高等院校计算机应用系列教材

人工智能程序设计

缑 锦 应 晖 主编

清華大學出版社
北 京

内 容 简 介

本书是一本介绍如何使用 AI 程序员辅助进行 Python 语言程序设计的通识教育类实验实践性教材，旨在为初学程序设计的读者提供 AI 辅助程序设计的解决方案。

本书共 12 章及附录，其中正文共分为三部分：第一部分包含第 1 章，作为概述、导论部分；第二部分包含第 2 章至第 10 章，作为基础编程部分，介绍了一些基础的程序编写方法；第三部分包含第 11 章和第 12 章，作为进阶编程部分，介绍了在摄像头获取的视觉视频中进行程序编写的方法。最后的附录介绍了与本书配套的 AI 辅助程序设计实验平台的使用方式。

本书通俗易懂，提供了几十个人工智能程序设计 (Python 语言) 的应用实例，帮助各个专业方向的读者学习和领悟人工智能 AIGC 辅助程序设计的方法。希望各专业的学生通过对本书的学习学会数据可视化、操作办公文档、NLP 统计过程、算法过程分析对比、数字图像处理、语音处理、人机可视化交互等内容，顺利完成大学 4 年程序设计的准备工作。

本书可以作为广大本科、高职院校人工智能程序设计通识课教材，也可作为初中、高中信息课上介绍程序设计方法的进阶性教材。

图书在版编目(CIP)数据

人工智能程序设计 / 缑锦, 应晖主编. -- 北京 : 清华大学出版社, 2026. 2. -- (高等院校计算机应用系列教材). -- ISBN 978-7-302-70720-2

Ⅰ. TP18

中国国家版本馆CIP数据核字第2026C535A6号

责任编辑：王　定
封面设计：周晓亮
版式设计：思创景点
责任校对：成凤进
责任印制：沈　露

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：河北龙大印务有限公司
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张：**15　　**字　　数：**346千字
版　　次：2026年2月第1版　　**印　　次：**2026年2月第1次印刷
定　　价：79.80元

产品编号：114868-01

前 言 PREFACE

人工智能是计算机科学与技术和信息技术的重要分支，致力于研究与开发用于模拟、延伸和扩展人类智能的理论、方法、技术和应用系统。人工智能技术涵盖了机器学习、深度学习、自然语言处理、计算机视觉等多个子领域。

2025 年 3 月，《政府工作报告——2025 年 3 月 5 日在第十四届全国人民代表大会第三次会议上》正式提出深入实施“人工智能 +”行动；7 月 31 日，国务院常务会议审议通过《国务院关于深入实施“人工智能 +”行动的意见》；10 月 24 日，中共中央新闻发布会宣布全面实施“人工智能 +”行动，加快人工智能等数智技术创新，强化算力、算法、数据等高效供给。以此为契机，“人工智能 +”行动步入快车道，在各行各业的社会生活中深刻影响并改变着人们的生活和工作方式。

随着该行动的深入开展，越来越多的高校师生开始学习人工智能领域的知识，希望使用“人工智能 +”的思维方式赋能专业拓展、课程改革，希望将人工智能快速应用到某些场景的程序开发中，用轻度的人类智力投入，在人工智能的辅助之下，实现开发目标多、降低个人程序编写工作难度、提升程序编写工作效率等。为满足广大学生、初级程序开发人员及其他读者在此领域的需求，我们编写了《人工智能程序设计》一书，旨在为读者提供一个简单、易用、容易上手的实例程序编写指南。

【本书主要内容】

第 1 章　人工智能程序设计的准备知识，介绍了程序设计的概述性知识；程序设计从无到有，由机器语言到高级语言的发展过程；Python 语言程序设计环境的安装；VSCode 集成开发环境的安装；通义灵码插件的安装；等等。

第 2 章　AI 程序员编程示例，使用奇偶数判断、小虫爬井、蒙特卡罗法求圆周率等小实例，介绍了如何在 AI 程序员辅助下进行包括顺序结构、分支结构、循环结构在内的基本程序设计。

第 3 章　matplotlib 的使用，介绍了数据可视化实现方式，包括饼图、折线图、散点图、柱状图的生成方式。

第 4 章　操作电子办公文档，介绍了电子表格、电子文档的操作方式，包括电子表格、电子文档的读写操作，批量操作电子文档。

第 5 章　综合练习：词频统计，介绍了 NLP 的词频统计方法，包括文本预处理方法、jieba 切词方案、停用词词典的生成方法、统计数据的可视化表达。

第 6 章　算法学习，介绍了各种排序方法的可视化对比方式，包括随机数序列的生成方法，冒泡、选择、插入、写入、归并、快速排序的算法生成，使用 gif 动画方式可视化排序过程的方法。

第 7 章　图片操作基础，介绍了数字图像处理基本方式，包括位图文件的本质，RGB、BGR、ALPHA 通道的意义，打开、保存图片，镂空图片、叠加图片，识别纯色物体，镜像、旋转图片，生成灰度图片、二值图片，改变图片的大小。

