行代码和文本编辑器编辑代码、运行代码。

第二步，快速阅读 Python“531”框架式学习法中的 5 个知识分类，然后在 IDLE 中敲代码，运行代码。

(1) 数据类型：3.2 基本数据类型，3.3.3 列表，3.3.4 元组，3.3.6 字典，9.1.2 异常类型。

(2) 运算符：4.1 算术运算符，4.2 赋值运算符，4.3 比较运算符，4.4 逻辑运算符。

(3) 流程控制：5.2 条件语句，5.3 循环语句。

(4) 输入输出：3.3.1 字符串，10.1.2 文件打开与关闭，10.1.3 文件读写。

(5) 程序结构：1.3 模块、包与库。

读者在快速阅读以上基本编程知识后，应该快速学习程序设计思维和方法，学会通过编程解决实际问题。

(1) 程序设计思维：3.4 问题描述，4.11 计算思维，5.1 程序流程。

(2) 基本程序设计方法：5.6 人机大战猜拳游戏程序设计案例，6.7 用列表实现人机大战猜拳游戏程序案例。

(3) 面向过程的程序设计方法：7.1 函数定义和调用，7.2 函数参数传递，7.9 面向过程编程案例。

(4) 面向对象的程序设计方法：8.1 类的定义和对象创建，8.2 属性，8.3 方法，8.4 Python 的对象体系，8.7 面向对象编程案例。

(5) 程序设计模式：8.4.1 object 基类以及单例设计模式(23 种程序设计模式之一)。

第三步，领悟 Python“531”框架式学习法中的三重编程境界。

(1) 语句编程：即基于 Python 基本语法知识逐句编写代码。前面第二步学习的编程都属于语句编程。

(2) 库编程：库编程一般是指基于函数库或类库的编程。Python 库没有明确的定义，是一个功能比较强大的模块或多个模块的一种笼统说法，可以理解为基于模块的编程。

(3) 框架编程：框架一般是指解决特定领域问题的相互联系的多个库的集合。框架编程就是基于框架来开发应用程序的编程，如基于 Scrapy 框架来开发网络爬虫程序。在 Python 中，库和框架的划分不是很明显，都由模块构成，可以笼统地称为库。一般把目的性强、功能强大的模块称为框架。

在第 11 章标准库应用编程和第 12 章第三方库应用编程中可以学习库编程和框架编程的知识。第三方库中著名的 Web 开发模块 Flask 和 Django，一般称为 Flask 框架和 Django 框架。著名的数据分析模块(俗称数据分析三剑客)NumPy、Pandas 和 Matplotlib，习惯上称为 NumPy 库、Pandas 库和 Matplotlib 库。

语句编程比较繁琐，容易出错，效率不高；库和框架编程，开发效率高，程序可靠性好。实际软件生产环境中，都是采用库和框架编程。

第四步，掌握 Python“531”框架式学习法中的软件工程思维，提高软件项目的开发质量。

在第 13 章虚拟环境与程序打包发布中可以学习实际项目开发环境搭建的相关知识。在第 9 章异常、调试与测试中可以学习软件调试与测试的知识。在第 14 章项目开发实战——茶叶数据爬虫开发中可以学习项目开发的思维、方法和流程。

本书旨在让读者快速掌握 Python 编程工具，并将其快速应用到专业研究与专业创新工作中，同时，也为读者成为优秀的 Python 程序员打下坚实的基础。阅读本书后，读者可以进一步深入学习 Python 元编程等高级技术，也可以更轻松地掌握其他编程语言。

配套资源

为便于教学，本书免费提供 PPT 教学课件、案例源代码、习题、试卷、题库及答案。另外，本书还配有实验指导，提供 24 个实验项目(附录 C，实验内容以 pdf 格式提供)。上述资源可通过扫描下方二维码下载。服务邮箱：476371891@qq.com。

教学资源下载

编程之旅

1985—1989 年，成都。初次接触计算机编程是在大二学习 Basic 语言时，那时机房用的是苹果公司的计算机。第一次敲入代码，计算机瞬间运行并显示了结果，那种兴奋之感令我至今难以忘怀，恍如昨天，历历在目。如果不是光学信息处理领域的魅力吸引了我，让我选择了攻读光学信息处理专业的研究生，那么，我一定会选择攻读计算机专业的研究生。

1989—1992 年，杭州。研究生阶段与本科阶段完全不同，可以用华丽来形容。尽管专业课程内容丰富，研究生文艺部的组织工作及文娱活动多姿多彩，绿茵场上野狼足球队热血沸腾，网球场上剑客们灵动潇洒，也不能阻挡我对计算机科学的追逐。在此期间，我自学了 C 和 Fortran 编程语言，还在图书馆阅读了大量计算机硬件、体系结构和算法等方面的书籍。当时我使用物理系机房的多终端 Unix 系统和实验室的 IBM PC/XT 计算机进行上机实验。我的硕士论文“激光环形聚焦系统研究”中就用了大量的计算机编程仿真运行结果。研究生毕业后，我在杭州一家知名的通信公司见习，成为研究光端机、光纤通信的早期追光者之一。然而，也许冥冥之中自有安排，命运把我驱向了计算机科学。

1992—2002 年，温州。当时素有“小香港”之称的温州声名远播，成为商品经济的神话，产业蓬勃发展，创业风起云涌。一脚迈出校门的青年书生，毅然地奔向这方碧波荡漾的大海，拾起 C 语言这把利剑，踏上了基于计算机控制的产品开发的漫漫征程。还记得当初用代码控制硬件点亮第一盏 LED 灯时的情景，LED 灯的闪烁，像夜空的星星一样，一闪一闪的，那种情景至今仍然让我心潮澎湃。天道酬勤，几年间我开发了一系列产品，部分产品得到了市场认可，获得了批量生产。

2002—2023 年，重庆。火辣辣的火锅，火辣辣的人。作为重庆人，我以游子回归的心情拥抱这山水相依、雄奇美丽、冷暖鲜明的城市，以极大的热情投入科技管理、科技成果转化、科技创新创业的企业孵化工作中。国家级两江新区、临空经济区、创新走廊、前沿科技新城，战略性新兴产业发展规划如火如荼；著名软件公司、台湾芯片企业圆桌论道；信息物理系统、工业互联网、云计算、大数据、人工智能推动人类社会走向数字时代和人工智能时代。Python 语言快速发展，我从产业界认识了 Python。产教融合，促使我从事计算机相关教学科研工作，并把 Python 引入课堂教学中。Python 语言以其简洁的语法和丰富的资源，让编程解决问题变得简单直接，让人感觉十分爽快，从此我爱上了 Python。我指导的研究生项目均采用了 Python 语言。

多年的 Python 语言教学经验，特别是基于 Python 语言的智能数据分析课程教学，使我充分地认识到 Python 语言的优点。Python 语言的面向对象思维与人类自然思维一致，语法接近自然语言，Python 的对象模型是编程语言中最好的对象模型。因此，Python 易学、易用、功能强大、软件质量高。对于计算机专业人士来说，Python 是优秀的编程语言之一；对于非计算机专业人员来说，Python 是很友好的编程语言。特别是对于初次学习计算机高级语言的人来说，Python 绝对是不二之选。

在多年的教学科研过程中，我接触了几十本 Python 书籍，但我发现它们都没能充分展示 Python 的对象模型特点，基本上都是按照传统的 C++、Java 等面向对象模型来编写的，这样不仅使简单问题变得复杂，还增加了理解难度。面向对象、分类建模、抽象概念是人类的自然思维方式，也是基本的哲学认知。Python 语言完全符合这一哲学认知，因此，学习 Python 语言应该是自然且容易的。教学实践中也充分证明，按照日常的自然语言思维方式引入 Python 的对象体系进行编程，会使学生学习起来更轻松、愉快。为了让更多的学习者受益，我萌生了编写这本 Python 书籍的念头。经过编著团队近一年的编写，本书终于创作完成了。这是一本使编程不再神秘，变得简单自然的书籍，适合所有对 Python 感兴趣的读者阅读。

从最初接触编程到如今已有 30 多年，作为一个非计算机专业的编程爱好者，我没有经历过系统的专业课程体系的学习，都是通过自己收集信息、查阅资料来学习和应用编程语言，这期间经历过愉快、兴奋，也经历过苦闷、彷徨。如何快速学会与应用一门编程语言，是我一直在探索的问题。最初，我学习编程语言都是通过书籍、杂志、期刊，或者与身边同仁进行简单交流，可以说是一个孤独的学习者。后来才有了伟大的互联网及令人愉快的学习者社区。

如今，Web 2.0 也已经发展了很多年，网上涌现了庞大的开发者社区，提供了丰富的 Python 学习资料。从表面上看，任何人都可以获取这些资源并通过自学掌握 Python。但实际上，网上的资源数量庞大且杂乱无章，知识缺乏系统性，是不利于初学者学习的。在没有足够的编程实践经验之前，仅靠网上资源进行自学是非常困难的。甚至 Python 的官方文档也不适合初学者阅读，更适合有一定经验的软件工程师。因此，为了快速掌握和应用 Python 语言，选择一本好书是非常有效的途径，而本书正是一个较好的选择。

有了一本好书，如何快速、有效地掌握知识也是一个关键问题。对于学习某种编程语言来说，我个人的经验是先把有关编程语言的知识分成 5 部分，即数据类型、运算符、流程控制、输入输出和程序结构，然后去寻找能够明确反映以上 5 部分知识的书籍进行学习。若书中结构清晰并配有简洁的练习代码，那这样的书籍就是用来学习一门编程语言的好书。当然，好书是相对的，没有绝对十全十美的书籍。选择一本好书以后，首先按照以上编程语言的 5 个知识分类，在书中查找并快速阅读相关内容，不求完全理解，只求对该种语言有一个初步的了解即可。例如，对于数据类型部分，快速地阅读该种语言有哪些数据类型；对于运算符部分，快速地阅读该种语言有哪些运算符。然后按照这 5 个知识分类集中敲代码，直观感受代码的运行结果。很多编程知识只看理论很难理解和记忆，但一敲代码，就会立刻理解和掌握。因此，人们常说学习编程语言的唯一秘诀就是敲代码。我个人学习编程语言的诀窍是“五分类快速阅读＋五分类快速敲代码”，我将这种方法称为框架式学习法。采用这种学习方法，学习第一门编程语言可能需要多花费一些时间，但学习第二门及以后的编程语言时，基本上只需要花费一到四周的时间，就可以熟练掌握并将其应用于实际编程问题的解决中。当然，编程功底、技巧的修炼，却是一个长期持续的过程。

