高等院校计算机应用系列教材

Python程序设计基础与应用

孙福权　余浩谦　孙梦迪　主　编
唐四元　刘　明　郝政鑫　崔建江　副主编

清华大学出版社
北　京

内 容 简 介

本书提供系统、全面的python入门指导，通过短小精悍的基础示例呈现语法的作用和使用要点。本书减少传统教材中晦涩的底层原理与复杂语法细节的讲解比重，从Python语言简介与环境搭建入手，逐步覆盖语法基础、程序控制结构、组合数据类型、函数定义与使用等核心知识，再延伸至模块与库、文件处理、网络爬虫、数据分析与可视化等应用技能，每个章节都配有清晰的示例代码、详细的步骤说明与针对性习题。

本书由浅入深、层层递进，不仅注重理论知识的讲解，还通过实际案例和项目实践帮助学生将理论应用于实际问题。在内容编排上遵循认知规律，从基础概念到高级应用层层递进，采用通俗易懂的语言，避免复杂的公式和术语，力求让没有技术基础的学生也能轻松理解python编程思维和应用场景。

本书旨在为高等院校的学生提供一本通俗易懂、内容全面的python编程入门教材，适用于非工科类专业作为公共计算机教学用书，也可作为非计算机专业学生的程序设计课程教材，还可以作为办公自动化和数字媒体技术相关从业人员的自学用书。

图书在版编目 (CIP) 数据

Python 程序设计基础与应用 / 孙福权 , 余浩谦 , 孙梦迪主编 . -- 北京 : 清华大学出版社 , 2026. 3.
(高等院校计算机应用系列教材). -- ISBN 978-7-302-70998-5

Ⅰ. TP312.8

中国国家版本馆 CIP 数据核字第 2026HK8686 号

责任编辑：刘金喜
封面设计：高娟妮
版式设计：思创景点
责任校对：成凤进
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：三河市铭诚印务有限公司
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张：**13.5　　**字　　数：**304 千字
版　　次：2026 年 2 月第 1 版　　**印　　次：**2026 年 2 月第 1 次印刷
定　　价：58.00 元

产品编号：114727-01

前言

在人工智能、大数据与自动化技术深度渗透各行各业的今天，编程已不再是计算机专业人员的专属技能，而是成为跨越文理学科人员的通用素养。Python作为一门兼具简洁性与强大功能的编程语言，以其灵活高效的特性，渗透到各个行业的核心场景，成为连接技术与需求的重要桥梁。教育部将Python纳入全国计算机等级考试科目，各大高校纷纷开设相关课程，企业对Python技能人才的需求持续攀升，足以见证其在当下技术体系中的核心地位。无论是计算机相关专业的学生、寻求技能提升的职场人士，还是对编程充满好奇的入门者，选择Python作为编程之旅的起点，都能快速收获成就感与实用技能。在新文科、新理科交叉融合的教育背景下，Python以其低门槛、高扩展性的特质，成为培养大学生计算思维、赋能专业学习的理想载体。

本书在编写过程中，始终贯彻立德树人、培养德智体美劳全面发展的社会主义建设者和接班人这一精神，注重知识传授与价值引领的结合。本书立足“基础扎实、实战导向、应用为王”的编写理念，旨在为读者打造一套系统全面、循序渐进的Python学习体系。我们深知，编程学习的核心不仅是语法规则的记忆，更是逻辑思维的培养和实践能力的提升。因此，本书在内容编排上遵循认知规律，从基础概念到高级应用层层递进，既保证知识的完整性，又注重学习的连贯性。

本书专为各专业学生的通识教育课程量身打造，我们深知核心读者中既有具备编程基础的学生，更有大量零编程经验的非计算机专业学习者。因此，本书减少传统教材中晦涩的底层原理与复杂语法细节的讲解比重，将培养计算思维置于核心位置，引导学生学会像计算机科学家一样分解问题、抽象建模、高效求解，最终能够运用Python解决本专业的实际问题。

本书的编写贯穿三大核心教学理念：一是思维先行，不局限于代码语法的讲解，更注重通过编程案例培养学生的逻辑推理、问题拆解与抽象概括能力，让计算思维成为可迁移的核心素养；二是学以致用，坚持从实践中来，到实践中去，每个基础知识点都配套源自现实生活或跨学科场景的案例——从简单的成绩统计到复杂的数据分析，从网络爬虫获取专业数据到可视化呈现研究结果，让学生直观感受到Python的实用价值；三是降低焦虑感，采用友好通俗的语言风格，将编程错误视为学习过程的自然部分，通过详细的异常解析与分步指导，鼓励学生大胆试错、主动探索。

在内容设计上，本书具有鲜明的通识教育特色：结构由浅入深、层层递进，从Python语言简介与环境搭建入手，逐步覆盖语法基础、程序控制结构、组合数据类型、函数定义与使用等核心知识，再延伸至模块与库、文件处理、网络爬虫、数据分析与可视化等应用技能；案例跨学科属性突出，涵盖金融、社科、生物、管理等多个领域，让不同专业的学

生都能找到贴合自身需求的实践场景；注重实用性与可操作性，每个章节都配有清晰的示例代码、详细的步骤说明与针对性习题，同时融入AI工具辅助编程的实用技巧，帮助学生高效上手。

本书采用模块化设计，全书共9章，构建了“基础铺垫—核心技能—应用拓展”的完整知识体系：

第1章从Python的诞生背景、版本演进和核心特点入手，详细讲解环境搭建流程，涵盖IDLE、PyCharm等常用开发环境的使用，搭配简单实例帮助读者快速上手。

第2章系统梳理语法基础，包括数据类型、运算符、内置函数等核心知识点，通过对比示例和实操代码，让读者理解语法本质。

第3章聚焦程序控制结构，深入解析顺序、选择、循环结构的逻辑原理，结合异常处理机制，培养读者编写健壮代码的能力。

第4章详解列表、元组、字典、集合等组合数据类型，剖析其特性与适用场景，提升读者数据处理的灵活性。

第5章围绕函数展开，从定义调用、参数传递到变量作用域、高阶函数，全面覆盖函数编程的核心知识点，通过模块化编程示例，让读者体会代码复用的高效。

第6章介绍模块与库的使用，既讲解math、datetime等标准库的实用功能，也详细说明第三方库的安装与应用，拓展读者的工具储备。

第7章专注文件处理，涵盖文件读写、目录操作、CSV处理等实用技能，通过日志分析、数据备份等案例，强化读者的实际操作能力。

第8章系统讲解网络爬虫技术，从requests、BeautifulSoup等库的使用到动态网页抓取，再到数据存储方案，完整呈现爬虫开发的全流程。

第9章深入数据分析与可视化，通过NumPy、Matplotlib等库的实战应用，让读者掌握数据处理、图表绘制的核心技能，实现从数据到洞察的转化。

对于学生而言，学习本书时请保持开放与探索的心态：不必畏惧初期的语法困惑，多动手编写代码、调试错误，将每一次实践都视为思维训练的过程；善用书中的案例与习题，结合自身专业场景进行拓展练习，让Python成为表达创意、解决专业问题的工具。对于教师而言，本书章节设计灵活，可根据授课对象的专业背景调整教学重点——面向文科学生可侧重数据处理与可视化，面向理工科学生可强化算法与应用开发，通过案例的个性化选取实现“通识教育+专业赋能”的双重目标。

本书具有三大鲜明特色：一是注重实操性，每个核心知识点都搭配短小精悍的示例代码，关键章节设置综合案例，从需求分析到代码实现逐步拆解，让读者在实践中巩固知识；二是强化实战导向，实战例题源于 Web 开发、数据分析、自动化运维等实际应用场景，覆盖职场与科研常见需求，帮助读者实现知识向技能的转化；三是强调易读性，语言通俗易懂，避免复杂术语堆砌，通过图表辅助、代码注释等方式，降低学习门槛，让不同基础的读者都能轻松跟进。

本书的编写凝聚了多位兼具计算机教学经验与通识教育理念的教师的智慧与心血，具体分工如下：第1章由东北大学孙梦迪博士与李琳合作编写，第2章、第3章由余浩谦编写，

第4章由李楠编写，第5章由孙宇编写，第6章由孙梦迪博士与郝政鑫合作编写，第7章由王莹编写，第8章、第9章由刘明编写。全书由孙福权、余浩谦、孙梦迪担任主编，负责整体框架设计、内容统稿和质量把控工作。在编写过程中，编写团队多次召开研讨会，就知识体系构建、示例选取、表述方式等进行了深入讨论，确保本书内容的科学性、系统性和实用性。

本书编写过程中，我们参考了大量行业实践案例和最新技术文档，力求内容准确、与时俱进。本书可作为高等院校计算机及相关专业的教材，以及非计算机专业的通识课程教材，也可作为编程爱好者、职场提升者的自学用书。我们衷心希望这本书能成为读者学习Python之路上的良师益友，帮助大家不仅掌握编程技能，更能在学习过程中逐步培养创新能力及解决实际问题的应用能力；能够帮助广大学生跨越编程门槛，掌握计算思维，让Python成为学业深造与职业发展的有力助手，在跨学科融合的时代浪潮中收获更多可能。