第 8 章　声音处理，介绍了音频文件处理方式，包括音频信号的录制、获取音频信号的时域特征和频域特征，进行降噪处理，文字合成语音。

第 9 章　AIGC 工具的 API 使用，介绍了调用 AIGC 工具的 API 方法，包括 DeepSeek 的工作方式，使用 API 从命令行到图形用户界面一步一步生成大语言模型支持的人工智能聊天机器人助手。

第 10 章　综合练习：答题卡辨识，通过综合练习检验读者的学习成果。

第 11 章　综合练习：手势控制，使用人工智能辅助的程序设计方式，对媒体流进行程序设计，包括打开摄像头的方法、保存视频的方法、手部特征点的获取方法、使用手势在视频流中进行交互式编程、使用手势进行截图。

第 12 章　综合练习：脸部识别，介绍了脸部检测、获取脸部特征、脸部特效加持方法、疲劳检测方案、记录颈部活动的小应用生成方案。

附录　配套 AI 程序员平台的使用，介绍了用于人工智能程序设计的线上实验平台的使用方法。

【适用范围】

（1）想利用人工智能辅助程序设计的一般程序员。

（2）希望在大学期间学习程序设计，用于有程序开发需求的文理科学生。

（3）用于辅导高中阶段信息技术课程中编程的进阶性课程。

（4）用于有课程出海、科技出海需求的高校课程开发教师。

（5）用于有程序设计需求的各行各业的非计算机专业程序员社会群体。

【本书特点】

本书内容简单、易懂、容易上手实现。本书在线上实验平台的辅助下，图文并茂，以简单的小实例讲解带领读者进入人工智能辅助程序设计的领域。相信读者能够跟随本书，快速地掌握人工智能程序设计的方法，快速开发即用即抛的小型应用程序，以满足日常生活、工作等方面的需求。

【鸣谢】

本书获华侨大学教材建设项目资助，由缑锦、应晖担任主编，华侨大学计算机科学与技术学院的教师团队共同编写。在本书的编写过程中，编者参阅了一些参考文献资料，得到了众多学界前辈、同仁及师友的悉心指导。蚂蚁科技集团股份有限公司、易联众信息技术股份有限公司为本课程开发了线上人工智能程序设计实验、实训平台。在此表示由衷的感谢！

【教学资源】

本书配套有教学大纲、电子课件、实验素材、实验代码、配套 AI 程序员平台使用方法等教学资源，读者可扫描下列二维码获取。

教学大纲　电子课件　实验素材　实验代码　配套 AI 程序员平台使用方法

编者

2026 年 1 月

目 录 CONTENTS

第1章

人工智能程序设计的准备知识

程序设计语言用于编写各种类型的软件和应用程序。计算机程序设计语言是人与计算机交流的桥梁，是一种可以同时被人与计算机识别，被计算机执行的语言。简单来说，计算机程序设计语言就是人对计算机下达的命令的集合。计算机程序设计语言的作用在于定义计算机程序的结构和行为。人类通过定义这些特殊的语法和语义规则，描述计算机程序的逻辑结构、数据处理过程以及与外部环境交互数据的方式。

程序设计语言的出现极大地提高了软件开发的效率和质量。通过使用合适的程序设计语言，程序员可以更快地编写出满足用户需求的软件。程序设计语言提供了严格的类型检查和错误检测机制，可以在编译阶段就发现潜在的问题，提高软件的开发速度和开发效率。

计算机程序设计语言的出现，促进了软件开发过程中程序员与程序员之间的交流。程序设计语言为软件开发团队提供了统一的标准和规范。通过使用相同的程序设计语言，不同的程序员个体之间可以更好地理解对方的思想，在此基础上程序员团队能够协同合作，共同开发出复杂度更高、逻辑更缜密、业务流和运行效率更高、质量更好的应用软件。

1.1 程序设计语言的发展阶段

程序设计语言的历史可以追溯到计算机诞生之初。随着计算机的发展，程序设计语言不断深化、发展，并历经了三个阶段，即机器语言、汇编语言、高级语言。本节将介绍程序设计语言的演变过程。

1.1.1 机器语言

机器语言，也称为机器码，是计算机处理器可以直接识别和执行的指令的集合。机器语言是一种低级语言，由一组纯粹的二进制数字组成，每一条指令都对应着处理器可以直接进行的操作。简言之，机器语言是计算机硬件的唯一“母语”。在计算机的世界中，只存在 0 和 1。一个 0 或者一个 1 被称为 1 位 (1 bit)；8 位二进制数通常被称为 1 个字节。

机器语言的历史可以追溯到计算机产生之初。早期的计算机使用的是真空管和继电器等元件，这些元件只能理解电信号的开和关，因此只能用 0 和 1 来表示指令和数据。随着科技的进步，计算机硬件变得更加复杂，但是，不管是电子管、晶体管、集成电路，还是超大规模集成电路，机器语言仍然是直接操作硬件的唯一途径。