对于 Python 语言的学习，可以总结为“531”框架式学习法。“5”是指把 Python 语言知识分为 5 部分，即数据类型、运算符、流程控制、输入输出和程序结构；“3”是指编程的三重境界，即语句编程、库编程、框架编程；“1”是指软件项目解决问题的 1 个思维，即软件工程思维(包括计算思维)。

“531”框架式学习法，是学习编程语言的高效途径！

编　者

2023 年 12 月

作者简介

汪治华：重庆人，副教授，近年主讲 Python 智能数据分析程序设计课程，曾从事计算机硬软件产品开发工作，担任企业技术部门开发工程师、技术中心负责人；曾从事政府部门科技产业规划、科技企业孵化工作，担任科技部门负责人；长期从事高等院校计算机硬软件技术教学科研工作，讲授 Python、C/C++、C#、Java、PHP、JavaScript 等编程语言，以及 SQL、Cypher 等数据库语言，教学经验丰富。研究方向为信息物理系统、人工智能和教育机器人。

张虎：河南孟津人，讲师，讲授 Java 程序设计、Python 程序设计、Android 移动开发、计算机组成原理等课程；曾获省教学技能大赛一等奖，荣获省教学技能标兵、国家电子设计大赛优秀指导教师、省工训大赛优秀指导教师、蓝桥杯优秀指导教师称号；研究方向为软件工程和单片机应用。

崔艳：河南沁阳人，副教授，讲授 Python、Java 语言等课程，主持学习通平台精品资源课程建设；多次指导学生参加蓝桥杯软件大赛、职业技能大赛等学科竞赛；多次荣获优秀指导老师称号；研究方向为软件开发技术和物联网应用技术。

王艳玲：黑龙江哈尔滨人，副教授，黑龙江省第一届职业技能大赛人工智能训练赛项铜牌获得者，龙江技术能手；讲授 Python、Java 等计算机语言课程，主持过中国大学 MOOC 平台开放课程建设；研究方向为软件开发、大数据分析和大数据应用开发。

杨娜娜：山东滨州人，讲师，讲授 Python 运维开发等课程；多次指导学生参加全国与省级云计算、大数据应用设计等技能大赛，并多次荣获优秀指导教师称号；研究方向为云计算、大数据和信息安全。

目　录

∞ 第1章 ∞

绪　　论

学习目标：

1. 了解 Python 语言的发展、特点与应用
2. 掌握 Python 解释器的安装、编程环境的配置
3. 掌握利用集成工具编写、运行 Python 程序的方法
4. 掌握 Python 模块的安装与使用方法
5. 熟悉面向对象程序 turtle 绘图

思政内涵：

Python 语言的发展体现了创新与共享精神，我国开发的北斗卫星导航系统已向全球提供服务，展现了中国的科技创新与共享精神，是科技强国的重大成果。广大学子要勇于开拓，敢于创新，树立忠于祖国、勇攀科学高峰、报效国家的理想信念。

1.1 Python 概述

1.1.1 Python 的发展

1989 年，荷兰人吉多 • 范罗苏姆(Guido van Rossum)在圣诞节休假期间，为了打发时间，决定开发一门解释型编程语言。由于吉多是英国蒙提 • 派森(MontyPython)喜剧团的忠实粉丝，于是他把开发的语言取名为 Python，从此，宣告了一门优秀计算机语言的诞生。

吉多设计 Python 语言的灵感来自开发 ABC 语言的积累，他是 ABC 语言设计团队的一名成员。ABC 语言是 NWO(荷兰科学研究组织)旗下 CWI(荷兰国家数学与计算机科学研究中心)主导研发的一种交互式、结构化高级语言，旨在替代 BASIC、Pascal 等语言。吉多认为 ABC 语言非常优美而强大，但是其最终却没有流行起来。就吉多本人看来，ABC 失败的原因是它作为一门高级语言出现得为时过早，并且平台迁移能力弱，难以添加新功能。另外，ABC 语言过于专注于编程初学者，没有把有经验的编程人员纳入其中。吉多通过 Python 解决了这些问题，使得拓展模块的编写非常容易，并且可以在多平台运行。Python 实现了他的愿景，即在 C 和 Shell 之间创建一种全功能、易学、可扩展的语言。

1991 年，Python 第一个公开发行版本发行。

2000 年 10 月，发布 Python 2.0 版本。

2001 年 3 月，Python 软件基金会(Python Software Foundation，PSF)在美国特拉华州成立。PSF 是致力于推进 Python 编程语言的开源技术的非营利性组织。

2008 年 12 月，发布 Python 3.0 版本，形成了 Python 2 和 Python 3 双线发展的格局。

2011 年 1 月，Python 被 TIOBE 编程语言排行榜评为 2010 年度语言。

2014 年 11 月，Python 组织宣布在稳定版本 Python 2.7 后，于 2020 年起停止支持 Python 2，并将专注于 Python 3 的发展。

2021 年 12 月，Python 在 TIOBE 编程语言排行榜升至第一名。

2022 年 12 月，Python 继续保持在 TIOBE 编程语言排行榜第一名的位置。

Python 已经成为较受欢迎的程序设计语言。在当前数字时代，数据科学、人工智能蓬勃发展，Python 更是数据分析和人工智能领域的首选语言。越来越多的研究机构选择使用 Python 进行科学计算，一些知名大学也已经采用 Python 来教授程序设计课程。例如，哈佛大学的计算机课程 CS50、卡内基梅隆大学的编程基础、麻省理工学院的计算机科学及编程导论课程等都使用 Python 语言。

1.1.2 Python 的特点

1. 简单易学

Python 的设计理念是优雅、明确、简单。Python 有极其简单的语法，使用户能够专注于解决问题而不是去搞明白语言本身。

2. 免费开源

Python 是自由/开放源码软件之一。用户可以自由地发布该软件的副本，使用和改动它的源代码或将其中一部分用于新的自由软件中。正是这种开源和共享的特性促进了 Python 的快速发展。

3. 高级解释性语言

Python 语言是一门高级编程语言，Python 解释器先把源代码转换成字节码的中间形式，然后把它翻译成计算机使用的机器语言并运行。程序员在开发时无须考虑底层细节。

4. 面向对象

Python 既支持像 C 语言一样面向过程的编程，也支持如 C++、Java 语言一样面向对象的编程。

5. 可移植性

Python 可在 Linux、Windows、FreeBSD、Macintosh、Solaris、OS/2 和 Android 等平台上运行。

6. 可扩展性

Python 提供丰富的应用程序接口、模块和工具，使得程序员可以轻松地使用 C、C++语言来编写扩充模块。

7. 可嵌入性

Python 程序可以嵌入 C、C++、Matlab 程序，从而向用户提供脚本。

8. 丰富的库

Python 具有丰富的标准库和第三方库，可以处理各个领域的问题。

1.1.3　Python 的应用

Python 已经形成了全球最大的编程计算生态体系，在科学研究、工业生产、金融投资和商务办公等方面都得到了广泛应用。

1. 科学计算与数据分析

Python 被广泛运用在科学计算和数据分析中。例如，Numpy、Scipy 等 Python 扩展工具，经常被应用于生物信息学、物理、建筑、地理信息系统、图像可视化分析、生命科学等领域。

2. 人工智能与机器学习

当前具有较大影响力的人工智能框架 TensorFlow、PyTorch 都提供了 Python 支持，一些机器学习方向、深度学习方向和自然语言处理方向的网站基本都是通过 Python 实现的。

3. 系统网络运维

在运维工作中有大量的重复性工作，需要采用管理系统、监控系统、发布系统等实现自动化，提高工作效率。在这样的场景下，Python 是一门很合适的用于网络运维的语言。

4. 常规软件开发

Python 支持函数式编程和面向对象程序设计，能够承担各种类型软件的开发工作。

5. 图形用户界面

Python 带有图形用户界面（GUI）开发模块，可以很好地支持应用程序图形用户界面开发。

6. 数据库编程

程序员可通过遵循 Python DB-API 规范的模块与 Microsoft SQL Server、Oracle、Sybase、DB2、MySQL、SQLite 等数据库通信。

7. 网络编程

Python 可提供丰富的模块支持 Sockets 编程，能方便、快速地开发分布式应用程序。很多大规模软件开发计划(如 Zope、BitTorrent、Google)都在广泛地使用它。

8. 网络爬虫

网络爬虫是大数据行业获取数据的核心工具。能够编写网络爬虫的编程语言有很多，但 Python 绝对是其中的主流语言。

9. Web 应用开发

Python 具有一些优秀的 Web 框架，如 Django、Flask 等。很多大型网站都使用基于 Python 的 Web 框架开发，如百度、豆瓣等。

1.2 Python 开发环境

Python 可以在 Windows、Linux、Mac 等操作系统中运行，在不同的操作系统中安装 Python 的方法是不同的，Python 的开发环境也有很多种，本节介绍在 Windows 系统中安装 Python 官方标准版开发环境、PyCharm 集成开发环境、Python 的第三方发行版 Anaconda 开发环境和基于 Web 的在线开发环境。

1.2.1 Python 官方标准版开发环境

Python 的官方网址是 https://www.python.org，进入官网主页，在 Downloads 菜单中，可以下载 Python 的最新版和各个历史版本开发环境；在 Documentation 菜单中，可以下载 Python 的文档、教程和指南等。操作步骤如下。

(1) 进入 Python 的下载页面(见图 1.1)。选择 Windows 平台安装包，选择需要的 Python 版本。本书采用 Python 3.11.1 版本。下载后的安装包文件名是 python-3.11.1-amd64.exe。