为便于教学，本书提供了PPT教学课件、教学大纲、案例源码、教学日历和习题答案等教学资源，读者可通过扫描下方二维码下载。

教学资源

由于计算机程序语言发展迅速，加之编者学识有限，本书难免存在疏漏和不足之处，恳请广大读者批评指正，以便我们后续不断完善。

服务邮箱：476371891@qq.com

编　者

2025年12月

目录

第 1 章

Python 入门

本章导言

本章开启Python入门之旅，先从语言特色与多元应用领域，帮您建立基本认知；接着手把手教您搭建环境，涵盖Python下载安装、自带IDLE的两种执行模式，以及常用开发环境(PyCharm、Anaconda)的使用；最后通过程序示例实操，搭配异常解析与DeepSeek辅助分析，让您快速上手写代码，迈出Python学习第一步。

本章学习目标

(1) 掌握Python的诞生背景、核心特点、版本演进及多领域应用场景，明确解释型与编译型语言的差异，熟记IDLE、PyCharm等开发环境的关键操作要点。

(2) 能够独立完成Python环境搭建与验证，熟练运用IDLE的交互式和文件式两种开发模式编写运行简单程序，初步学会借助 AI 工具辅助代码分析与优化。

(3) 建立Python编程的基本思维逻辑，培养代码可读性与规范性意识，激发对Python在Web 开发、数据分析等领域的应用兴趣。

1.1 Python语言简介

1.1.1 Python语言诞生背景与设计理念

20世纪80年代末，计算机科学领域正处于快速发展期，当时的编程语言要么过于复杂(如C、C++)，要么功能局限(如Shell脚本)，缺乏兼顾可读性与高效性的通用语言。

荷兰数学和计算机科学研究所(CWI)的研究员Guido van Rossum，希望创造一种“优雅”“明确”的编程语言，以满足快速开发原型和系统脚本编写的需求，Python由此应运而生。

Python的设计深受ABC语言的影响，ABC语言是为教学和小型软件开发设计的语

言，强调代码可读性与模块化。Guido汲取了ABC语言的“用统一的语法处理不同数据类型”“强类型检查”等理念，同时摒弃其过于学术化的设计，例如，Python引入了ABC语言的缩进规则，强制代码块以缩进界定，彻底解决了代码块归属混乱的问题，让代码结构更加清晰直观。

1.1.2 Python语言的版本演进

1991年，Python的第一个解释器诞生。它由C语言实现，而且受ABC语言的影响，因此其中很多语法来源于C语言和ABC语言。而Python 1.0版本真正发布于1994年1月，这是Python首个正式版本，引入了lambda、map、filter和reduce等函数式编程特性，以及对交互式编程的支持。该版本奠定了Python“脚本语言”的基础定位，通过内置的丰富标准库，如os模块支持系统调用、socket模块实现网络编程，极大地降低了开发门槛。

Python 2.0版本是在2000年10月发布的，从此版本开始，Python开始成为互联网时代最流行的编程语言之一，其被广泛应用于Web开发(如Django、Flask框架)、科学计算(NumPy、SciPy库)等领域。这个版本的主要新功能是内存管理和循环检测垃圾收集器以及对Unicode的支持，这些功能构成了现在Python语言框架的基础。之后在2004年，Python升级到2.4版本，同年最流行的Web框架Django诞生。之后Python陆续推出Python 2.5/2.6/2.7版本。截至目前，仍然有很多企业在使用Python 2.7版本。

Python 2.x与Python 3.x存在显著差异：Python 2.x默认使用 ASCII 编码，导致中文等非英文字符处理困难；语法上保留了大量历史遗留特性(如print语句，而非函数)。2014年，Python 社区宣布2020年停止对Python 2.x 的支持，这一过渡旨在解决Unicode兼容性、优化内存管理等底层问题，虽然给存量项目带来迁移成本，但为Python的长期发展扫清了障碍。

目前，Python开发主流应用的是Python 3.x版本，以Python 3.0为起点，彻底革新了编码系统，默认支持Unicode，从根源上解决了字符编码问题。后续版本持续迭代，如Python 3.5引入asyncio库，推动异步编程的普及；Python 3.8 优化了f-string 语法，让字符串格式化更简洁高效；Python 3.10 新增结构模式匹配(类似其他语言的switch语句)，进一步提升了代码表达能力。这些更新使 Python 在数据科学、人工智能、自动化运维等新兴领域保持领先地位。建议大家如果要学习Python编程语言，可以直接从Python最新版本开始，本书将采用Python的3.11.5版本进行实例介绍。Python版本演进过程如表1-1所示。

表 1-1 Python 版本演进过程

发布版本	源自	年份	所有者	GPL兼容
0.9.0至1.2	n/a	1991—1995	CWI	是
1.3至1.5.2	1.2	1995—1999	CNRI	是
1.6	1.5.2	2000	CNRI	否
2.0	1.6	2000	BeOpen.com	否

续表

发布版本	源自	年份	所有者	GPL兼容
1.6.1	1.6	2001	CNRI	否
2.1	2.0+1.6.1	2001	PSF	否
2.0.1	2.0+1.6.1	2001	PSF	是
2.1.1	2.1+2.0.1	2001	PSF	是
2.1.2	2.1.1	2002	PSF	是
2.1.3	2.1.2	2002	PSF	是
2.2至3.0	2.1.1	2001至今	PSF	是
3.0及更高	2.6	2008至今	PSF	是

1.1.3 Python语言特点

1. 简单易学

Python以其简洁的语法闻名。其他编程语言，比如C++ 与Java，它们在定义变量、函数等结构时，往往需要遵循较为复杂的语法规则，书写大量的符号和语句。而Python采用了更加简洁明了的语法，代码行数常常更少。

如下例，在Python中定义一个简单的函数，只需寥寥几行代码，就能清晰地表达出函数的功能。同时，Python的代码风格非常接近自然语言，就像日常用英语描述逻辑一样，很容易让初学者理解其逻辑含义，极大地降低了编程学习的门槛。

【示例1-1】计算1到10的整数之和的Python和C程序对比。

```
#python程序
sum = 0
for i in range(1, 11):
    sum += i
print(sum)  # 输出: 55

//C语言程序
#include <stdio.h>
int main() {
    int sum = 0;
    for (int i = 1; i <= 10; i++) {
        sum += i;
    }
    printf("%d\n", sum);  // 输出: 55
    return 0;
}
```

2. 免费开源

Python是免费开源的编程语言。开源意味着其源代码对所有人公开，开发者能够自由获取、修改和分发。这一特性催生了庞大且活跃的开源社区，众多开发者积极贡献代码，不断丰富Python的功能。大量功能强大的开源库和框架由此涌现，涵盖了从数据处理、网络编程到图形界面开发等各个领域。同时，开源社区还为开发者提供了交流和学

习的平台，大家可以在社区中分享经验、讨论技术问题，共同推动Python的持续发展与创新。

3. 高级语言

Python属于高级编程语言。高级语言的显著特点是，它在很大程度上屏蔽了底层硬件的细节，如内存管理。在使用Python编程时，开发者无须手动去分配和释放内存，Python内部的解释器或编译器会自动处理这些底层操作，让开发者能够将更多的精力集中在业务逻辑的实现上。

4. 可移植性

Python具备出色的可移植性，能够在多种主流操作系统上运行，包括Windows、Linux、macOS等。这得益于Python的解释器设计，它可以将Python代码转换为对应操作系统能够理解和执行的指令。无论在Windows系统下编写的Python程序，还是在Linux系统中开发的代码，只要遵循Python的标准规范，在不同操作系统之间迁移时，往往无须修改或仅需少量调整就能正常运行。这一特性使得Python开发者能够轻松地在不同平台上部署和运行自己的程序，极大地拓展了Python的应用范围。

5. 解释型语言

Python是解释型语言。与编译型语言(如C、C++)不同，解释型语言在运行时，是由解释器逐行读取并执行代码的。在编译型语言中，代码需要先经过编译器一次性将整个源程序编译成机器语言，然后才能运行。而Python的解释执行方式，使得开发者在编写代码后可以快速测试运行，及时发现语法错误或逻辑问题。不过，解释型语言也存在一定的性能劣势，在运行速度上通常不如编译型语言，但Python通过不断的优化以及针对特定场景使用扩展模块等方式，在一定程度上弥补了这一不足。

6. 面向对象

Python全面支持面向对象编程(OOP)。面向对象编程的核心概念包括类、对象、封装、继承和多态。在Python中，开发者可以方便地定义类，类中包含属性和方法，通过实例化类来创建对象。例如，定义一个“汽车”类，其中可以包含“颜色”“速度”等属性，以及“加速”“刹车”等方法。封装是指将数据和操作数据的方法绑定在一起，隐藏内部实现细节，只对外提供必要的接口。继承允许一个类继承另一个类的属性和方法，实现代码的复用。多态则是指相同的方法在不同对象上可以有不同的表现形式。Python的面向对象特性使得代码具有更好的组织性、可维护性和可扩展性，特别适合大型项目的开发。