在早期的计算机上进行程序编写的时候，程序员需要深入了解计算机硬件的结构和工作原理，然后直接编写机器语言指令来控制计算机硬件。这种编程方式非常复杂，程序员不仅需要具有非常丰富的硬件知识、熟记操作数与被操作数的规则，还需要具有非常好的耐心，细心地应对海量的 0 和 1。

比如，在早期的计算机上，进行两个数字的加法运算的时候，需要编写的机器语言如下。

(1) 将一个字节的数据从内存加载到寄存器中：

00000001 00000100 00100101

(2) 将另一个字节的数据从内存加载到寄存器中：

00000001 00000101 00100110

(3) 将两个寄存器中的数据相加，结果放到第三个寄存器中：

00000101 00000000 00000100 00000101

(4) 将结果输出至内存：

00000011 00000000 00000110

很简单的 $a + b$ 求和的操作，转化为完整的二进制机器语言代码如下。

00000001000001000100101

00000001000001010100110

00000101000000000000010000000101

00000011000000000000000110

很明显，这样的编程方式对于人类来说难学、难记、难理解、难纠错、难以交流。随着科技的进步与发展，虽然只有 0 和 1 才能被计算机理解和执行，但是这样的编程方式注定会被其他编程方式所取代。

1.1.2 汇编语言

机器语言编程极其复杂且极易出错，为了提高编程效率和减少错误，当时的计算机科

学家设计了一种“符号地址”的概念，用字母或者简单的单词来替代机器语言中出现的一长串的 0 和 1。

1952 年，IBM(国际商业机器公司) 开发出世界上第一款商用计算机 IBM 701。为了推广这款商业计算机，IBM 为其开发了一套汇编语言，汇编语言开始正式进入计算机程序设计语言的发展历史。但是由于早期的计算机并没有统一的生产标准，早期的汇编语言并不通用，不同的计算机需要使用专用汇编语言。20 世纪 60 年代，美国国防部资助了一项名为标准化汇编语言 (Standard Assembly Language) 的项目，由此，创立了世界上第一种通用的、可以在该国所有军用计算机上运行的汇编语言 (Portable Assembly Language，PAL)。

汇编语言是一种低级语言，使用助记符 (mnemonic) 来代替机器语言中的操作码，相比机器语言更容易理解和书写。汇编语言的基本单元是指令 (instruction)，每条指令对应计算机硬件上的一条操作，如加法、乘法、内存读写等。这些指令由特定的助记符表示，如 LD 表示数据传送指令，ADD 表示加法指令。在汇编语言中，程序员需要直接使用这些助记符来编写程序。汇编语言代码可以通过汇编器 (assembler) 转化成机器语言，以便计算机能够理解和执行。汇编语言转化成机器语言的过程是非常复杂和精细的，需要考虑不同硬件架构的特性和指令集的差异。因此，汇编器通常需要针对特定的硬件平台进行定制开发，以确保生成的机器语言能够正确地被计算机执行。

比如，当需要进行两个数字的加法运算时，使用 8 位微处理器 Z80 的汇编语言是这样实现加法操作的。

(1) 将一个字节的数据从内存加载到 a 号寄存器中：

```
LD a, 10
```

(2) 将另一个字节的数据从内存加载到 b 号寄存器中：

```
LD b, 20
```

(3) 两个寄存器中的数据相加，结果放到 a 号寄存器中：

```
ADD a, b
```

(4) 将结果输出至内存中：

```
LD (0010h), a
```

(5) 停止程序：

```
HALT
```

$a + b$ 求和的操作转化为完整的 Z80 汇编语言代码如下。

```
LD a, 10
LD b, 20
ADD a, b
LD (0010h), a
HALT
```

其中，LD 是 load 的缩写，表达导入的意思；ADD 表达相加的意思；HALT 表达停止

的意思。只要略懂 Z80 汇编语言，甚至略懂英语，就能了解这 5 行代码的意思。

尽管汇编语言相对于机器语言来说已经是前进了一大步，为人类提供了编写程序的便利，但是它依旧存在一些缺点：①汇编语言与特定的硬件体系结构相关，编写的汇编代码通常不具备通用性，不能在不同的处理器上直接运行；②汇编语言相对于高级语言来说依旧显得复杂晦涩，依旧难读、难懂、难以沟通；③汇编语言要求程序员对计算机体系结构有较深入的了解，若程序员不了解某种型号的计算机，则无法编写适合该型号计算机使用的汇编语言程序；④汇编语言往往与机器指令是一一对应的关系，使用汇编语言编写的代码依旧比较冗长。因此，汇编语言逐渐被高级语言所取代。

但是近年来，随着物联网 (IoT)、人工智能 (AI) 等新兴技术的发展，汇编语言又开始受到人们的关注。在一些处于终端位置的传感器或者执行器上，设备的运算能力往往十分有限，但是又需要在这些资源非常有限的设备上进行高效编程。在某些工业设备自动控制开发场景下，人们开始重新使用汇编语言来进行设备开发。因此，汇编语言还没有完全退出历史的舞台。