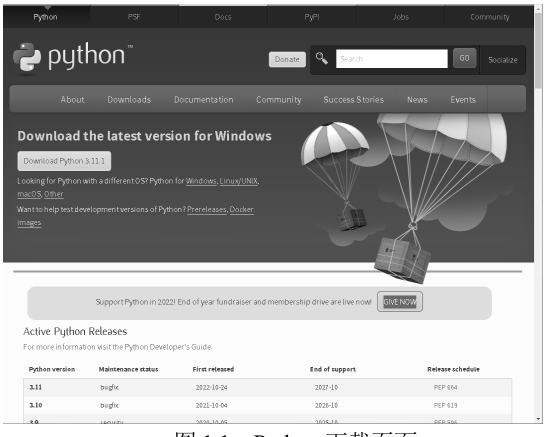

图 1.1 Python 下载页面

(2) 双击安装包 python-3.11.1-amd64.exe，进入 Python 解释器安装界面(见图 1.2)，按照提示进行安装即可。

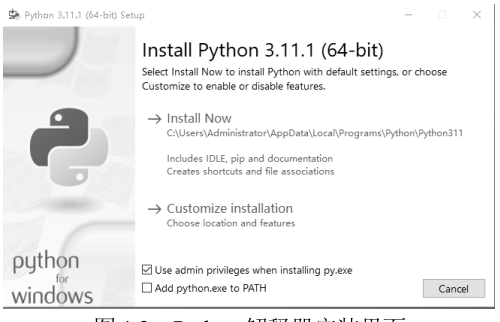

图 1.2 Python 解释器安装界面

1.2.2　第一个 Python 程序

Python 安装成功后，进入 Windows 10 操作系统开始菜单。单击 Python 3.11 文件夹图标，可以看到文件夹中有 4 个文件，第一个 IDLE(Python 3.11 64-bit)是 Python 集成开发环境，第二个 Python 3.11(64-bit)是 Python 交互模式执行文件，第三个 Python 3.11 Manuals(64-bit)是 Python 使用手册，第四个 Python 3.11 Module Docs(64-bit)是 Python 模块文档，如图 1.3 所示。Python 的执行有两种方式，一种是以交互模式(interactive mode)执行，另一种是以 Python 文件方式执行。

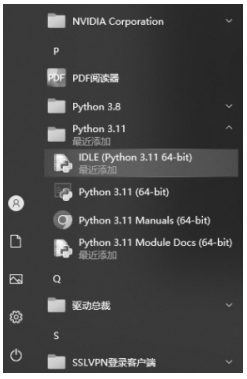

图 1.3　Windows 开始菜单中的 Python 3.11 文件夹

1. Python 交互模式

(1) 单击 Windows 开始菜单中的 Python 3.11 文件夹下的第二个文件 Python 3.11(64-bit)，执行后界面如图 1.4 所示。符号>>>是交互运行模式主提示符。

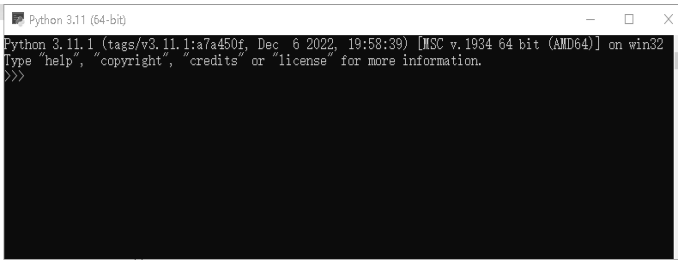

图 1.4　Python 解释器命令行交互执行界面

(2) 在主提示符>>>后输入 Python 的输出字符串语句“print('Hello,Python！')”，执行后输出字符串“Hello,Python!”。Python 默认编码是 UTF-8，可以直接输出汉字。例如，输入语句“print('你好，Python!')”，执行后输出“你好，Python!”，如图 1.5 所示。

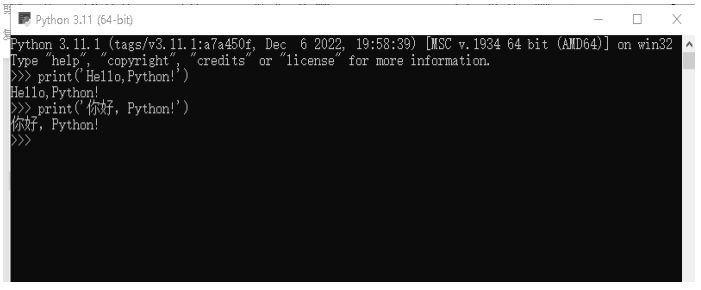

图 1.5　Python 解释器命令行执行语句情况

2. Python 文件执行模式

(1) 在文件执行模式中，首先用文档编辑器编辑 Python 源程序，简单的程序可以用记事本编辑。编辑只有两个输出语句的程序，如图 1.6 所示。将编写的程序保存为名为 hello.py 的文件，并将其放置在 D:\test 文件夹中。

(2) 在操作系统的命令行(在 Windows 中是 CMD 命令行)找到 Python 源文件所在的文件夹 D:\test，然后输入 python hello.py，按回车键后解释器执行 Python 文件并输出结果，如图 1.7 所示。

图 1.6 用记事本编辑 Python 文件

图 1.7 解释器执行 Python 文件

1.2.3 Python IDLE 的使用

单击 Windows 开始菜单中的 Python 3.11 文件夹下的第一个文件 IDLE(Python 3.11 64-bit)，执行后进入 IDLE Shell 界面，如图 1.8 所示。在 IDLE Shell 的命令行提示符>>>后也可以输入 Python 语句，由 Python 解释器交互执行，执行语句后立即输出结果。这里主要介绍 IDLE 集成开发环境。

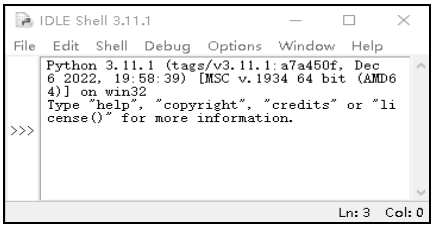

图 1.8 IDLE Shell 界面

(1) 单击 IDLE Shell 界面菜单中的 File(见图 1.9)，选择 New File，进入 IDLE 文件编辑界面(见图 1.10)。

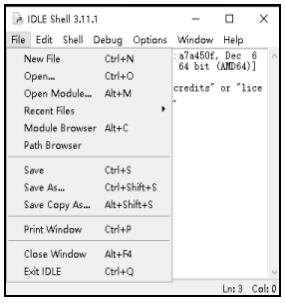

图 1.9 IDLE Shell 菜单界面

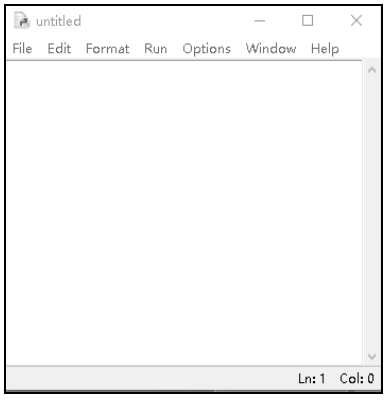

图 1.10 IDLE 文件编辑界面

(2) 在该界面中输入 Python 语句“print('Hello,Python! ')”，然后执行 File|Save As(另存为)命

令，自定义存储路径，Python 程序就编写完毕了，如图 1.11 所示。文件名为 HelloPython.py，Python 文件的后缀是.py。

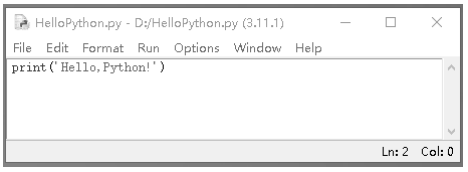

图 1.11　Python 程序 HelloPython.py

(3) 执行 Run | Run Module 命令(见图 1.12)，程序执行结果如图 1.13 所示。

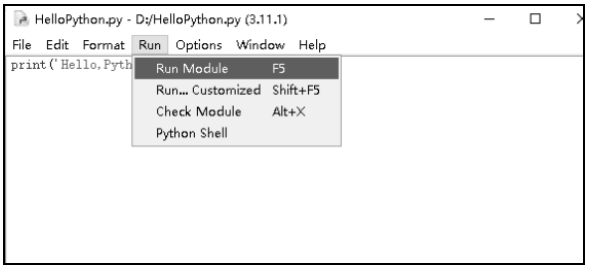

图 1.12　执行 Run | Run Module 命令

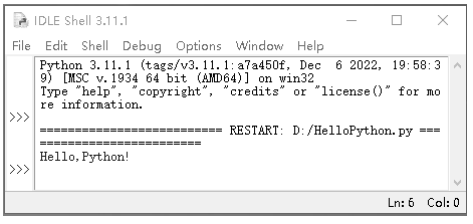

图 1.13　程序执行结果

1.2.4　PyCharm 集成开发环境

PyCharm 是一种 Python 集成开发环境(integrated development environment，IDE)，带有一整套可以帮助用户在使用 Python 语言开发时提高其效率的工具，如调试、语法高亮、项目管理、代码跳转、智能提示、自动完成、单元测试、版本控制等。PyCharm 分为专业版和社区版，专业版是付费的，功能齐全，可以免费试用 30 天；社区版是免费的，功能相对简化，但足够日常学习使用。这里下载社区版安装使用。

1. PyCharm 的下载与安装

(1) PyCharm 的官方网址是 https://www.jetbrains.com/，进入官网主页，如图 1.14 所示。单击 Developer Tools，进入如图 1.15 所示的页面，然后执行 IDEs | PyCharm 命令，进入下载页面。

(2) 进入下载页面后，下拉到下载部分(见图 1.16)，左边是专业版的下载选项，右边是社区版的下载选项，通过下拉按钮可以选择不同操作系统的安装文件进行下载，这里选择社区版.exe(Windows)安装文件进行下载。下载完成后，在浏览器的默认下载目录中有一个可执行的安装包，文件名是 pycharm-community-2022.3.exe。