7. 丰富的库和模块

Python拥有极为丰富的库和模块。其标准库涵盖了文件操作、网络编程、数据库访问、图形界面等众多领域。例如，通过标准库中的os模块可以方便地进行文件和目录操作；socket模块用于实现网络通信。

除了标准库，Python还有大量优秀的第三方库。在数据科学领域，NumPy用于高效的数值计算，Pandas提供了强大的数据处理和分析功能，Matplotlib则用于数据可视化。在Web开发方面，Django和Flask是常用的Web框架，能够帮助开发者快速搭建功能完善的网站。

这些丰富的库和模块，极大地拓展了Python的应用能力，让开发者无须从头开始编写大量基础代码，通过调用库中的函数和类，就能轻松实现复杂的功能。

8. 可扩展性

Python具有良好的可扩展性。在一些对性能要求较高的场景下，Python允许与其他语言(如C、C++)结合使用。这是因为Python提供了与其他语言交互的机制，开发者可以使用C或C++编写核心的、对性能敏感的代码，然后在Python中调用这些代码。

例如，在开发一个涉及大量数值计算的科学应用程序时，对于计算密集型的部分，可以用C或C++编写，以提高执行效率，而程序的整体逻辑控制和其他部分则使用Python实现，充分发挥Python开发效率高、代码简洁的优势。通过这种方式，既能满足特定场景下的性能需求，又能利用Python的优势进行快速开发。

接下来将深入介绍Python在各应用领域的具体情况，通过典型案例和技术要点，让读者更清晰地认识Python强大的应用能力。

1.1.4　Python语言的应用领域

1. Web开发

Python在Web开发领域凭借Django、Flask等框架大放异彩。

Django是一个功能完备的框架，遵循模型-视图-控制器(MVC，在Django中通常称为模型-视图-模板，即MVT)架构，自带了对象关系映射(ORM)、用户认证、管理后台等诸多功能。以搭建一个博客网站为例，开发者利用Django的ORM可以轻松定义文章、用户等数据模型，并与数据库交互；通过内置的用户认证系统，能快速实现用户注册、登录等功能。

Flask则是一个轻量级框架，灵活性极高，适合快速开发小型项目或搭建API。比如开发一个简单的天气查询API，只需寥寥几行代码就能定义路由和视图函数，接收用户请求并返回天气数据，在数据库连接方面，Flask可以自由选择SQLite、MySQL等数据库，并通过第三方扩展库实现数据的存取。

2. 数据分析与科学计算

在数据分析领域，Python已然成为主流工具。

NumPy提供了高效的数值数组对象和大量数学函数，能够快速处理大规模数值数据。例如在进行矩阵运算时，NumPy的运算速度极快，并且支持广播机制，简化了代码编写。

Pandas库擅长处理结构化数据，其核心数据结构Series和DataFrame，可以方便地

进行数据的读取、清洗、转换和分析。当处理一份包含销售数据的CSV文件时，使用Pandas能轻松读取数据，通过数据筛选、缺失值处理等操作，快速完成数据清洗工作。

SciPy库则在科学计算方面表现出色，涵盖了优化、积分、插值等众多功能，常用于物理、工程等领域的科学计算任务。

3. 人工智能与机器学习

Python是人工智能和机器学习领域的首选语言。TensorFlow是Google开发的深度学习框架，具有强大的分布式计算能力和可视化工具，广泛应用于图像识别、语音识别等任务。

例如在图像分类任务中，利用TensorFlow可以搭建卷积神经网络(CNN)模型，对大量图片进行训练，实现高精度的图像分类。

PyTorch以其动态计算图和简洁的代码风格受到众多研究者青睐，在自然语言处理领域，使用PyTorch构建循环神经网络(RNN)或Transformer模型，可以完成文本翻译、情感分析等任务。

Scikit-learn则是经典的机器学习库，它提供了丰富的机器学习算法，如分类、回归、聚类算法，适合初学者快速上手实现各种机器学习任务。

4. 自动化脚本与任务

Python在自动化领域的应用十分广泛。

在文件批量处理方面，通过Python的文件操作相关模块，如os和shutil，可以轻松实现文件的复制、移动、重命名等操作。比如，将一个文件夹下所有特定格式的文件，批量复制到另一个文件夹中，只需编写简单的Python脚本即可完成。

在系统管理任务自动化上，Python可以调用系统命令，结合subprocess模块实现系统监控、服务管理等操作。例如，编写脚本定期检查服务器的磁盘空间使用情况，当空间不足时自动发送邮件提醒管理员。

5. 游戏开发

虽然Python并非游戏开发的主流语言，但在2D游戏开发领域独具优势。Pygame库提供了一系列用于开发游戏的功能模块，包括图形绘制、声音处理、输入控制等。以开发一个简单的飞机大战游戏为例，利用Pygame可以创建游戏窗口，绘制飞机、子弹、敌机等游戏元素，通过监听键盘或鼠标事件实现飞机的移动和射击操作，利用碰撞检测功能判断子弹是否击中敌机，为玩家带来丰富的游戏体验。

6. 桌面应用开发

Python可以借助Tkinter、PyQt等库开发桌面应用程序。

Tkinter是Python的标准GUI库，简单易用，适合快速开发小型桌面应用。使用Tkinter创建一个简单的计算器应用，只需创建按钮、文本框等组件，并绑定相应的事件处理函数，就能实现基本的计算功能。

PyQt则是一个功能强大的GUI框架，基于Qt库开发，提供了丰富的界面组件和高级

功能，适用于开发复杂、美观的桌面应用，在企业级桌面应用开发中也有广泛应用。

7. 网络爬虫

网络爬虫用于从网页中提取数据，Python凭借其丰富的库在该领域极具优势。

BeautifulSoup是一个用于解析HTML和XML文档的库，使用它可以轻松定位网页中的元素，并提取所需的数据。当爬取一个新闻网站的文章标题和内容时，通过BeautifulSoup解析网页，利用CSS选择器或XPath表达式找到对应的HTML标签，即可获取数据。

Scrapy则是一个专业的网络爬虫框架，提供了高效的数据抓取、处理和存储功能，支持分布式爬虫，适用于大规模、复杂的爬虫项目，能够满足企业级的数据采集需求。

8. 物联网(IoT)

Python在物联网领域发挥着重要作用，尤其在树莓派等微型计算机设备上。树莓派搭载Linux系统，支持Python编程，开发者可以使用Python连接各类传感器，如温度传感器、湿度传感器、光线传感器等，实现数据采集。例如，通过Python代码读取温度传感器的数据，并将数据上传至云端服务器。

在设备控制方面，Python可以控制舵机、电机等执行器，实现智能家居设备的自动化控制，如根据光线强度自动调节灯光亮度。

9. 教育领域

Python在编程教育中占据重要地位，因其简单易学的特性而成为初学者入门编程语言的首选。在学校教学中，Python被纳入中小学信息技术课程，帮助学生初步建立编程思维。在线编程课程也大量使用Python作为教学语言，通过生动有趣的项目案例，如开发小游戏、制作简单动画等，激发学生的学习兴趣。学生在学习Python的过程中，不仅掌握了编程语法，还能培养解决问题的能力，为进一步学习其他编程语言和计算机科学知识奠定基础。

1.2　Python环境搭建

1.2.1　Python下载和安装

Python是跨平台的开发语言，支持Windows、Linux、Mac OS等多种平台，本书演示在Windows环境下安装和配置python 3.11.5。

1. 下载

(1) 打开Python官网https://www.python.org，将鼠标指针移至网页中的Downloads栏目，选择Windows操作系统，如图1-1所示。

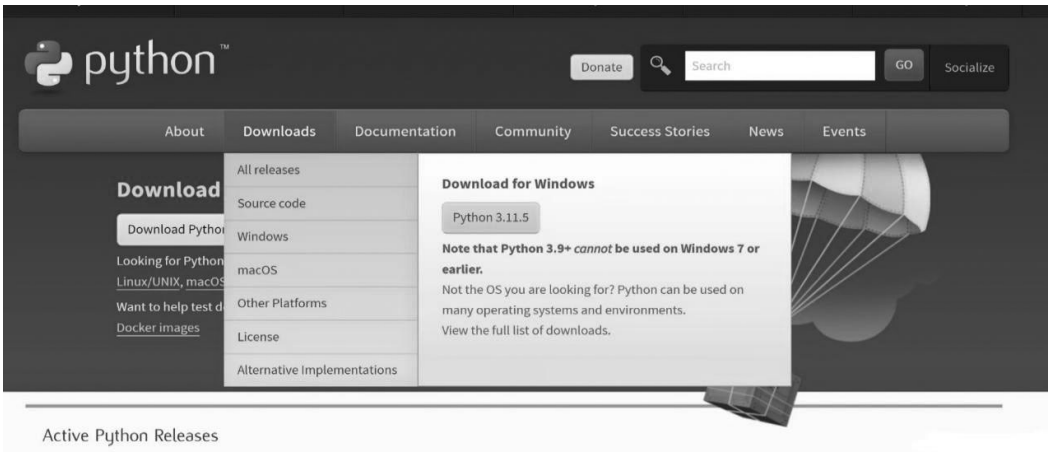

图 1-1　python 官网界面

(2) 进入Windows对应的页面，选择Python的稳定发布版本Stable Releases，如图1-2所示。

图 1-2　Python3.11.5 版本详情

图片左侧是稳定发布版本Stable Releases，右侧是预发布版本Pre-releases，前者是经过测试，相对完善、稳定的版本；后者还处于测试中，可能不完善。因此，下载左侧的稳定发布版本Stable Releases。