1.1.3 高级语言

经过十多年的发展，程序设计语言由编程语言进化发展为高级语言。高级语言为程序员提供了更接近自然语言的编程方式。时至今日，使用高级语言已经成为计算机编程的主流方式。程序员能够更加灵活地使用高级语言进行程序编写工作。

1. 高级语言的出现

20 世纪 50 年代，尽管已经有了汇编语言的助力，但是开发程序依旧是一件耗时耗力且非常复杂的事情。程序员依旧需要对冗长晦涩的代码进行编写，工作效率依旧十分低下。为了简化编程过程，人们开始尝试开发高级语言。高级语言是相对于低级语言而言的，它更加抽象和易于理解。在当时的计算机专家的设想中，使用高级语言编写程序时不需要考虑底层硬件的细节；编写好的程序可以跨平台运行；代码本身具有更好的可读性和可维护性。在这样的背景下，第一批高级语言开始进入应用领域。

(1) Fortran。1957 年，IBM 推出了世界上第一种通用高级语言公式翻译 (Formula Translation，Fortran)，它是一种专门用于科学计算的编程语言。Fortran 的应用标志着高级语言时代的到来，它极大地简化了编程工作。由于 Fortran 简单易懂，原本不会计算机程序编写的科学家也能很快上手进行编程。Fortran 的诞生为后续的高级语言提供了思想基础与发展方向。

(2) Lisp。1958 年，Lisp(List Processing) 由约翰·麦卡锡教授在麻省理工学院开发而成。Lisp 最初是为了研究人工智能而设计的。Lisp 的设计理念是以列表 (list) 为基础，允许将代码和数据视为同一种东西，进行元编程，这使得 Lisp 具有很强的灵活性和表达能力，成为一种专门用于人工智能领域的编程语言。Lisp 语言具有强大的符号处理能力和递归特性。在人工智能领域，Lisp 曾经是主流编程语言，因为它对符号处理和逻辑推理有着天然的支

持。Lisp 语言对现代编程语言产生了深远的影响，如 Javascript 和 Python 等语言就借鉴了 Lisp 的一些特性。

(3) COBOL。1959 年，通用面向商业语言（Common Business-Oriented Language，COBOL）问世。COBOL 是一种面向商业企业的计算机编程语言。1959 年，美国国防部召开专门会议，讨论建立通用商业语言的要求和可能性，确定了这种语言的基本设计思想和应具有的特点。COBOL 最初是为了商业数据处理而设计的，因此在金融领域得到了广泛的应用。COBOL 是一种目前依然在用的语言。1974 年，美国国家标准化协会（America National Standards Institude，ANSI）对 COBOL 做了补充，公布了 ANSI COBOL X3.23-1974。到了 1978 年，该版本的 COBOL 语言被国际标准化组织（International Organization for Standardization，ISO）推荐为国际标准文本，最终修订为 ISO COBOL-78 版本。在美国，很多政府统计救助金的程序、保险公司计算赔偿金的系统都是使用 COBOL 编写的。COBOL 系统由于出得很早且比较稳定，至今还有 43% 的银行在使用，95% 的自动柜员机（Auto Teller Machine，ATM）交易，80% 的面对面交易靠 COBOL 代码支持。将古老代码进行更新转化的成本是十分高昂的。2012 年，澳大利亚联邦银行想要更新自己业务中的 COBOL 代码，雇用了 2 家公司帮忙。最终耗时 5 年，花了 7.5 亿美金，把 7500 万行 COBOL 代码转换成 Java 代码，但在使用新业务逻辑几个月后，因为各种未知因素的框架性故障，不得不更换回原来的 COBOL 代码，再痛定思痛地开发全新代码用于承载未来银行的业务平台需求。

(4) Algol。1958 年，由国际算法语言小组 (International Algorithmic Language Group) 设计和开发了算法语言 (Algorithmic Language，Algol) 的第一个版本。Algol 是一种通用的高级编程语言，对后来很多高级语言的设计产生了深远影响。Algol 的设计目标是提供一种结构清晰、可读性强的编程语言，以便编写和理解复杂的算法与程序。它强调程序的模块化和可重用性，为程序员提供了丰富的控制结构和数据类型，使得编写高质量的代码变得更加容易。Algol 的语法和语义的设计思想、设计方案对后来的许多程序设计语言产生了深远的影响。后来出现的 C 语言和 Pascal 语言就借鉴了 Algol 的许多特性，并在此基础上进行了扩展和改进。Algol 的结构化编程设计也为后来的面向对象编程语言奠定了基础。尽管 Algol 在其诞生之初就受到了广泛关注和认可，但它并没有成为主流编程语言。其主要原因是当时的计算机硬件和软件技术还不够成熟，程序员数量并不多，无法很好地、成规模地支持 Algol 语言。此外，Algol 的语法和规范也被认为过于严格和复杂，这使得学习和使用 Algol 的门槛明显高于 COBOL。