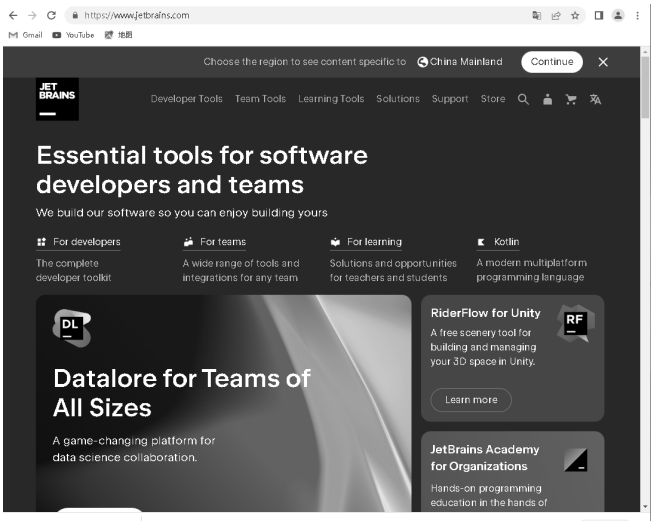

图 1.14　PyCharm 官网主页

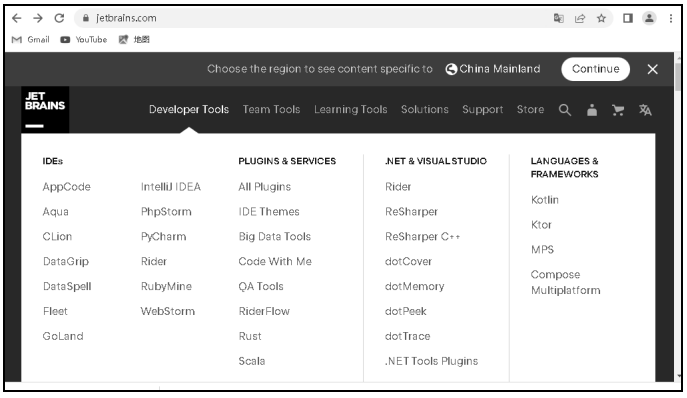

图 1.15　Developer Tools 页面

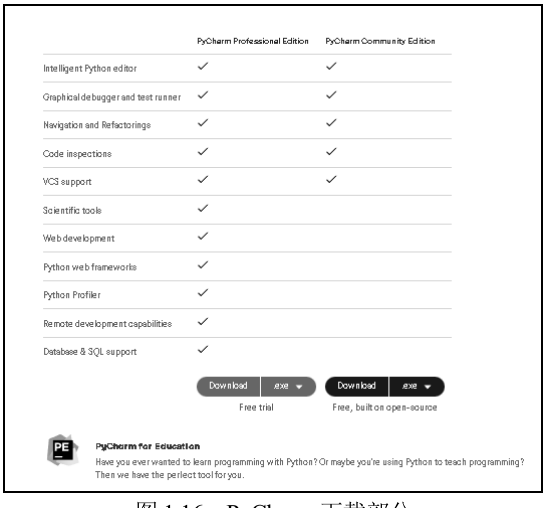

图 1.16　PyCharm 下载部分

(3) 双击安装包 pycharm-community-2022.3.exe，进入安装界面(见图 1.17)，按照提示进行

安装即可。目录可自行选择，初学时最好选择默认目录。

图 1.17　PyCharm 安装界面

2. PyCharm 的使用

(1) PyCharm 安装成功后，进入 Windows 10 操作系统开始菜单，单击 JetBrains 文件夹图标，再单击文件夹中的 PyCharm Community Edition 2022.3 启动 PyCharm 集成开发环境，如图 1.18 所示。首次启动时会弹出一个接受协议的界面，直接单击 Continue 按钮，即可进入 PyCharm 的开始界面，如图 1.19 所示。

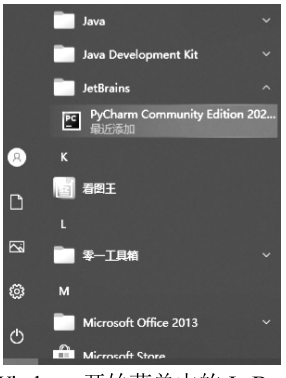

图 1.18　Windows 开始菜单中的 JetBrains 文件夹

图 1.19　PyCharm 的开始界面

(2) 在 PyCharm 的开始界面单击 New Project 按钮创建新工程，创建新工程界面如图 1.20 所示。新工程界面有 3 个选项，第一项是在 Location 输入框中选择工程文件夹，这里选择 D:\python\chapter01；第二项是在 Python Interpreter:New Virtualenv envrionment 下为新工程配置 Python 解释器，初学者选择默认模式即可；第三项是 Create a main.py welcome script，用于在新工程中创建显示欢迎文本的 Python 文件 main.py，一般不需要创建，因此不勾选这一项。单击 Create 按钮，即可进入 PyCharm 开发界面，如图 1.21 所示。

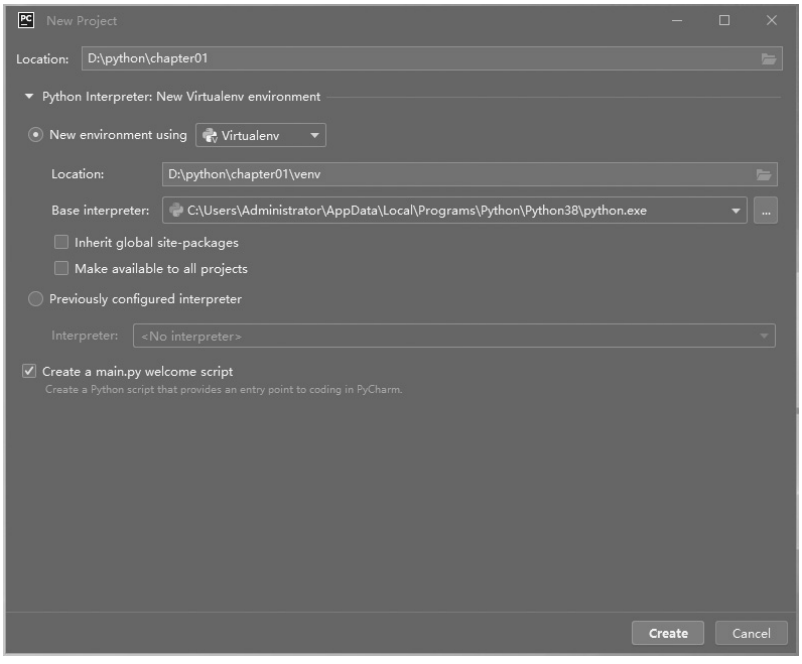

图 1.20　创建新工程界面

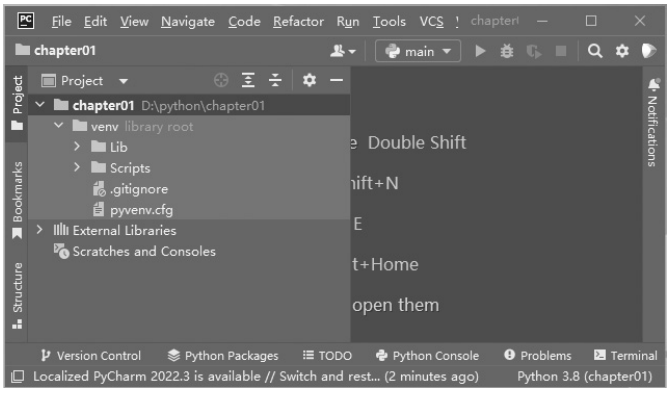

图 1.21　PyCharm 开发界面

(3) 在 PyCharm 开发界面的工程项目名称上右击，在弹出的快捷菜单中执行 New | Python File 命令(见图 1.22)，即可弹出 New Python file 对话框，如图 1.23 所示。

(4) 在 New Python file 对话框中输入文件名“HelloWorld”，按回车键，新的 Python 文件 HelloWorld.py 就创建好了。输入 Python 语句“print('Hello,World! ')”，只有一个输出语句的 Python 程序就编辑好了，如图 1.24 所示。

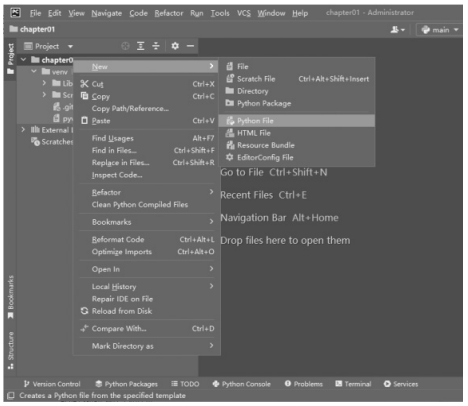

图 1.22　新建 Python 文件界面

图 1.23　New Python file 对话框

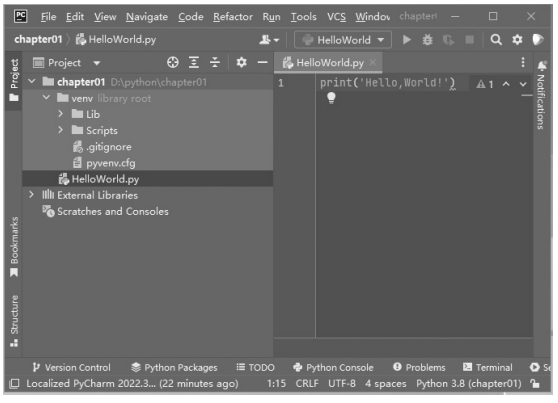

图 1.24　编辑 Python 文件

(5) 单击 Run 菜单，选择运行 HelloWorld 文件，运行显示窗口输出了字符串“Hello,World!”，如图 1.25 所示。PyCharm 集成开发环境的第一个 Python 程序运行成功。

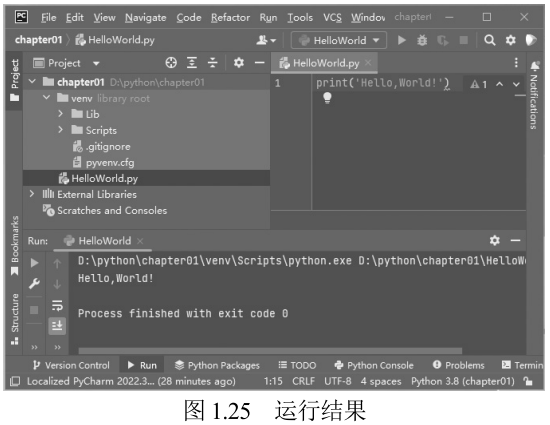

图 1.25　运行结果

1.2.5 Anaconda 集成开发环境

Anaconda 是一个开源的 Python 发行版本，包含了 conda、Python 等 180 多个科学包及其依赖项。Anaconda 集成了 PyCharm、Spider 和 Jupyter Notebook 等开发系统，安装 Anaconda 可以很方便地管理 Python 相关的包、切换不同的环境。

1. Anaconda 的下载与安装

(1) Anaconda 的官方网址是 https://www.anaconda.com/，进入官网主页，如图 1.26 所示。单击 Download 按钮进行下载，下载完成后，在浏览器的默认下载目录中有一个可执行的安装包，文件名是 Anaconda3-2022.10-Windows-x86_64.exe。

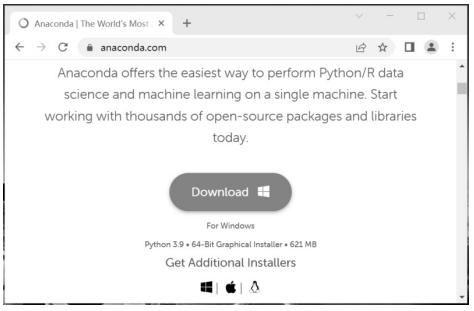

图 1.26 Anaconda 官网主页

(2) 双击安装包 Anaconda3-2022.10-Windows-x86_64.exe，进入安装界面(见图 1.27)，按照提示进行安装即可。目录可自行选择，初学时最好选择默认目录。