32-bit是指32位操作系统，通过此类链接下载的包适合32位操作系统，基于32位处理器。64-bit是指64位操作系统，基于64位处理器，下载的安装包适合64位操作系统。

(3) 用鼠标右键单击“我的电脑”，选择菜单中的“属性”，可以查看自己电脑的硬件配置和操作系统的类型，如图1-3所示。

(4) 根据自己电脑的操作系统的位数，选择图1-4中红框对应的Python的安装程序Windows Installer。

图 1-3　“我的电脑”配置详情

图 1-4　Python 安装包版本选择

Windows embeddable package是一个Python简化版本，主要用于嵌入其他应用程序。用Python进行程序开发，需要下载Windows Installer安装程序，这个安装程序具有一个较为完整的Python开发环境。

2. 运行安装Python的安装程序Windows Installer

(1) 运行下载的安装程序，在出现的图1-5所示的安装界面中选中Use admin privileges when installing py.exe和Add python.exe to PATH，单击Customize installation进入自定义安装Python可选功能设置界面。

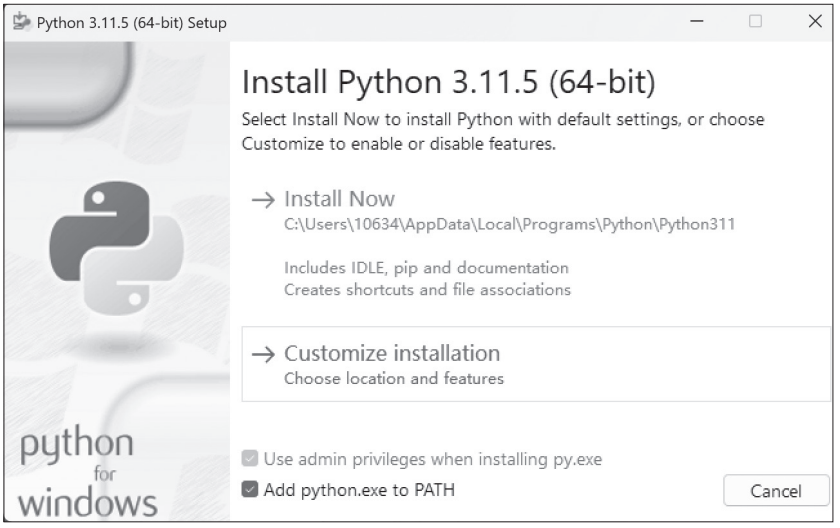

图 1-5　Python 安装界面

InstallNow默认安装会自动额外安装IDLE、pip、documentation、Creates shortcuts and file associations(在“开始”菜单中创建快捷方式，创建文件关联)。默认安装是安装相对基础的内容，但已具备Python基本开发功能，如果要减少或增加功能，或指定安装路径，可以选择自定义安装。通常选择自定义安装，可以增加更多的功能。

(2) 按图1-6所示设置自定义安装Python可选功能。

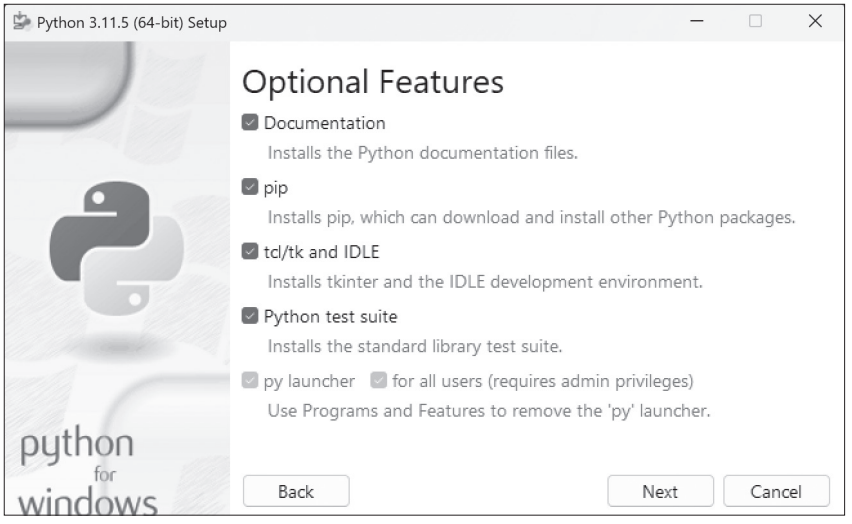

图 1-6　设置自定义安装功能

(3) 单击Next按钮，进入如图1-7所示安装界面，选中全部选项。用户可以根据自己的需要进行选择，这里选择前面五项。指定安装路径，建议安装在D盘新建的文件夹中。单击Install按钮进行安装，进入Python安装进度界面。

(4) 安装进度完成后，出现Python安装成功的提示界面，如图1-8所示。至此，Python安装完成。

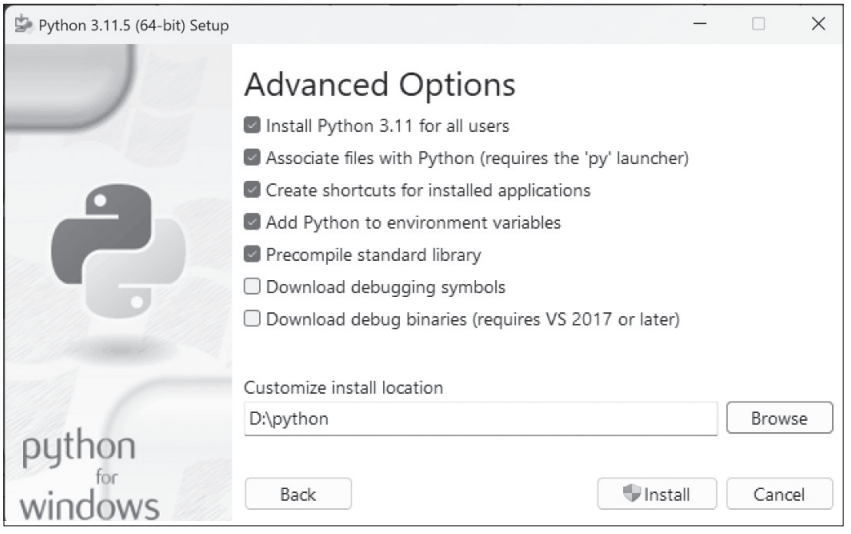

图 1-7　选择要安装的项目

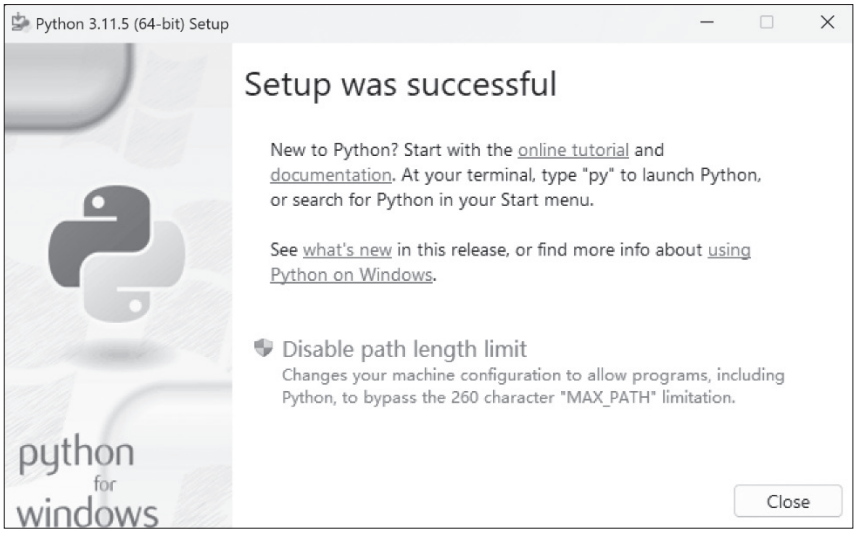

图 1-8　安装完成界面

3. 检验Python环境

(1) 按Windows+R组合键，弹出“运行”对话框，输入命令cmd，如图1-9所示。

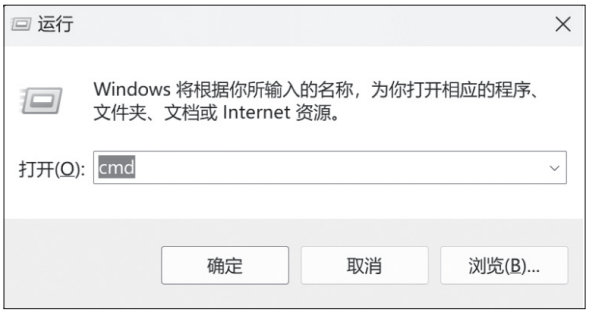

图 1-9　打开“运行”对话框

(2) 在打开的命令提示行中输入Python并按Enter键，会出现Python的版本信息，此处版本是python 3.11.5，如图1-10所示。此时输入行前面出现>>>符号，表示已经进入Python环境。输入quit()可以退出Python，接着输入exit可退出命令行界面。