20 世纪 50 年代末至 70 年代初，高级语言得到了迅速发展。随着计算机技术的不断进步，越来越多的高级语言被开发出来。高级语言具有较高的可读性和可维护性。相对于低级语言 (汇编语言)，高级语言更接近人类自然语言，编程方式更符合人类习惯的表达方式，既容易被程序员理解和维护，也容易被其他人理解和修改，有利于团队协作和项目维护。

比如，需要进行两个数字的加法运算的时候，使用高级语言是这样的：

$$c = a + b$$

一般学生容易理解这样操作的含义。相较于机器语言和汇编语言，高级语言的优势十分明显。人工智能的应用与普及离不开高级语言的发展。

2. 编译与解释

编译 (compilation) 与解释 (interpretation) 是两种常见的高级程序语言处理方式。如前文介绍机器语言时所述，向计算机提供机器语言是直接操作计算机硬件设备的唯一途径。程序员按照业务逻辑编写好高级语言源代码后，经过处理，把高级语言转换为机器语言代码指令，再把机器语言代码指令提供给计算机的中央处理器 (CPU)，供 CPU 执行对应指令。以上过程的实现方法和运行方式，不同的高级语言转化方式是有所不同的。

(1) 编译。编译是将整个完整的高级语言程序源代码文件作为输入，经过词法分析、语法分析、语义分析、中间代码生成、代码优化和目标代码生成等多个阶段 (图 1-1)，最终生成可执行的机器语言代码的复杂过程。编译器 (compiler) 是用于执行这一过程的程序工具。通过编译过程，我们可以将高级语言代码转换成可执行的机器语言代码，从而实现高级语言编写的程序在计算机上的运行。编译型语言的开发周期相对较长，但一旦编译完成，程序在运行时通常有较好的性能表现。

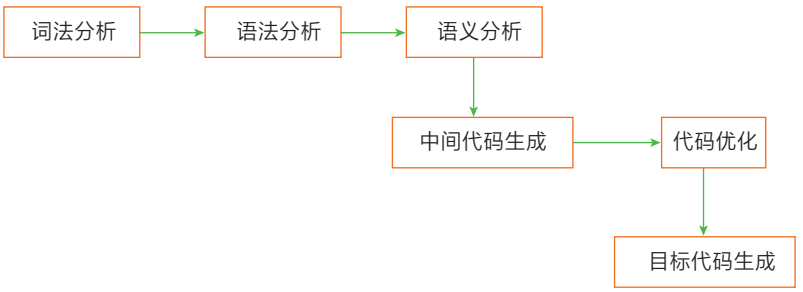

图 1-1　编译过程

① 词法分析是编译过程的第一步。词法分析器将源代码分解成一个个词法单元或标记。这些词法单元是高级语言语法中的最小单位，包括关键字、标识符、运算符、常量和分隔符等。词法分析器会忽略空白字符、空白行或注释等无关元素，生成一个记号序列流进入下一阶段的语法分析。

② 语法分析器会根据语法规则对记号序列流进行语法分析，对记号序列构建语法树 (解析树)。语法树是源代码的抽象表示，它能够分析代码各部分之间的关系，检查代码是否符合语法规则。如果源代码中存在语法错误 (如缺少冒号或缺少配对的括号、引号)，编译器就会报错并指出存在语法错误的代码所在的位置，给出相应的错误信息。

③ 语义分析器会对语法树进行遍历，检查代码中的语义错误并进行类型检查。语义分析器会判断代码中的变量有没有声明，使用是否正确，函数有没有定义，调用是否合法，参数数量和参数类型是否匹配，等等。如果发现了语义错误，编译器也会给出相应的信息。

④ 在通过语义分析之后，编译器会根据语法树生成中间代码。中间代码是介于高级语

言和机器码之间的一种表示形式，它包含了源程序的结构信息和执行逻辑。它通常与平台无关，便于优化，可以被不同目标机器上的后端编译器进一步转换为机器语言代码。

⑤ 代码优化是编译过程中非常重要的一部分。优化器会对中间代码进行优化，以提高程序的性能和效率，减少资源消耗。常见的代码优化技术可以对中间代码进行重写和重组，以减少指令数、内存访问次数、运行时间等。

⑥ 生成目标代码是编译过程的最后一步，它将优化后的中间代码转换成目标机器语言代码。目标机器语言代码生成器会根据目标机器的特性和指令集，进行寄存器分配、指令调度等处理，将中间代码转换成可直接执行的二进制文件或目标文件，以便在目标机器上运行。

常见的编译型语言有 C、C++、Java 等。

(2) 解释。解释是将高级语言程序逐行翻译成机器语言并立即执行的过程。解释方式在每次运行程序时都动态地解释和执行高级语言代码。每次执行解释性的高级语言时，都需要进行翻译和执行操作，这意味着使用解释方式运行的高级语言程序的性能和运行效率比使用编译方式运行的高级语言程序要低。