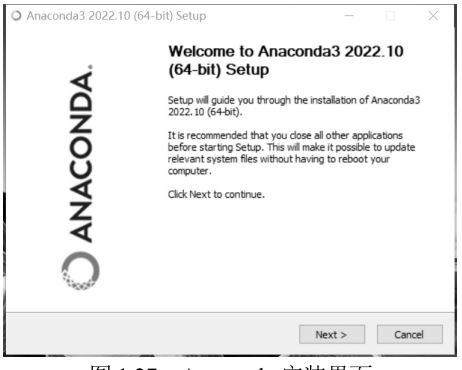

图 1.27 Anaconda 安装界面

2. Anaconda 的使用

Anaconda 安装成功后，进入 Windows 10 操作系统开始菜单，单击 Anaconda3(64-bit)文件夹图标，可以看到该文件夹中有 6 个执行文件选项，如图 1.28 所示。第一项 Anaconda Navigator(anaconda3)是进入 Anaconda 图形界面的命令文件。第二项 Anaconda Powershell Prompt(anaconda3)和第三项 Anaconda Prompt(anaconda3)都是 Anaconda 的命令行工作模式，都可以进行包安装、环境管理等，只是第二项的命令多一些而已。第四项 Jupyter Notebook(anaconda3)是数据分析使用的主要开发系统，后面会详细介绍。第五项和第六项是关于 Spyder 开发系统的，暂不使用。下面简单介绍 Anaconda 的 Jupyter Notebook，以便开发 Python 程序。

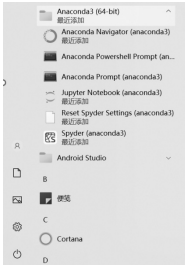

图 1.28　Windows 开始菜单中的 Anaconda3(64-bit)文件夹

(1) 在 Windows 开始菜单中的 Anaconda3(64-bit)文件夹下选择第一项，进入 Anaconda 图形界面(见图 1.29)，然后单击 Jupyter 图标按钮，进入 Jupyter Notebook 开发环境。或者选择 Anaconda3(64-bit)文件夹中的第四项直接进入 Jupyter Notebook 开发环境。

图 1.29　Anaconda 图形界面

(2) 进入 Jupyter Notebook 后，主界面如图 1.30 所示。单击右侧的 New 按钮，选择 Python 3，进入编辑界面，如图 1.31 所示。

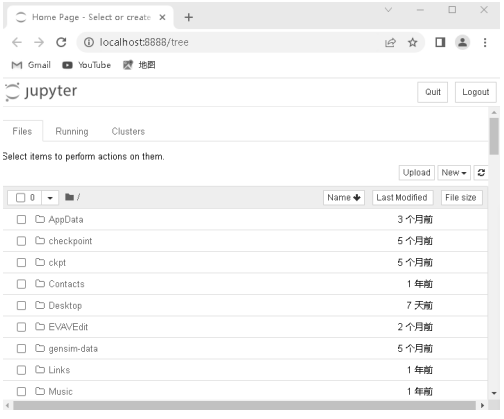

图 1.30　Jupyter Notebook 主界面

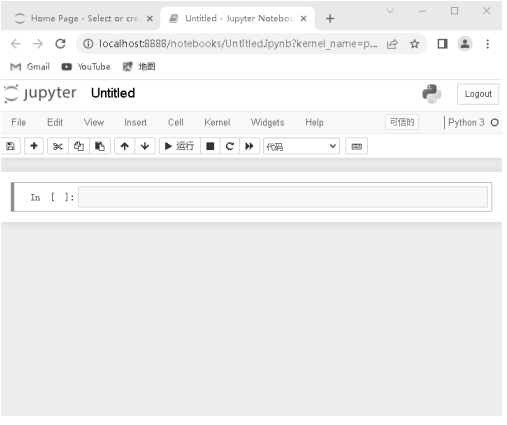

图 1.31　编辑界面

(3) 在编辑框中输入 print("Hello, jupyter!")语句，单击“运行”按钮，程序立即执行并在下方输出结果，如图 1.32 所示。单击 File 菜单，可以保存文件，文件的后缀是.ipynb，也可以保存为.py、.html 等格式的文件。Jupyter Notebook 开发环境兼有交互命令模式和文件模式的优点，不仅可以反复修改编辑框中的程序，还可以立即查看执行结果，十分方便，在数据分析中得到了广泛使用。

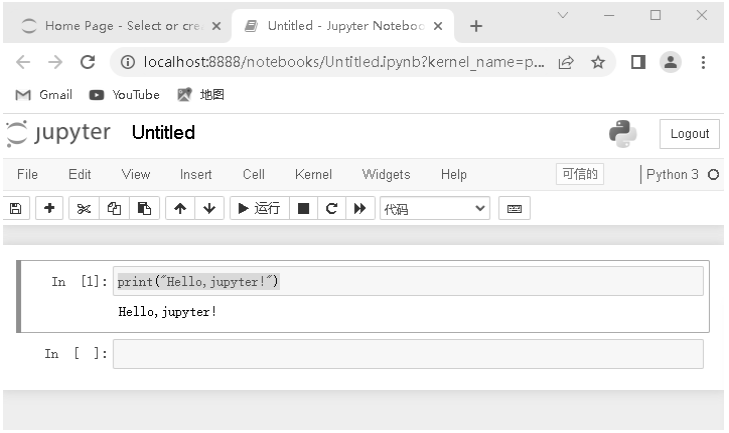

图 1.32　编辑与运行程序

1.2.6　Web 版在线开发环境

除了本地安装开发环境，还可以直接使用基于浏览器的解释器和集成开发环境。在浏览器中编写和运行 Python 代码，可即时获得运行结果，无须安装任何软件。常用的基于 Web 的 Python 开发环境有：Online Python、Online GDB、Python Anywhere、Programmiz、Google Colab。大家可以选择一款喜欢的进行学习、使用。对于初学者，推荐使用 Online Python，其主页如图 1.33 所示。

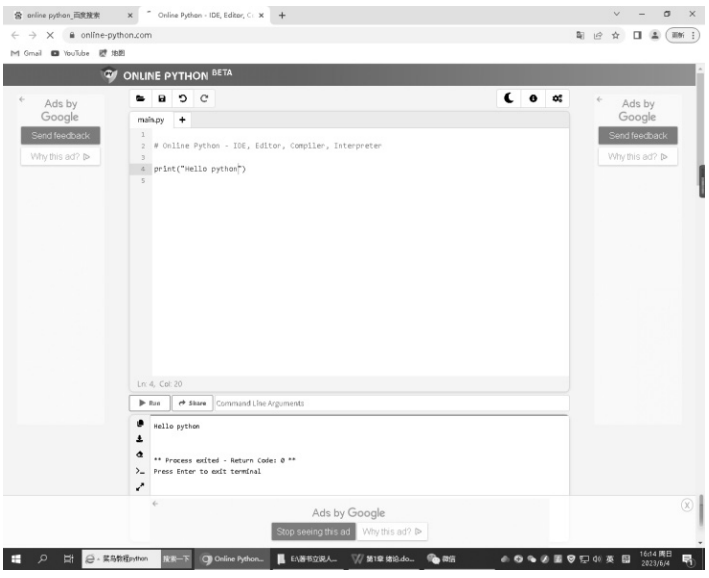

图 1.33　Online Python 主页

1.3　模块、包与库

Python 代码通常存储在一个后缀为.py 的文件中，因此 Python 代码的基本组织单位是文件，也称为模块(module)。一个 Python 项目的程序往往存储在多个文件中，即一个项目由多个模块

组成。Python 程序设计遵循标准的模块化编程理念，Python 程序由模块组成，模块由语句组成。

当一个项目中有多个模块(多个.py 文件)时，需要再进行组织，将功能类似的模块放到一个文件夹中就形成了包。Python 包(package)就是一个包含__init__.py 文件的文件夹。包可以包含模块，也可以包含子包(subpackage)，就像文件夹可以包含文件和子文件夹一样。

Python 库(library)借用其他编程语言的概念，没有特别具体的定义。库强调的是功能性，而不是代码组织。通常，我们将实现某一功能的包或模块的集合称为库。

Python 模块可分为内建模块、标准模块和第三方模块。内建模块是在运行解释器时自动导入的模块，可以直接使用模块中的变量、函数等。标准模块是在安装解释器时自动安装到硬盘上的模块，但在运行解释器时并不会自动导入，需要用 import 语句将其导入才可以使用。第三方模块是在安装解释器时没有安装到本地的模块，使用前需要下载并安装。自己建立的.py 文件称为自定义模块，文件名称就是模块名称。

1.3.1　模块的安装

Python 模块的安装有在线安装和离线安装两种方式。

1. 在线安装

在线安装是比较常用的安装方法。利用 Python 内置的 pip 工具可以非常方便地安装 Python 第三方模块。进入 CMD 窗口(操作系统命令行窗口)，使用如下命令进行安装。

```
pip　install　模块名
```

如果不再使用已安装的模块，则可以卸载该模块，其语法格式如下。

```
pip　uninstall　模块名
```

2. 离线安装

第三方库(模块)一般是打包压缩上传到网上的，因此，离线安装需要先把库下载到本地，然后把包解压，在解压后的文件夹中，找到该安装包中的 setup.py 文件，最后进入 CMD 窗口，使用如下命令进行安装。

```
python　setup.py　install
```

当第三方库、包或模块被安装到本地之后，其使用方法与标准库中模块的使用方法一样。

1.3.2　模块的导入与使用

模块化设计的优点之一就是“代码复用性高”。写好的模块可以被反复调用、使用。模块的导入就是“在本模块中使用其他模块”。

1. 模块的导入

模块的导入有两种方式：import 语句导入和 from…import 语句导入。模块就是.py 类型的 Python 文件，在导入时不需要加.py 后缀，直接导入文件名即可。

(1) import 语句导入。

import 语句的基本语法格式如下。

```
import 模块 1,模块 2...                    # 导入多个模块
import 模块名 as 模块别名                  # 导入模块并使用新名字
```

(2) from…import 语句导入。

Python 中可以使用 from…import 导入模块中的成员，基本语法格式如下。

```
from 模块名 import 成员 1,成员 2...        # 导入模块中的多个成员
from 模块名 import 成员 as 成员别名        # 导入模块中的成员并使用新名字
from 模块名 import *
```

上式中的*表示导入模块中所有的名字不是以下画线(_)开头的成员(成员是指模块中的类、函数等)。一般情况下尽量避免使用这种格式，因为导入的名字很有可能会覆盖之前已经定义的名字，而且可读性极差。

2. 包的导入

包就是包含很多模块的文件夹，包内还可以有子包。利用 import 语句可以直接导入包(本质是执行包中的__init__.py 文件，把包也转换成了模块，所以仅导入__init__.py 中的内容)，语法格式如下。

```
import 包名
```