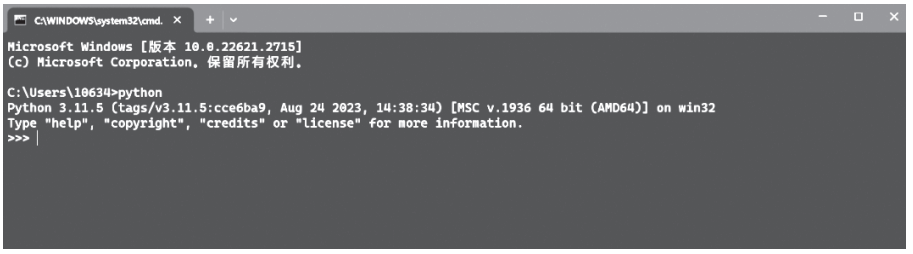

图 1-10　测试 python 安装是否成功

1.2.2　Python自带开发环境IDLE

IDLE 是 Python 自带的简易开发环境，Python安装完成后，在 Windows 的“开始”菜单中可以找到Python文件夹。

接下来介绍开发环境的搭建。Python 提供两种开发环境：交互式开发环境和集成开发环境，前者用于测试，后者用于生产。本节介绍在两种开发环境下如何编写程序。

1. 交互式开发环境

(1) 单击Windows系统的“开始”菜单，然后依次选择“所有程序”→Python→IDLE命令，即可打开IDLE窗口，如图1-11所示。

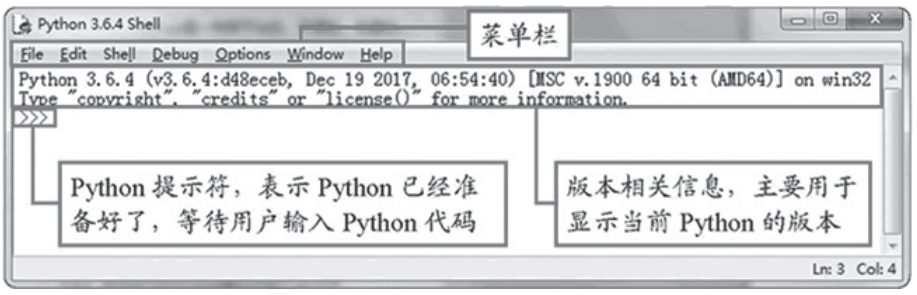

图 1-11　IDLE 窗口功能介绍

(2) 在IDLE主窗口的菜单栏上，选择File→New File命令，将打开一个新窗口，在该窗口中，可以直接编写Python代码。

(3) 打开IDLE后，会直接进入交互式命令行窗口，在这里输入Python代码，按下Enter键就能立刻看到执行结果，就像和Python进行对话一样。比如，输入print("Hello, Python!")，按下Enter键，会显示"Hello, Python!"；输入2 + 3，会得到5。

这种方式特别适合快速测试一段代码，验证某个函数的用法，或者进行简单的数学计算。例如，想知道len()函数如何计算字符串长度，直接输入len("example")，就能得到结果7。它还能查看变量的值，先定义x = 10，再输入x，就能看到10显示出来，方便随时检查程序状态，如图1-12所示。

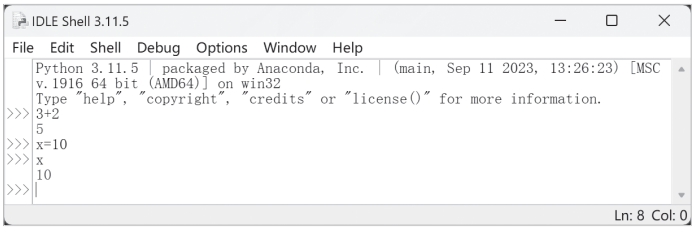

图 1-12　交互式开发界面

2. 集成开发环境

当需要编写较长、较复杂的程序时，交互执行方式就不太方便了，这时可以使用程序文件执行方式。

(1) 在IDLE中，选择菜单栏中的File→New File命令，会弹出一个空白的编辑窗口。在这个窗口里，可以像写文章一样，把完整的 Python 程序代码编写进去。比如，编写一个计算斐波那契数列的程序，如图1-13所示。

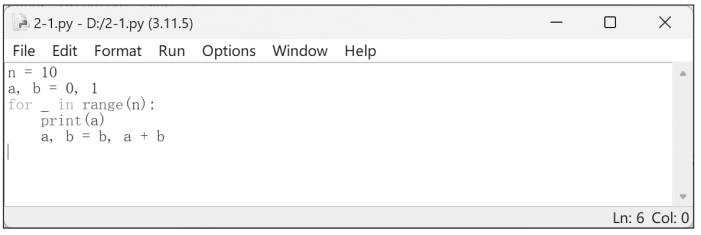

图 1-13　脚本式开发界面

【示例1-2】计算斐波那契数列。

```
n = 10
a, b = 0, 1
for _ in range(n):
    print(a)
    a, b = b, a + b
```

(2) 编写完成后，选择File(文件)→Save(保存)命令，给文件取个名字，注意要加上.py的扩展名，比如2-1.py。保存文件后，选择菜单栏中的 Run(运行)→Run Module(运行模块)命令，或者直接按下键盘上的F5键，IDLE 就会执行这个Python程序文件，在交互式命令行窗口中显示运行结果，如图1-14所示。

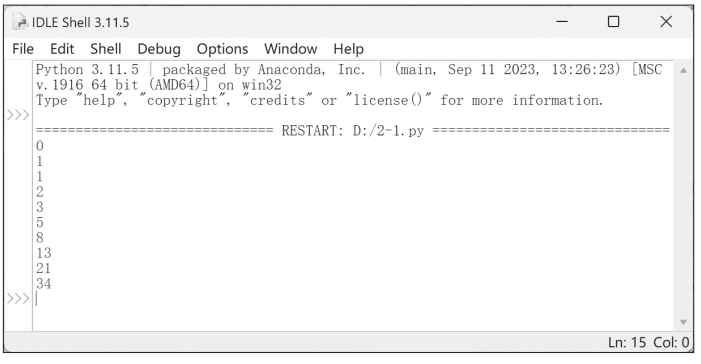

图 1-14　脚本式开发运行结果界面

如果程序运行过程中出现错误，IDLE 会在命令行窗口中提示错误信息，帮助找到代码中的问题并进行修改。通过程序文件执行方式，能够系统地组织代码，完成各种复杂的编程任务。

1.2.3 Python常用开发环境PyCharm

PyCharm 是一款专门为Python开发的跨平台集成开发环境(IDE)，由捷克布拉格的JetBrains公司开发。它适用于Windows、Linux和macOS操作系统。除了支持Python语言开发外，还可通过安装插件支持Java、C++、Go等其他语言，以及数据库工具和SQL，其专业版还支持HTML、CSS和JavaScript等前端技术。

PyCharm带有一整套能帮助用户在使用Python语言开发时提高效率的工具，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，还提供了一些高级功能，用于支持Django框架下的专业Web开发。

官网提供的PyCharm分为专业版(Professional)和社区版(Community)。

❖ 注意：

专业版是收费的，功能更加全面，社区版是简洁版本，但它是免费的。一般来说，使用社区版就够了，除非需要用Python进行Django等Web开发，才需要用到专业版。

(1) 程序下载。

① 打开PyCharm官网https://www.jetbrains.com/pycharm，单击右上角的语言设置图标，选择“简体中文”，将网页调整为中文模式，如图1-15所示。