解释器 (interpretator) 是一种可以直接执行解释性高级语言程序的软件工具。解释器能够逐行读取源代码，将源代码转换成对应的机器语言指令，并且立即执行这些指令。这种实时性的特点使得解释性高级语言在开发阶段具有较好的灵活性，适合快速开发原型或简化复杂任务，可以快速进行代码调试和修改。解释性高级语言通常比编译方式的高级语言慢，性能上存在一定的劣势，每次执行解释性语言的程序，都需要进行翻译和执行操作。在一些对性能要求较高的场景中解释性编程语言可能显得不太适用。

为了克服解释方式的性能劣势，一些解释器引入了即时编译 (JIT) 技术。即时编译是指在程序运行过程中将部分代码转换成机器语言，并缓存起来以备下次使用。这种方式虽然可以在一定程度上提高程序的性能，但是增加了解释器本身的复杂度和内存占用。

常见的解释型语言有 Python、Javascript、Ruby 等。

总的来说，编译与解释是两种不同的程序语言处理方式，它们在程序执行效率、跨平台性和灵活性等方面有所不同。在选择开发语言时，需要根据具体需求和应用场景权衡各种因素，以选择最合适的程序语言处理方式和类型。

1.2 高级程序语言的分类

1.2.1 根据程序设计思想和程序设计范式分类

高级程序语言根据程序设计思想和程序设计范式的不同，可以分为面向过程、面向对象和非过程化三种。

1. 面向过程

面向过程编程的程序设计思想主要指的是将程序分解为一系列具有时间顺序的子问题、子步骤或子过程；子问题、子步骤或子过程都是按照解决问题的时间先后顺序，一步一步地设计程序的。这种编程方式强调程序的执行顺序和每个步骤之间的过程关系，而不是将程序看作一系列对象的集合或者函数的集合。面向过程编程通常用于解决一些简单的问题。

在面向过程编程中，程序员需要清楚地定义每个步骤的执行流程，以及每个步骤之间的数据传递和依赖关系。计算机高级语言从诞生采用的就是面向过程的逻辑思维方式，这种思维方式一直延续到今天。面向过程编程通常使用一些基本的编程逻辑结构来组织程序的执行流程，如顺序结构、选择结构和循环结构。这些结构可以帮助程序员清晰地表达程序的逻辑，从而使程序更易于理解和维护。面向过程编程不但可以使程序更加高效和直观，而且可以减少额外的开销，降低复杂性。

面向过程编程有一些局限性，特别是在面对庞大而复杂的业务逻辑需求，或者需要编程团队进行大规模团队协作开发的情况下。采用面向过程编程逻辑，往往会导致程序结构混乱，难以维护和扩展，更严重的问题是不利于代码重用和后续的模块化开发。

总的来说，面向过程编程是一种重要的编程范式，它在特定的场景下仍然具有一定的优势和价值。然而，在面对复杂和大规模的软件开发问题时，选择合适的编程范式是非常重要的，只有根据具体情况选择合适的编程方式，才能更好地解决问题并提高软件开发效率。

面向过程的编程语言主要有 Fortran、COBOL、Basic、C 和 Pascal 等。

2. 面向对象

面向对象编程是一种以对象为中心的编程逻辑方法，它主要关注的是业务对象的数据属性和行为方法的封装。面向对象的程序设计思想是将行为主体抽象成对象，分析出在业务逻辑中需要被采集、运算的对象属性；在程序中投影映射出行为主体进行的操作逻辑行为方法；在行为主体间进行参数的传递，协作完成整个业务逻辑。

面向对象编程的优点：代码的重用性高、可维护性好、扩展性强、抽象能力强等；可以更好地组织和管理代码，提高代码的复用性和可维护性，非常适用于处理复杂的、大型的项目；面向对象编程可以将复杂的问题分解成多个相互关联的对象，从而更加清晰地理解和解决问题；面向对象编程可以提高团队合作的效率：不同的开发人员可以独立地开发和测试不同的对象，最后将所有的对象组合在一起形成一个完整的程序；面向对象编程更加符合人类对现实世界的认知方式，因此更容易理解和使用。

面向对象编程有三个基本编程逻辑，即封装、继承和多态。封装是指将行为主体所需要的数据和操作数据的方法放在一起开发，形成一个独立的实体，外部只能通过指定的接口来访问对象的数据和方法，从而有效地隐藏对象的内部实现细节，提高代码的安全性和可靠性。继承是指一个对象子类可以继承另一个对象父类的属性和方法，从而可以在不改

变原有代码的基础上进行功能扩展。多态是指同一操作作用于不同的对象会产生不同的行为，以提高代码的灵活性和可复用性。

面向对象的编程语言主要有 C++、Java、Python 和 Ruby 等。

3. 非过程化

非过程化编程是一种不依赖固定步骤和顺序的编程范式，它主要关注问题的描述和解决方法的表达。非过程化编程通常使用逻辑语句和规则来描述问题和解决方法，而不依赖特定的执行顺序。在非过程化编程中，程序员不需要显式地定义每一个步骤和流程，而是通过声明所需的结果，让计算机自行推导出实现这些结果的步骤和流程。非过程化编程语言的特点之一是强调的不是“怎样做”而是“做什么”。这意味着程序员可以专注于描述问题的本质和所需的结果，而无须关心具体的实现细节。这种思维方式有助于提高代码的可读性和可维护性，降低程序的复杂度。