直接导入一个包，就可以使用包中__init__.py 文件中的全部内容。

也可以导入包中的某一个模块，语法格式如下。

```
import 包名.模块名
```

还可以导入包中指定模块的某一个类或函数，语法格式如下。

```
import 包名.模块名.类名(或函数名)
```

3. 模块的使用

(1) 内建模块。

内建模块是在运行解释器时自动导入的模块，可以直接使用模块中的变量、函数等。在 Python 中，有一个内建模块(builtins)，该模块在 Python 启动后会自动加载到内存，可以直接使用内建模块中的函数或其他功能。例如，内建模块中有一个 abs()函数，其功能是计算一个数的绝对值，解释器命令行执行 abs(−20)，将返回 20，如图 1.34 所示。

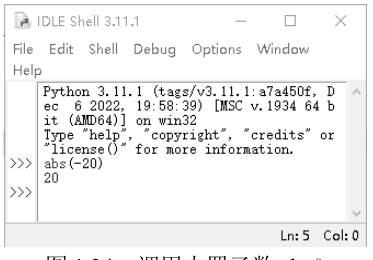

图 1.34　调用内置函数 abs()

(2) 标准模块。

标准模块是在安装解释器时自动安装到硬盘上的模块，但在运行解释器时并不会自动导入，

需要用 import 语句将其导入才可以使用。例如，标准模块中的数学模块 math，要将其导入以后才能使用，示例代码如程序段 P1.1 所示。在 Python 中，有许多标准模块，随着 Python 的发展，标准模块也在发展。

```
P1.1 导入 math 模块
import  math
print(math.pi)
print(math.sqrt(9))
```

运行代码，输出结果如下。(注意：本书把所有竖列的输出结果都用横排表示)

```
3.141592653589793        3.0
```

导入模块后，在使用时要加上模块名。例如，使用 math.pi 输出圆周率；使用 math.sqrt(9) 计算 9 的平方根，计算后输出 3。

(3) 第三方模块。

第三方模块需要先安装、导入，然后才能使用，如数值计算模块 numpy。以在线安装为例，首先在操作系统命令行输入以下语句。

```
pip install numpy
```

执行命令后，显示结果如图 1.35 所示。

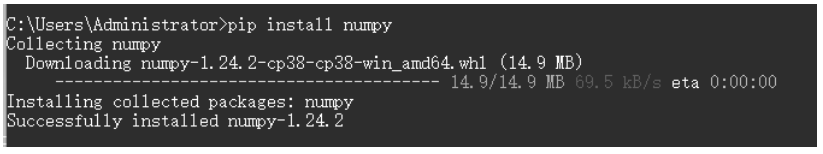

图 1.35　安装第三方模块 numpy

显示安装成功后，与标准模块一样在程序中导入使用。示例代码如程序段 P1.2 所示。

```
P1.2 导入 numpy 模块
import numpy as np
print(np.pi)
x=np.array([1,2,3])
print(x)
```

运行代码，输出结果如下。

```
3.141592653589793        [1 2 3]
```

在导入 numpy 模块时，给它取了一个别名 np，用 np.pi 输出了圆周率，用 np.array([1,2,3]) 定义了一个数组[1,2,3]。

Python 是一个开放的生态系统。开放系统的重要特征是每个开发者都有权编辑和发布模块(或包)，人人都可以为这个系统做出贡献。Python 拥有丰富的标准库和第三方库，Python 的第三方库会在 PyPI 网站(https://pypi.org/)上发布，在安装某个第三方库之前，建议先到 PyPI 官方网站找到该库，了解其基本情况，特别是它支持的 Python 版本，以及最新版本的发布时间。当然很多开发者也使用 GitHub 网站的源码安装。

1.4 turtle 对象绘图库

1.4.1 turtle 对象编程思维

对象这个概念，我们并不陌生。现实世界中，我们把感知到的一切有形、无形的事物都称为对象。例如，大家手里的一本书、校园里的一棵树、动物园里的一只老虎都可以称为一个对象，这些是有形的对象。同时，“书”这个概念表示世界上所有的书籍(提到这个概念，你自然会想到所有的书类)；“树”这个概念表示地球上所有的树木；“老虎”这个概念表示地球上所有的虎类动物，这些都是现实世界中人类创造的概念事物，可以称为概念对象，属于无形的对象。这是人类的自然思维方式。

我们每天都会使用各种不同的软件，软件几乎成了我们生活的必需品。可以说，大部分软件都是采用面向对象编程技术开发的基于对象的软件系统，人类创造了一个软件对象世界，这个世界由信息世界、网络空间、虚拟世界和未来的元宇宙等组成。就像分子、原子是现实世界的基本事物一样，软件世界中的对象是软件世界中的基本事物。那么，软件世界中的对象是什么呢？

这个问题其实很简单，软件世界中的对象就是现实世界中对象的映射。现实世界由现实对象构成，软件世界由软件对象构成。一切皆对象，Python 语言充分实现了这个哲学认识。因此，在现实世界中我们用自然语言表达的对象，在软件世界中可以很容易地用 Python 语言描述出来，因为所采用的思维方式是一样的，即人类的自然思维方式。这就是 Python 易学、易用且功能强大的根本。

人类认识事物所创造的概念对象，实际上是通过思维活动对现实事物进行模式识别，再进行归纳、抽象而建立的分类模型概念，并用语言符号进行命名，这就是类型对象，简称为类对象。例如，书、树、虎就是现实世界中的类对象，而具体的一本书、一棵树、一只老虎就是实例对象。任何对象都有自己的属性特征和功能行为，因此，通过自然语言描述对象的属性特征和功能行为，我们就可以识别对象。

Python 语言如何描述对象呢？Python 语言采用 class、def、self 等关键词加上语言符号来描述对象。用 Python 语言和中文字符描述海龟类对象的简单模型如程序段 P1.3 所示。

```
P1.3  用 Python 语言和中文字符描述的海龟类
class 海龟:                          # 用 class 关键词给海龟类模型命名
   颜色= "黑色"                      # 描述海龟的颜色属性
   腿数= 4                           # 描述海龟的腿数属性
   def 前进(self):                   # 用 def 关键词给海龟前进方法命名，self 关键字表示方法所属的对象
        print("前进 10 米")          # 方法体描述前进方法要做的事情，这里只输出“前进 10 米”
   def 后退(self):
        print("后退 8 米")
   def 左转(self):
        print("左转 5 度")
   def 右转(self):
        print("右转 3 度")
小海= 海龟()                         # 用海龟类模型创建一个海龟实例，并命名为小海
print(小海.颜色)                     # 查看小海的颜色属性
print(小海.腿数)
小海.前进()                          # 调用海龟的前进方法
```

```
小海.后退()
小海.左转()
小海.右转()
```

运行代码，输出结果如下。

```
黑色    4    前进 10 米    后退 8 米    左转 5 度    右转 3 度
```

从上面对海龟类模型的描述中可以看到，Python 语言通过定义变量描述对象的属性特征，这里称为属性；通过 def 定义的函数描述对象的功能行为，在类模型中，定义的函数的第一个参数必须是表示自身对象的 self 参数，因此，在类对象中定义的函数称为对象的方法。对象的属性和方法，称为对象的成员。

用 Python 语言和英文字符描述海龟类对象的模型如程序段 P1.4 所示。

```
P1.4 用 Python 语言和英文字符描述的海龟类
class turtle:
    color = "black"
    legs = 4
    def forward(self):
        print("Forward 10 meters")
    def backward(self):
        print("Backward 8 meters")
    def left(self):
        print("Turn left 5 degrees")
    def right(self):
        print("Turn right 3 degrees")
xiaohai = turtle()
print(xiaohai.color)
print(xiaohai.legs)
xiaohai.forward()
xiaohai.backward()
xiaohai.left()
xiaohai.right()
```

运行代码，输出结果如下。

```
black          4          Forward 10 meters          Backward 8 meters
Turn left 5 degrees          Turn right 3 degrees
```

在上面的模型中，class 定义类对象的名称 turtle，def 定义类对象的行为方法，self 作为行为方法的第一个参数，表示使用这个行为方法的对象自身。

与自然语言描述类对象一样，用 Python 语言描述类对象实际上就是用 Python 语言对现实世界中的对象建模，构建一个现实世界中的对象映射到软件世界的模型。这个模型就是软件世界中的类对象。利用这个模型可以创建软件世界中的一个个实例对象，这些实例对象是现实世界中一个个具体实例对象在软件世界中的映射。例如，上面程序中的海龟(turtle)是类对象，映射现实世界中的海龟类概念；创建的小海(xiaohai)是一个实例对象，映射现实世界中的某个具体的海龟。

Python 的 turtle 绘图库，就是由一系列软件对象构成的软件系统。

1.4.2 turtle 库概述

turtle 是 Python 的一个编程教育类库，广受教育者的喜爱。许多智能设计与创意编程比赛均使用 turtle 库。turtle 库也称为海龟绘图库，它模拟一只小海龟在爬行，爬行轨迹就是绘制出来的图形。

在 turtle 中，小海龟是绘图的画笔(pen)对象，记录小海龟爬行轨迹的是画布(canvas)对象。画笔对象的形状、颜色和大小等特征，称为画笔对象的属性；画笔对象的前进、后退、左转和右转等动作，称为画笔对象的方法。在绘图编程中，根据绘图要求设置画笔属性、调用画笔方法，就可以绘制出需要的图形。turtle 绘图就是采用这种面向对象的编程思维，对画布对象和画笔对象等进行编程来实现绘图功能的。只是这里的对象都是 turtle 模块已经设计好的，而在解决其他问题的编程实践中，其对象往往需要自己设计。

turtle 是 Python 的标准图形模块，当通过 import 语句导入 turtle 时，就创建了一个 turtle 对象(模块对象)，其中包括屏幕窗口对象、画布对象、画笔对象等。在屏幕坐标系中，turtle 绘图窗口及画布示意图如图 1.36 所示。

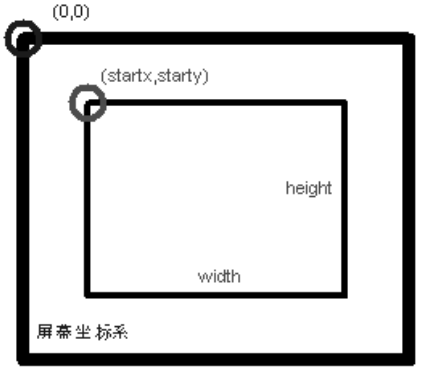

图 1.36　turtle 绘图窗口及画布示意图

绘图窗口及画布属性如表 1-1 所示。

表 1-1　绘图窗口及画布属性

属性名	功能描述
width	绘图窗口宽
height	绘图窗口高
startx	绘图窗口左上角顶点 *x* 坐标
starty	绘图窗口左上角顶点 *y* 坐标
canvwidth	画布宽
canvheigh	画布高
bg	画布背景颜色