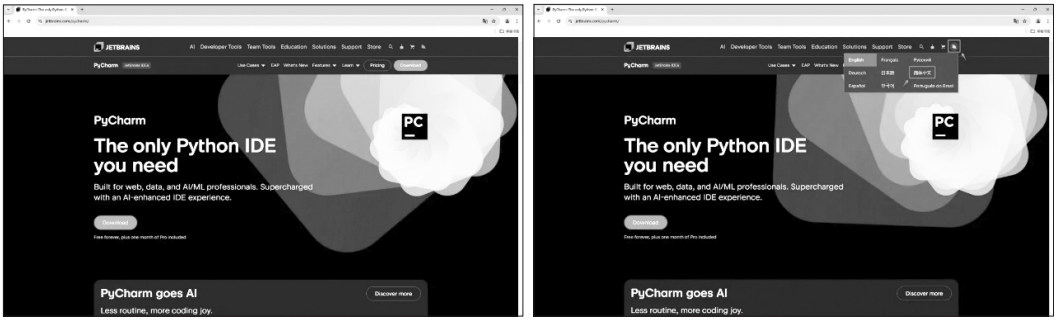

图 1-15　PyCharm 官网

② 单击“下载”按钮，进入下载页面下载Windows应用程序，如图1-16所示。

图 1-16　下载 PyCharm

(2) 安装PyCharm。

① 下载完成后，打开安装程序，根据提示输入安装地址，如图1-17所示。

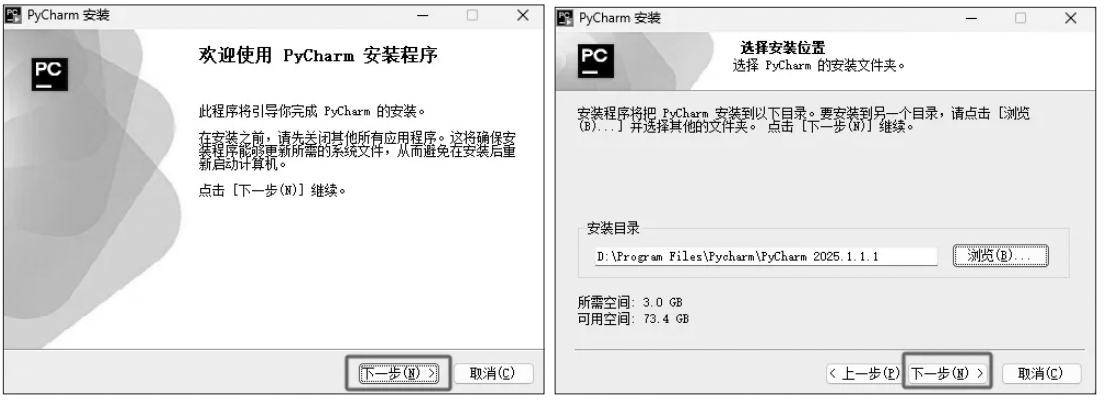

图 1-17　输入安装地址

② 单击“下一步”按钮，打开如图1-18所示的“安装选项”界面。选中PyCharm，安装后会在桌面生成PyCharm的快捷图标；选中“将‘bin’文件夹添加到PATH”，则把PyCharm的bin目录加入系统环境变量PATH，让系统能直接识别PyCharm的命令，比如通过终端快速启动；选中.py：可让系统默认用PyCharm打开.py后缀的Python文件。

图 1-18　选择 PyCharm 安装选项

③ 单击“安装”按钮，开始安装，并显示安装进度，直至完成安装，如图1-19所示。

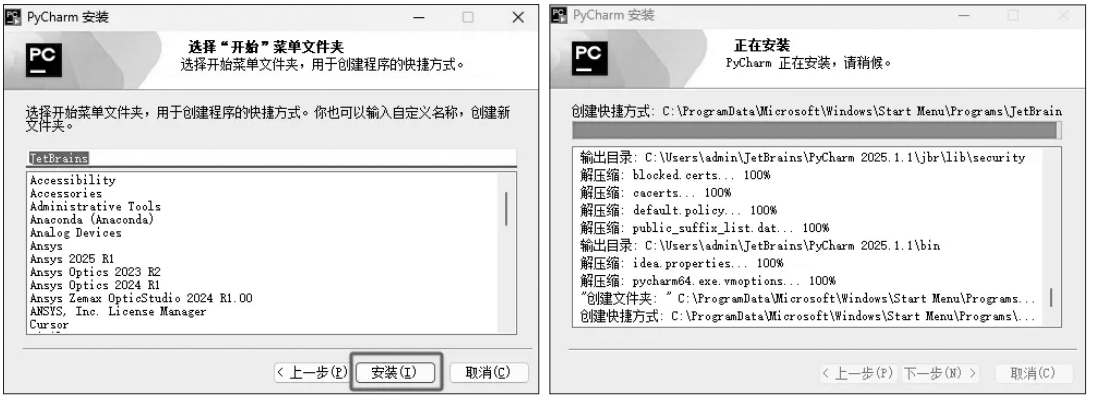

图 1-19　PyCharm 安装进度

(3) 配置PyCharm。

① 打开PyCharm，其运行界面如图1-20所示。

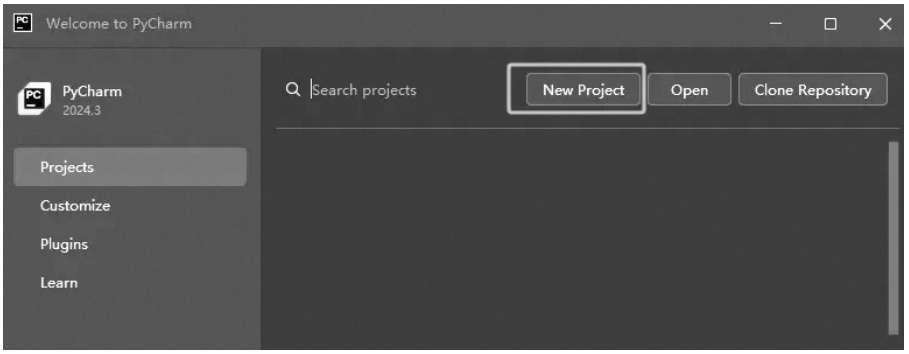

图 1-20　PyCharm 运行界面

② 单击New Project(新建项目)按钮，将打开如图1-21所示的New Project窗口。

- Location：用于指定代码保存路径(示例路径 E:\pythonProject\newproject)，设置后项目代码文件会存在这个文件夹里，方便后期找代码、管理文件。路径不要带中文 / 空格字符，避免程序运行报错。
- Interpreter type(解释器类型)：选择Custom environment(自定义环境)，指不用PyCharm自动生成的环境，而是用已有的Python环境，比如Anaconda配置好的环境。
- Environment：选中Select existing(选择已存在的环境)，不是重新生成新环境。
- Type：选择Python，明确使用Python语言进行开发。PyCharm也支持其他语言，这里选择Python。
- Python path：指定具体的Python解释器位置，此处为D:\Anaconda\envs\newenv\python.exe。