非过程化编程语言主要有 Haskell、Lisp、Prolog 和 SQL 等。

1.2.2 其他分类方法

1. 根据语法、类型检查、运行时的行为等分类

高级语言根据语法、类型检查、运行时的行为等可以分为动态语言和静态语言。这也是一种常见的计算机高级语言分类方法。

(1) 动态语言是一种在运行时可以改变数据变量类型和对象行为方式的编程语言。动态语言变量的类型不是在编译时确定的，而是在运行时确定的。动态语言通常具有灵活的语法，可以更快地实现功能，具有较高的可读性和灵活性。动态语言通常被用于 Web 开发、数据分析、人工智能等领域。

常见的动态语言有 Python、Javascript 和 Ruby 等。

(2) 静态语言是一种在编译时就执行类型检查的编程语言。这意味着变量的类型需要在编写代码的时候就预先明确声明，在编译时编译器的语义分析机制会进行变量类型检查。静态语言通常具有更严格的类型系统和更高的性能，这使得它们在开发大型、复杂的项目时，特别是在进行团队协作时更加可靠和高效。

常见的静态语言有 Java、C++ 和 C# 等语言。

2. 根据高级语言的使用场景分类

高级语言根据使用场景可以分为通用编程语言、脚本语言、Web 开发语言、嵌入式系统语言、并发编程语言等。

随着计算机技术的不断进步，人类程序员可以使用各种各样的高级语言进行编码工作。本书随后将介绍的编程方式是采用人工智能的生成内容 (AIGC) 功能，由人类给出描述，而由人工智能程序员 (以下简称 AI 程序员) 确认人类需求之后，给出代码建议的。

1.3 AI 程序员的基本原理与功能

1.3.1 AI 程序员的基本原理

AI 程序员基于深度学习技术，通过对以大规模程序代码文本为主体的数据集进行训练，实现从自然语言描述的应用需求过渡到生产对应程序代码产品的自动化转换。虽然有时 AI 程序员提供的代码中存在一些错误，但是，以 AI 程序员为工具的人类程序员，在合理的交互策略下，是可以比较容易地发现与修改这些错误的。目前，业界的观点是：AI 程序员的出现是软件工具的一次革命性进步，它能够极大地提高软件公司的生产效率。

AI 程序员的本质是一种自动化代码生成器。它能够根据程序员提供的需求，自动生成代码。相比人类程序员的工作，AI 程序员的使用可以提升开发效率，既降低了开发过程中的人力成本与时间成本，也降低了程序员的门槛。这一技术的背后依赖强大的大规模语言模型和深度学习模型。

AI 程序员的训练数据通常由自然语言描述和实际程序代码两个部分组成。AI 程序员工具的开发者会收集大量公开可用的代码库 (如各种开源项目、编程文档以及技术博客等)，将其与自然语言描述 (一般是描述性的注释内容) 配对。得益于 Transformer 架构 (后续章节中会详细介绍) 的使用，AI 程序员学习到如何将自然语言需求映射到具体的代码中实现。

Transformer 架构作为一种革命性的神经网络模型，不仅在自然语言处理领域取得了卓越的成果，还被广泛应用于其他领域，如计算机视觉 (computer vision，CV)、语音处理和推荐系统等。Transformer 架构凭借其强大的序列建模能力和上下文理解能力，极大地提升了 AI 程序员的业务能力。阿里云推出的通义灵码就是基于 Transformer 架构开发的 AI 程序员应用。通义灵码支持多种编程语言。在集成开发环境中使用通义灵码插件，可以准确地理解程序员自然语言描述需求的细节，可以联系上下文对代码进行修改，可以生成逻辑正确、语法规范的代码。

1.3.2 AI 程序员的功能

1. 代码生成

代码生成是 AI 程序员模型的重要功能之一，其目的是根据自然语言描述需求，自动生成符合该需求的代码。例如，人类程序员可以输入“生成一个图形人机交互界面”，AI 程序员就会输出对应的代码。

2. 代码补全

代码补全是人类程序员期望 AI 程序员辅助的日常工作中常用的功能。AI 程序员可以

根据已有的代码片段预测并补全缺失部分，从而提高编程效率。以往的集成开发环境会以句中的函数为单位，建议一些代码方式。AI 程序员时代，代码补全的意义则是通过上下文信息和代码语法结构，成段且大量地进行功能性代码补全。其效率远超传统的集成开发环境中的人工编程方法。Transformer 架构使得 AI 程序员能够理解自然语言需求中的一个或多个关键点，从而生成符合逻辑的程序段补全代码。