画笔属性如表 1-2 所示。

表 1-2　画笔属性

属性名	功能描述
shape	画笔形状
pensize	画笔线条的宽度
colormode	画笔颜色模式
pencolor	画笔颜色
speed	画笔移动速度

turtle 绘图是通过控制海龟(画笔)对象在创建的绘图窗口(画布对象)上移动来实现图形绘制的。海龟在绘图窗口的移动以两个坐标系为参考。

- 空间坐标系：空间坐标系以绘图窗口画布的中心点为坐标系原点，原点的绝对坐标为(0,0)。在图 1.37 中，第一至第四象限四点的绝对坐标分别为(100,100)、(−100,100)、(−100, −100)、(100, −100)。默认状态下海龟以水平向右为前进方向。
- 角度坐标系：角度坐标系是以海龟为中心的相对坐标系，顺时针方向一周为 0～−360°，逆时针方向一周为 0～360°，如图 1.38 所示。

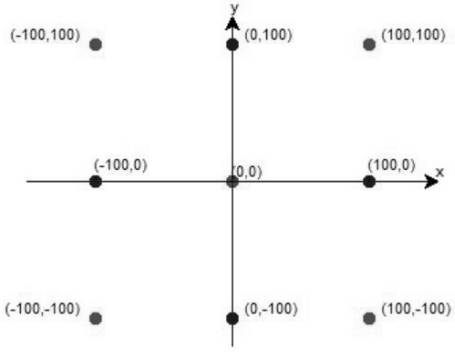

图 1.37　空间坐标系

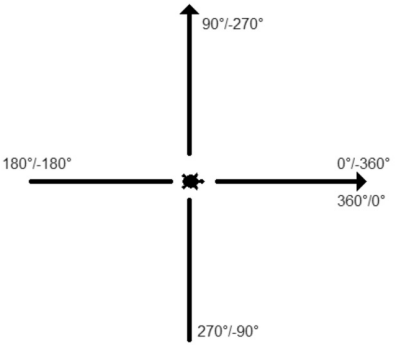

图 1.38　角度坐标系

以两个坐标系为参考，控制海龟在画布上移动，它爬行的路径就是绘制的图形。turtle 绘图对象常用的控制方法如表 1-3 所示。

表 1-3　turtle 绘图对象常用的控制方法

方法名	功能描述
设置窗体、画布	
turtle.setup(width,height,startx,starty)	创建设置窗体的宽、高、坐标
turtle.screensize(canvwidth,canvheight,bg)	创建设置画布的宽、高、背景色
turtle.title("s")	设置窗体标题
turtle.bgcolor()	设置背景颜色
turtle.bgpic()	设置背景图片
screen_name = turtle.Screen()	创建窗体对象并取名 screen_name
canvas_name = turtle.Canvas()	创建画布对象并取名 canvas_name

(续表)

方法名	功能描述
设置画笔	
turtle.shape("turtle")	设置画笔的形状(turtle-海龟；arrow-箭头；circle-圆圈；square-实心正方形；triangle-三角形；classic-默认箭头)
turtle.pensize(width)/width()	设置当前画笔线条的宽度为 width 像素
turtle.colormode(1.0[255])	设置画笔颜色模式
turtle.pencolor(colorstring)/color()	设置画笔的颜色，参数 colorstring 可以是"green"、"red"、"blue"、"yellow"等英文字符串
turtle.speed(5)	设置画笔的移动速度，取值范围为[0,10]的整数，数字越大，画笔移动的速度越快
pen_name=turtle.Pen()	创建画笔对象并取名 pen_name
控制画笔	
turtle.penup()/pu()/up()	提起画笔，不绘图
turtle.pendown()/pd()/down()	画笔移动时绘制图形
turtle.forward(100)/fd(100)	画笔向当前方向移动 100 像素距离
turtle.backward(100)/bk(100)	画笔向相反方向移动 100 像素距离
turtle.right(45)/rt(45)	画笔顺时针移动 45°
turtle.left(45)/lt(45)	画笔逆时针移动 45°
turtle.setheading(45)/seth(45)	设置当前画笔朝向为 45°
turtle.goto(x,y)	移动画笔到指定坐标位置
turtle.hideturtle()	隐藏画笔 turtle 形状
turtle.showturtle()	显示画笔 turtle 形状
turtle.setx(x)	海龟的 *x* 坐标移动到指定位置
turtle.sety(y)	海龟的 *y* 坐标移动到指定位置
turtle.home()	设置当前画笔位置为原点，朝向东(默认值)
图形与填充	
turtle.circle(5,[extent,steps])	绘制半径为 5 的圆形
turtle.dot()	画一个圆点(实心)
turtle.stamp()	复制当前画笔图形
turtle.fillcolor('red')	设置填充颜色
turtle.color(pencolor,fillcolor)	同时设置画笔颜色(边框颜色)和填充颜色
turtle.begin_fill()	以当前为起点，开始填充颜色
turtle.end_fill()	以当前为终点，结束填充图形
自定义画笔形状	
begin_poly()	开始记录图形
end_poly()	结束图形记录

(续表)

方法名	功能描述
get_poly()	获取画笔图形
addshape('name',shape)	增加画笔形状
register_shape('name', shape)	注册画笔形状
register_shape("file.gif")	注册 gif 格式图片为画笔形状
全局控制	
turtle.done()	绘图结束后，保留窗口。必须是最后一个语句
turtle.clear()	清空 turtle 窗口，但是 turtle 的位置和状态不会改变
turtle.reset()	清空 turtle 窗口，重置 turtle 状态为起始状态
turtle.undo()	撤销上一个 turtle 动作
turtle.isvisible()	返回当前 turtle 是否可见
turtle.write("文本" ,align="center",font=("微软雅黑",20,"normal"))	写文本。align(可选)：left、right、center；font(可选)：字体名称、字体大小、字体类型
屏幕事件	
turtle.mainloop()	启动事件循环
turtle.listen()	监听屏幕事件
turtle.onkeypress()	当键按下
turtle.onkeyrelease()	当键按下并释放
turtle.onclick(fun,btn=1,add=None)	当单击画布屏幕时，执行函数(fun 为传入的函数)，btn 值：1 为鼠标左键，2 为鼠标中间键，3 为鼠标右键。当 add 为 True 时将添加一个新绑定
turtle.ontimer(fun,t=0)	定时器，在达到 t 毫秒后，执行 fun 函数
动画控制	
turtle.delay()	延迟(毫秒)：连续两次画布刷新的间隔时间
turtle.tracer(n,delay)	追踪小海龟的绘图，当 *n* 为 0 或为 False 时，禁用追踪，默认为 1；delay 为延迟(毫秒)
turtle.update()	更新
保存图形	
image = pen.getscreen()	将屏幕转换为图形对象
image.getcanvas().postscript(file="demo.eps")	保存画布上的图形。仅能保存为 eps 矢量图格式，可用 GIMP 转换为其他格式

1.4.3　turtle 绘图操作方法

1. 创建并设置窗体、画布属性

窗体是绘图展示区域，画布是绘图区域，创建与设置窗体的语法格式如下。

```
turtle.setup(width=200,height=100,startx=100,starty=100)
```

参数说明如下。

- width：窗体宽，当为整数时，表示像素；当为小数时，表示占据计算机屏幕的比例。
- height：窗体高，当为整数时，表示像素；当为小数时，表示占据计算机屏幕的比例。
- startx、starty：表示矩形窗口左上角顶点的位置坐标，如果为空，则窗口位于屏幕中心。

如果程序中未调用 setup()函数，那么绘图程序执行时会生成一个默认窗口。在程序中绘制完图形后，应调用 done()函数声明绘制结束，此时 turtle 的主循环会终止，但直到用户手动关闭图形窗口时图形窗口才会退出。

创建与设置画布的语法格式如下。

```
turtle.screensize(canvwidth=400,canvheight=300,bg='black')
```

参数说明如下。

- canvwidth：画布宽，单位为像素。
- canvheight：画布高，单位为像素。
- bg：背景颜色。可以是"green"、"red"、"blue"、"yellow"等英文字符串。

设置窗体和画布的示例代码如程序段 P1.5 所示。

```
P1.5 设置窗体和画布
import turtle                                  # 导入 turtle 模块
turtle.setup(200,100,100,100)                  # 窗体尺寸为 200×100，左上角顶点坐标为(100, 100)
turtle.screensize(400,200,'green')             # 画布尺寸为 400×200，颜色为绿色
turtle.done()                                  # 声明绘制结束
```

运行代码，显示窗口与画布，如图 1.39 所示。

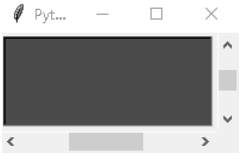

图 1.39　窗口与画布

设置的画布比窗体尺寸要大，因此会显示窗口滑动条。

2. 设置画笔属性

(1) 画笔形状。

使用 shape()方法设置画笔的形状，语法如下。

```
turtle.shape("turtle")
```

输入参数表示设置画笔的形状，可以设置以下 6 种形状的画笔：turtle-海龟；arrow-箭头；circle-圆圈；square-实心正方形；triangle-三角形；classic-默认箭头。

(2) 画笔粗细。

使用 width()方法设置画笔线条粗细，语法如下。

```
turtle.width(width)
```

参数 width 用于设置画笔线条的宽度，用像素表示。

(3) 画笔颜色模式。

使用 colormode()方法设置画笔颜色模式，语法如下。

```
turtle.colormode(1.0[255])
```

设置颜色的模式，如果参数 mode 的取值为 255，则采用的是 RGB 整数值模式；如果参数 mode 的取值为 1.0，则采用的是 RGB 小数值模式。颜色模式配色数值如表 1-4 所示。

表 1-4　颜色模式配色数值表

颜色	RGB 整数值	RGB 小数值	十六进制字符串
白色	(255,255,255)	(1,1,1)	#FFFFFF
黑色	(0,0,0)	(0,0,0)	#000000
红色	(255,0,0)	(1,0,0)	#FF0000
绿色	(0,255,0)	(0,1,0)	#00FF00
蓝色	(0,0,255)	(0,0,1)	#0000FF
黄色	(255,255,0)	(1,1,0)	#FFFF00
紫色	(160,32,240)	(0.63,0.13,0.94)	#A020F0