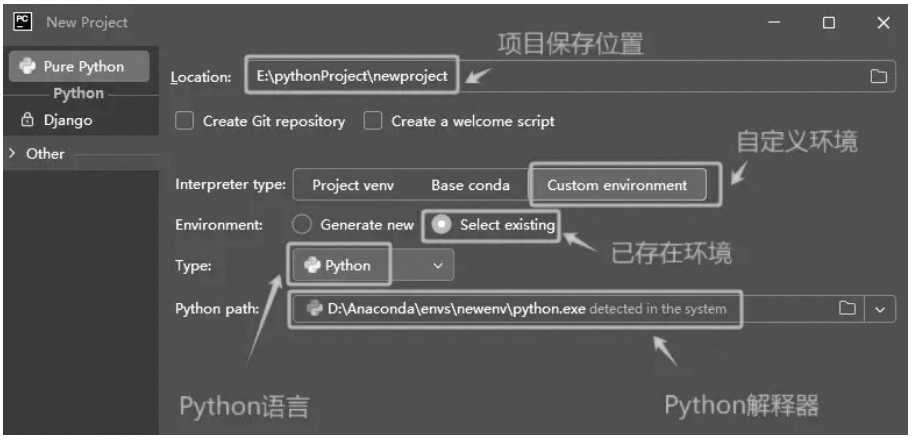

图 1-21　配置 PyCharm

③ 配置好PyCharm之后，依次选择New Project→New→Python File命令新建文件，如图1-22所示。

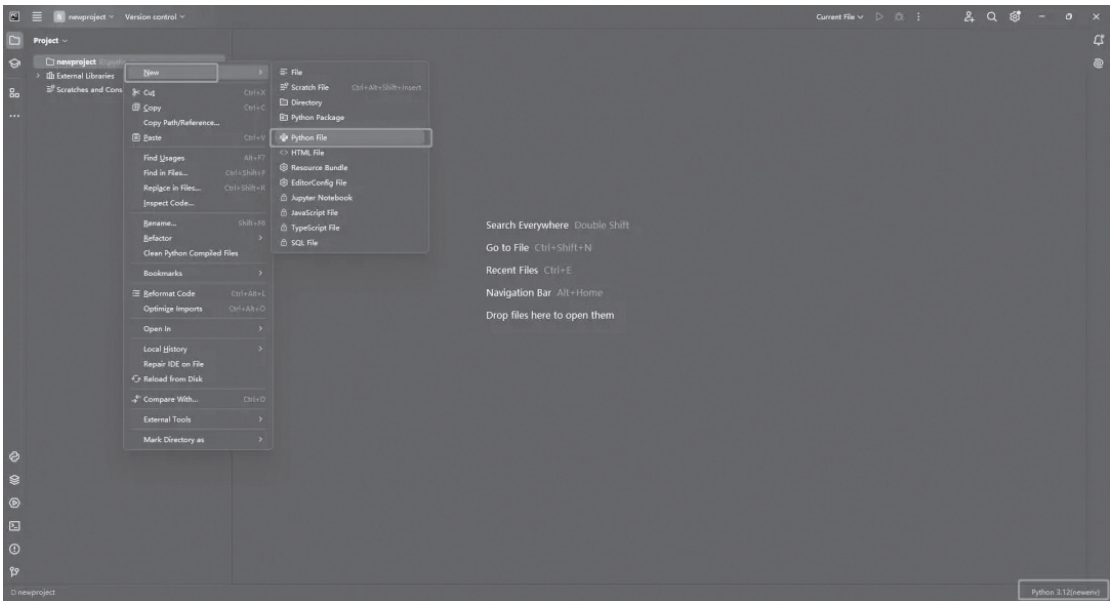

图 1-22　新建 Python 文件

1.3　Python程序设计示例

1.3.1　交互方式示例

交互方式在IDLE或命令行中逐行输入并执行。

【示例1-3】简单数学计算。

```
2+3*4
5**2
```

将上述两个算式分别输入IDLE开发环境中并执行，结果如图1-23所示。

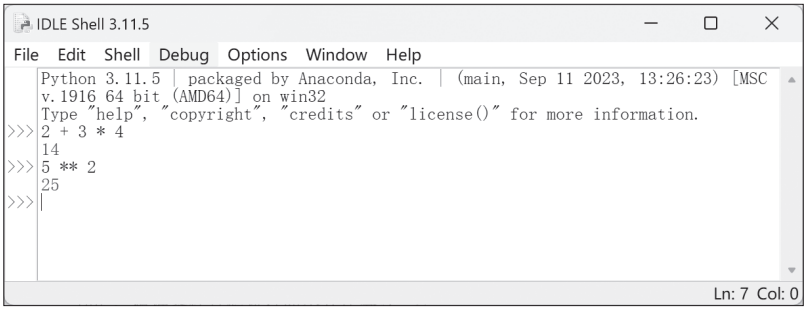

图 1-23　交互式运行示例 1

【示例1-4】变量赋值与使用。

```
name = "Alice"
age = 25
print(f"你好，我是{name}，今年{age}岁。")
```

将上述代码输入IDLE开发环境中并执行，结果如图1-24所示。

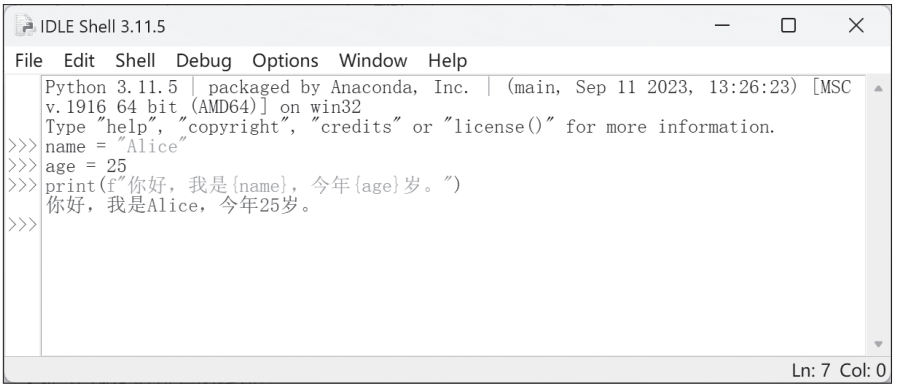

图 1-24　交互式运行示例 2

【示例1-5】条件判断。

```
x = 10
if x > 5:
    print("x大于5")
else:
    print("x小于等于5")
```

将上述代码输入IDLE开发环境中并执行，结果如图1-25所示。

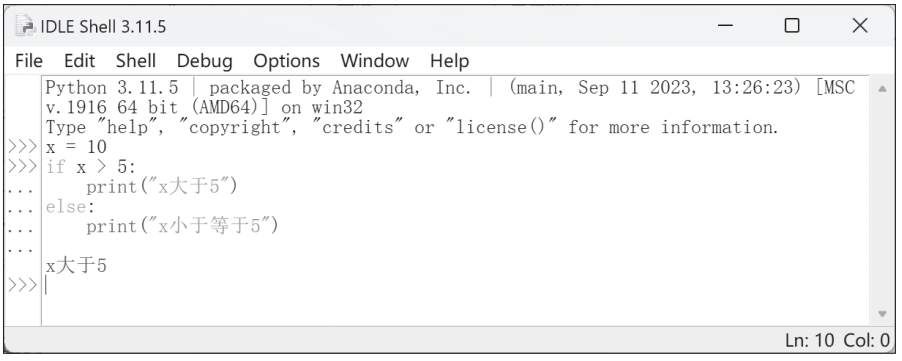

图 1-25　交互式运行示例 3

1.3.2　程序文件方式示例

程序文件方式是指将代码保存为.py文件后整体运行。

【示例1-6】计算斐波那契数列。

```
n = 10
a, b = 0, 1
print(f"斐波那契数列前{n}项：")
for _ in range(n):
    print(a, end=" ")
    a, b = b, a + b
print()
```

将上述代码保存为.py文件后，整体运行，结果如图1-26所示。

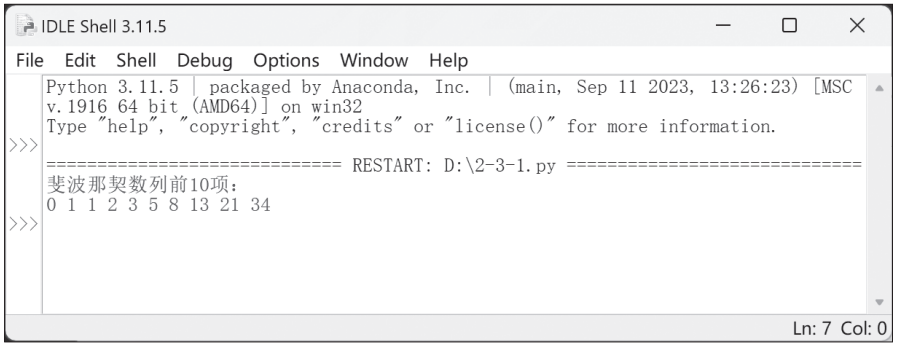

图 1-26　脚本式运行示例 1

【示例1-7】温度转换程序。

```
def celsius_to_fahrenheit(c):
    """将摄氏度转换为华氏度"""
    return (c * 9/5) + 32

def fahrenheit_to_celsius(f):
    """将华氏度转换为摄氏度"""
    return (f - 32) * 5/9

# 主程序
choice = input("请选择转换类型 (1: 摄氏度→华氏度, 2: 华氏度→摄氏度): ")
if choice == '1':
    c = float(input("请输入摄氏度: "))
    print(f"{c}° C 等于 {celsius_to_fahrenheit(c)}° F")
elif choice == '2':
    f = float(input("请输入华氏度: "))
    print(f"{f}° F 等于 {fahrenheit_to_celsius(f)}° C")
else:
    print("无效的选择")
```

将上述代码保存为.py文件后，整体运行，结果如图1-27所示。

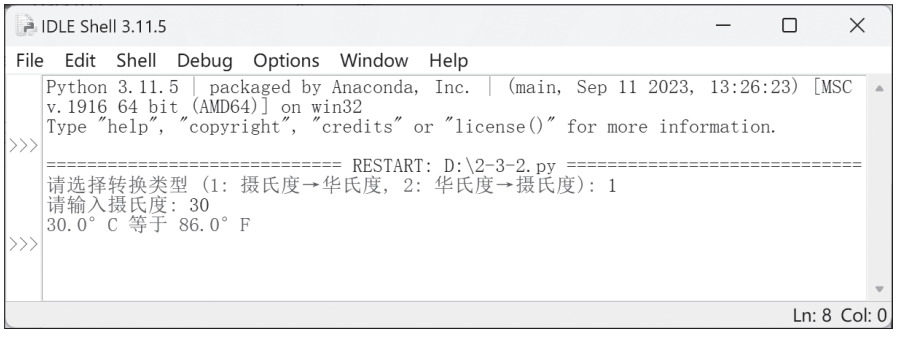

图 1-27　脚本式运行示例 2

【示例1-8】文件操作——写入和读取。

```
# 写入数据到文件
with open("data.txt", "w") as file:
    file.write("Hello, Python!\n")
    file.write("这是一个文件操作示例。\n")
```

```
# 从文件读取数据
with open("data.txt", "r") as file:
    content = file.read()
    print("文件内容：")
    print(content)
```

将上述代码保存为.py文件后，整体运行，结果如图1-28所示。

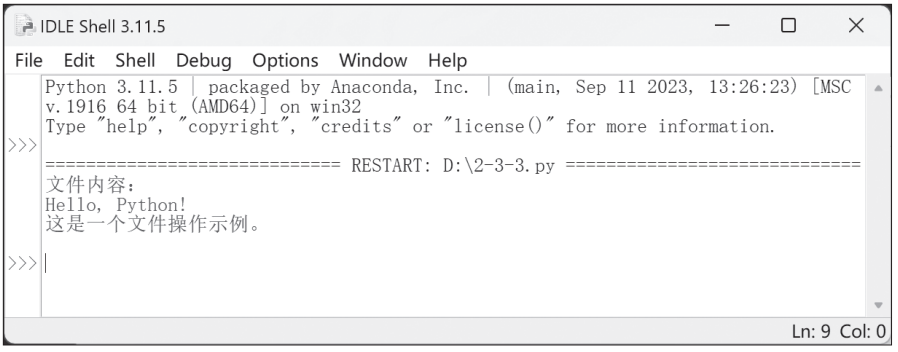

图 1-28　脚本式运行示例 3

1.3.3　使用AI工具辅助编程入门

1. 核心概念

在程序开发与调试过程中，利用AI大模型(如DeepSeek)分析程序是一种创新且高效的辅助手段。其核心在于借助模型对Python代码的理解能力，实现代码逻辑解析、潜在问题排查、优化建议提供等功能。简单来说，就是让大模型成为程序分析的“智能助手”，像资深程序员一样去审视代码，从语法、逻辑、功能实现等维度给出反馈。

2. 基础流程

(1) 代码输入与提交

将待分析的Python程序代码(可以是完整脚本、函数片段等)，按照DeepSeek交互界面或API要求的格式进行提交。这一步需注意代码的完整性与可读性，清晰的代码结构有助于模型更准确地理解。例如，提交一个简单的计算斐波那契数列的函数：

```
def fibonacci(n):
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return fibonacci(n - 1) + fibonacci(n - 2)
```

(2) 模型分析与反馈

DeepSeek接收代码后，会基于其训练的知识，对代码进行多方面分析。包括但不限于语法检查(是否存在语法错误，如拼写错误、缩进问题等)、逻辑合理性分析(如循环是否可能陷入死循环、条件判断是否全面)、算法效率评估(像上述斐波那契数列递归实

现，模型可指出其存在重复计算、效率低的问题)，然后以文本形式返回分析结果，给出问题说明、优化建议等内容。

(3) 结果解读与应用

开发者需要理解模型反馈的内容，区分不同类型的提示(如明确的语法错误提示、潜在逻辑风险预警、优化方向建议)。对于语法错误，可直接依据提示修改代码；对于逻辑和效率相关建议，需结合程序实际需求判断是否采纳优化，比如若斐波那契数列计算用于小规模数据场景，递归实现简单直观可保留；若用于大规模计算，则需按模型建议优化为迭代方式或带记忆化的递归。

3. 模型理解的局限性

DeepSeek虽具备强大的代码分析能力，但并非完美无缺。一方面，对于一些极度复杂、涉及特殊领域算法(如特定科学计算、加密算法等)且代码实现晦涩的程序，模型可能无法精准理解其全部逻辑，分析结果会存在偏差。例如，某段涉及量子计算模拟的Python代码，因模型训练数据中相关领域知识深度有限，分析时可能遗漏关键逻辑点。

另一方面，当代码存在多种合理实现方式且涉及业务逻辑权衡时(如为了兼容性牺牲部分效率，或为了简单性接受一定冗余)，模型给出的优化建议可能不符合实际业务场景需求，需要开发者结合业务判断，这对开发者的业务理解与技术判断力要求较高。

4. 代码提交的有效性

要让DeepSeek有效分析程序，提交的代码需具备良好的组织与描述。若代码碎片化严重、缺少必要注释与上下文说明，模型难以准确把握其功能与意图，分析质量会大打折扣。比如，提交一段抽取数据的代码片段，但未说明数据来源、整体业务流程，模型可能无法判断数据处理逻辑是否合理。同时，对于一些依赖特定环境(如特定库版本、系统环境变量)的代码，若提交时未补充相关环境信息，模型分析出的问题可能无法复现或解决，增加问题排查难度。

5. 分析结果的甄别与应用

DeepSeek返回的分析结果丰富多样，包含明确错误提示、模糊风险预警、多种优化方案等。其中，对于明确的语法错误(如 SyntaxError 相关提示)，甄别相对简单，按提示修改即可；但对于逻辑层面的建议(如建议调整循环条件、更换算法)，需要开发者深入理解程序功能、运行场景，判断修改是否会引发新问题。

例如，模型建议将某段遍历列表的 for 循环改为列表推导式以提升效率，但该循环中若包含复杂的异常处理、中间状态维护逻辑，盲目修改可能破坏原有逻辑，这就要求开发者具备扎实的Python编程基础与逻辑分析能力，对结果进行有效甄别和合理应用。

6. 程序实例

(1) 实例场景

以分析一个简单的Python程序为例，程序功能是从列表中筛选出偶数并计算其平方和，代码如下：

```
python
nums = [1, 2, 3, 4, 5, 6]
result = 0
for num in nums:
    if num % 2 == 0:
        result += num  2
print(result)
```

(2) 分析过程

① 代码提交：将上述代码完整提交至DeepSeek分析界面或通过API提交。

② 模型分析反馈：DeepSeek会进行如下分析。

- 语法与逻辑检查：代码语法正确，循环结构清晰，条件判断 num % 2 == 0 能准确筛选偶数，通过遍历累加偶数平方的逻辑实现了功能需求。
- 优化建议：可考虑使用列表推导式先筛选偶数，再结合 sum 函数计算平方和，使代码更简洁高效。优化后的代码示例：

```
python
nums = [1, 2, 3, 4, 5, 6]
result = sum([num  2 for num in nums if num % 2 == 0])
print(result)
```

- 潜在拓展提示：若后续需要处理更大规模的数据，可考虑引入生成器表达式替代列表推导式，减少内存占用，如 sum(num 2 for num in nums if num % 2 == 0)。

③ 结果解读与应用：开发者收到反馈后，理解到原代码功能正确但可优化简洁性。对于小规模数据，列表推导式的写法更简洁直观，可按建议修改；若考虑未来数据规模扩展，生成器表达式的方案也可提前储备。通过这样的分析与应用，提升代码质量与开发效率。

综上，利用DeepSeek分析Python程序是一种助力开发的有效方式，但需清晰认识其知识点、把握重难点、规避易错点，合理运用模型反馈提升程序开发与优化水平。

1.4 习题

一、选择题

1. 以下关于Python诞生背景的描述，错误的是(　　)。

A. Python 诞生于20世纪80年代末

B. 由荷兰数学和计算机科学研究所(CWI)的Guido van Rossum设计

C. 设计初衷是解决当时编程语言“过于复杂”或“功能局限”的问题

D. Python的设计完全脱离了ABC语言的影响，是全新独立的语言设计

2. Python 2.x与Python 3.x存在诸多差异，下列不属于两者差异的是(　　)。

A. Python 2.x默认使用 ASCII编码，Python 3.x默认支持Unicode

B. Python 2.x中print是语句，Python 3.x中print是函数

C. Python 2.x支持lambda函数，Python 3.x不支持lambda函数

D. Python社区宣布2020年停止对Python 2.x的支持

3. 下列哪项不属于 Python 的核心特点(　　)。

A. 解释型语言，无须编译可直接逐行执行

B. 必须手动管理内存，否则会出现内存泄漏问题

C. 支持面向对象编程，包含类、对象、封装、继承、多态等特性

D. 拥有丰富的标准库和第三方库，可覆盖多领域开发需求

4. 在Python Web开发领域，常用的框架是(　　)。

A. NumPy、Pandas　　B. Django、Flask

C. TensorFlow、PyTorch　　D. Pygame、Tkinter

5. 关于 Python 自带开发环境 IDLE 的说法，正确的是(　　)。

A. IDLE 仅支持交互式开发模式，无法运行.py程序文件

B. 在IDLE的交互式模式中，输入代码后需保存为.py文件才能执行

C. 执行IDLE中编写的.py程序文件，可通过Run→Run Module命令或按F5键实现

D. IDLE 的集成开发环境模式仅能编写简单代码，无法完成复杂程序开发

6. 下列对PyCharm的描述，错误的是(　　)。

A. PyCharm是JetBrains公司开发的Python专用IDE，支持跨平台开发

B. PyCharm社区版是免费的，专业版是收费的，且专业版功能更全面

C. 即使不配置Python解释器路径，PyCharm也能正常运行Python代码

D. PyCharm支持代码高亮、智能提示、调试、版本控制等开发辅助功能

二、填空题

1. Python 1.0版本于_____年正式发布，引入了lambda、map、filter和reduce等_____编程特性。

2. Python是_____型语言，运行时由解释器逐行读取并执行代码；而C、C++属于_____型语言，需先将整个源程序编译成机器语言才能运行。

3. 使用IDLE的交互式开发环境时，输入_____可以退出Python环境；在Windows系统中，通过按_____键打开“运行”窗口，输入cmd可进入命令行界面检验Python安装情况。