(4) 画笔颜色。

使用 color()方法设置画笔颜色，语法如下。

```
turtle.color(colorstring)
```

设置画笔的颜色，参数 colorstring 可以是"green"、"red"、"blue"、"yellow"等英文字符串，也可以是表 1-4 中的配色数值。

(5) 画笔速度。

使用 speed()方法设置画笔运动速度，语法如下。

```
turtle.speed(num)
```

参数 num 用于设置画笔移动的速度，其取值范围为[0,10]的整数，数字越大，速度越快。

设置画笔属性示例代码如程序段 P1.6 所示。

```
P1.6 设置画笔属性
import turtle as pen
pen.setup(800,600,100,100)
pen.screensize(800,600,'white')
pen.shape("turtle")                    # 画笔形状，turtle-海龟
pen.width(10)                          # 画笔线条宽度 10 像素
pen.color('red')                       # 红色画笔 red
pen.speed(5)                           # 画笔速度 5
pen.forward(100)                       # 画笔前进 100 像素
pen.done()
```

运行代码，显示结果如图 1.40 所示。

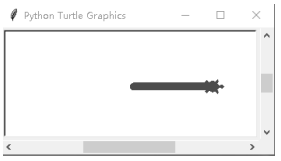

图 1.40　显示结果

3. 控制画笔

(1) 提起和放下。

正如在纸上绘图一样，turtle 中的画笔分为提起(up)和放下(down)两种状态。只有当画笔为放下状态时，移动画笔，画布上才会留下痕迹。turtle 中的画笔默认为放下状态，使用 penup()/pu()/up()方法可以提起画笔，使用 pendown()/pd()/down()方法可以放下画笔。

(2) 移动画笔。

当画笔状态为 down 时，通过移动画笔可以在画布上绘制图形。此时，可以将画笔想象成一只海龟(这也是 turtle 模块名字的由来)，当海龟落在画布上时，它可以向前、向后、向左、向右移动，海龟在爬动时可以在画布上留下痕迹，路径即为所绘图形。移动画笔的方法如下。

- forward(distance)方法用于向前移动画笔，参数 distance 为距离，单位为像素。
- backward(distance)方法用于向后移动画笔，参数 distance 为距离，单位为像素。
- goto(x,y)方法用于将画笔移动到画布上的指定位置，该函数可以使用参数 x、y 分别接收表示目标位置的横坐标和纵坐标，也可以仅接收一个表示坐标向量的参数。

(3) 转动画笔。

转动画笔的方法如下。

- right(angle)方法用于指定画笔向右转，参数 angle 为转动的角度，单位是度。
- left(angle)方法用于指定画笔向左转，参数 angle 为转动的角度，单位是度。
- setheading(angle)/seth(angle)方法用于设置画笔在坐标系中的角度。参数 angle 以 x 轴正向为 0°，以逆时针方向为正，角度从 0°逐渐增大；以顺时针方向为负，角度从 0°逐渐减小。

若要使画笔向左或向右移动某段距离，应先调整画笔角度，再使用移动画笔的方法。

移动与转动画笔的示例代码如程序段 P1.7 所示。

```
P1.7 移动与转动画笔
import turtle
turtle.goto(-100, -100)
turtle.forward(100)
turtle.left(45)
turtle.forward(100)
turtle.right(45)
turtle.backward(50)
turtle.left(90)
turtle.forward(100)
turtle.setheading(45)
turtle.forward(20)
turtle.done()
```

运行代码，显示结果如图 1.41 所示。

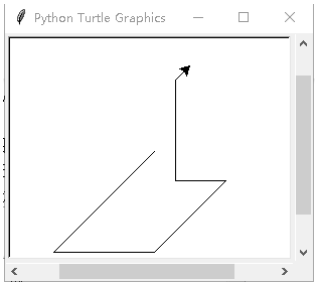

图 1.41　显示结果

4. 图形与填充

(1) 圆形绘制。

turtle 模块中提供了 circle()方法，用于绘制以当前坐标为圆心，以指定像素值为半径的圆或弧。语法格式如下。

```
turtle.circle(radius,[extent,steps])
```

参数说明如下。

- radius：用于设置半径。当 radius 为正时，画笔以原点为起点向上绘制弧线；当 radius 为负时，画笔以原点为起点向下绘制弧线。
- extent：用于设置弧的角度。当 extent 为正时，画笔以原点为起点向右绘制弧线；当 extent 为负时，画笔以原点为起点向左绘制弧线；当 extent 为默认值 None 时，绘制整个圆。
- steps：用于设置步长。circle()方法可用于绘制正多边形(圆由近似正多边形来描述)。

(2) 图形填充。

turtle 模块中提供了 fillcolor()函数用于设置填充颜色，使用 begin_fill()函数和 end_fill()函数可以填充图形，实现“面”的绘制。

例如，先绘制半径为 90/−90 像素、角度为 60°/−60° 的弧线组合的四段弧线；然后把中心点移到(0,100)，当参数 extent 为默认值 None 时，绘制整个圆；最后把中心点移到(0,−120)，参数 steps 用于设置步长，circle()方法用于绘制正多边形，并把圆形填充为红色。示例代码如程序段 P1.8 所示。

```
P1.8 图形与填充
import turtle
turtle.circle(90,60)
turtle.home()
turtle.circle(90,−60)
turtle.home()
turtle.circle(−90,60)
turtle.home()
turtle.circle(−90, −60)
turtle.home()
turtle.goto(0,50)
turtle.fillcolor('red')
turtle.begin_fill()
turtle.circle(50)
turtle.end_fill()
```

```
turtle.goto(0, -120)
turtle.circle(50,steps=6)
turtle.done()
```

运行代码，显示结果如图 1.42 所示。

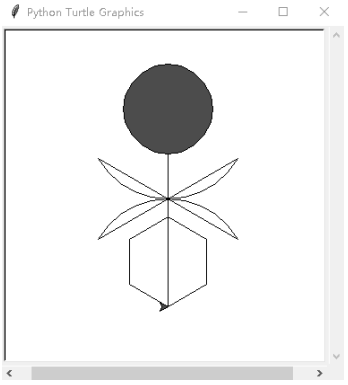

图 1.42　显示结果

通过对海龟对象绘图程序的使用可以发现，面向对象编写的程序逻辑十分清晰，可以通过对象完成程序功能。若要发挥对象的功能，就要设置好对象属性、使用好对象方法。了解了对象体系，也就熟悉了程序。

合理利用 turtle 模块，可绘制简单有趣的图形，也可结合逻辑代码生成可视化图表。此外，turtle 模块中还定义了实现更多功能的方法，如制作动画的方法等。有兴趣的读者可自行查阅 Python 官方文档进行深入学习。

实训与习题

实训

(1) 完成本章 P1.1～P1.8 程序上机练习。

(2) 熟练掌握一款集成开发环境(IDLE、PyCharm、Anaconda、Visual Studio Code)。

(3) 熟悉基于 Web 的 Python 开发环境(Online Python、Online GDB、Python Anywhere)。

(4) 导入标准库数学模块 math，用 sin()函数计算三角函数值。

(5) 导入标准库随机数模块 random，用 randint()函数生成随机整数。

(6) 导入标准库时间模块 time，调用 time()、localtime()、ctime()函数输出时间，调用 sleep(x)函数让计算机休眠 x 秒。

(7) 使用第三方库 numpy 编程：计算三角函数值。

习题

1. 填空题

(1) Python 文件的扩展名为_____。

(2) Python 是面向_____的高级程序设计语言。
(3) Python 模块的本质是_____文件。
(4) 在当前程序中导入模块，使用的关键字是_____。
(5) Python 可以在多种平台上运行，这是其_____的特点。

2. 选择题

(1) 下列选项中，不改变绘制方向的 turtle 命令是(　　)。
A. turtle.fd()　　B. turtle.seth()　　C. turtle.right()　　D. turtle.circle()
(2) Python 程序在 Windows 中的扩展名是(　　)。
A. exe　　B. py　　C. jpg　　D. docx
(3) 下列选项中，不属于计算机高级语言的是(　　)。
A. Python　　B. Java　　C. 机器语言　　D. JavaScript
(4) 下列选项中，不属于 Python 特性的是(　　)。
A. 开源免费　　B. 平台无关　　C. 可扩展　　D. 属于低级语言
(5) 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

3. 判断题

(1) Python 语言可以用面向对象的方法编程。(　　)
(2) Python 是一门跨平台、开源、免费的动态编程语言。(　　)
(3) 在同一台计算机上不可以安装多个不同的 Python 版本。(　　)
(4) 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。(　　)
(5) PyCharm 是 Python 的集成开发环境。(　　)

4. 简答题

(1) 简述 Python 的两种运行模式。
(2) 浏览 Python 主页，找出与我们所用的计算机相匹配的版本。
(3) 简述 Python 的特点与应用领域。
(4) 简述 Python 模块、包和库。
(5) 简述 turtle 程序面向对象特性。

5. 编程题

(1) 利用 turtle 库中的 fd()方法和 seth()方法绘制一个等边三角形，边长为 200。
(2) 利用 turtle 库中的 circle()方法绘制一个圆、一个半圆和一段 90° 的圆弧，自定义圆和弧的属性。
(3) 利用 turtle 库绘制直方图，宽度为 20，间距为 30，高度分别为 130、180、150。
(4) 编写一个模块，把 turtle 模块的主要绘图方法用中文实现，然后导入所编写的模块，用中文方法绘制正方形。
(5) 导入上题的中文模块，绘制一个正十边形。

第 2 章

Python基础

学习目标：

1. 掌握 Python 代码格式与风格
2. 掌握 Python 标识符与关键字
3. 掌握 Python 常量、变量与解释器命名空间
4. 熟悉内置函数、类与对象的概念及应用
5. 熟悉模块与文件对象的概念及应用

思政内涵：

通过学习和遵守命名规则与编程规范，广大学子要养成遵守规则，工作严谨、认真负责的职业精神。

2.1 基础语法

2.1.1 代码格式

良好的代码格式可提升代码的可读性。Python 代码的格式是 Python 语法的一部分，本节从缩进、换行和注释 3 个方面对 Python 代码格式进行讲解。

1. 缩进

一般使用“缩进”(即一行代码之前的空白区域)确定 Python 代码之间的逻辑关系和层次关系。Python 代码的缩进可以通过 Tab 键或空格键控制，但不允许混合使用 Tab 键和空格键。输入空格是 Python 3 首选的缩进方法，一般使用 4 个空格表示一级缩进。示例代码如程序段 P2.1 所示。

```
P2.1 代码缩进
if True:
        print("True")
else:
        print("False")
```