3. 程序错误检测与修复

程序错误检测 (找 BUG) 是人类程序员的必修课，一段程序不出 BUG 的可能性无限趋近于 0。传统的集成开发环境提供了一定的程序错误检测功能，能够在一定程度上帮助人类程序员进行程序错误的寻找与修复。AI 程序员在寻找与修复程序错误方面的高效是人类程序员难以比拟和想象的。通过分析代码中的语法错误、逻辑漏洞或潜在问题，AI 程序员可以提供诊断信息，并自动生成修复建议。Transformer 架构可以通过学习大量高质量代码样本，掌握丰富的语法规则和编程模式。AI 程序员不仅能够发现显性错误，还能够识别潜在问题，提供性能优化建议或安全漏洞修复建议。而且，AI 程序员将人类程序员逐行寻找程序错误的工作方式进化为语句段、语句函数甚至函数库范围内的诊断与修复。

4. 跨语言编程支持

软件开发可能涉及多种编程语言之间的协作，如将 Python 语言代码转换为 Java 语言实现等。AI 程序员通过学习不同语言之间的映射关系，可以实现跨语言代码的生成与转化。计算机语言是形式化的，每种计算机语言都有对应的关键字或者保留字。实际上，AI 程序员就是使用这些关键字和保留字进行千变万化的程序编写工作的。相较于人类自然语言，计算机语言对应的关键字十分有限，远没有人类自然语言那么复杂。计算机不同语言之间的转化难度甚至低于人类自然语言转化为计算机高级语言的难度。

以上是 AI 程序员能够提供的四大功能。在 Transformer 架构下，计算机训练了大量数据，建立了自然语言描述性文本与代码块之间的联系。这些联系使得 AI 程序员能够更快、更好、更高效地“理解”人类程序员希望达成的程序效果。在人类程序员与 AI 程序员的共同协作之下，软件开发更加简单了。

在本书后续章节中，我们将介绍如何使用 Python 语言作为编程语言，使用 VSCode 集成开发环境中的通义灵码插件来实现零基础编程。

1.4 Python 语言简介

1.4.1 Python 语言的产生

Python 语言正式诞生于 1991 年，由荷兰人吉多 • 范 • 罗苏姆 (Guido Van Rossum)

(图 1-2) 研发。英国的一部电视喜剧集 *Monty Python's Flying Circus*(《蒙提派森的飞行马戏团》) 在当时颇为流行。吉多很喜欢追这部剧，因此吉多给他研发的语言命名为 Python，实际上是希望 Python 语言能够像这部剧集一样在西方世界流行起来。在 30 多年后的今天，Python 的确成了目前世界上流行的高级语言之一。

图 1-2　吉多·范·罗苏姆

Python 语言的诞生很有戏剧性。吉多在阿姆斯特丹的荷兰数学与计算机科学研究所工作的时候，一直对当时研究所开发的 ABC 教学语言颇有微词，认为 ABC 语言存在很多不足。所以他准备说服他的同事，一同对 ABC 语言进行改进。但是他的同事一直下不了决心去做这件艰巨的事情。1989 年圣诞节假期的时候，吉多突然支走了自己的家人，一个人待在自己的小屋里。他灵感爆棚，决定利用假期大干一场，着手改造 ABC 语言。吉多经过几个不眠之夜，干脆抛弃了 ABC 语言的限制，自立门户，将这个 1989 年圣诞节的灵感汇聚成了一种简单而易于阅读的编程语言——Python。当吉多把自己的成果展示给同事的时候，他的同事也觉得这个临时起意的圣诞节项目十分有趣。1991 年，在吉多和他的同事的努力之下，Python 的第一个正式版本 Python 0.9.0 发布了。当时的 Python 毫无疑问是小众语言，吉多决定把 Python 开源，放到了互联网上，供感兴趣的程序员自行下载研究。对比当时的主流编程语言，Python 的语法规则更简单，使用灵活，易于上手，有利于一些小项目的开发。1994 年，Python 1.0 版本发布，Python 开源带来了意想不到的影响力。有人开始利用 Python 开发一些小的练手项目，并把这些开发出来的练手小项目放到 Python 的网上社区，供大家学习使用。随着时间的推移，Python 的网上社区积累的项目或者开发框架越来越多。

为了适应时代的发展，Python 网上社区开始着手对 Python 进行升级改造。2000 年，Python 2.0 版本发布。Python 2.0 首次支持 Unicode 字符串，Python 开始进行国际化改造，可以在程序的字符串中使用非英语的语言字符；它引入了当时流行的新的异常处理机制，使得异常中断的处理更加强大；它优化了标准库中的函数，提升了稳定性；由于同时改造了解释器，Python 2.0 版本可以引入新的内存机制，提升了运行效率；它引入了许多新的模块，包括集成的 XML 处理模块、网络编程模块、数据库访问模块；等等。在吉多和 Python 网上社区的各位“大神”的共同努力之下，面向更广泛、更挑剔的世界范围的用户群体的新 Python 网上社区开始吸收新的代码项目，为 Python 语言在各个领域的代码任务提

供更加强大和丰富的工具、框架及支持。这标志着 Python 语言迈向了一个新的发展阶段。Python 2.0 版本的发布使得 Python 语言逐渐成为一种备受关注和青睐的编程语言，推动了 Python 语言在科学计算、Web 开发、系统编程等领域的广泛推广应用。

2008 年，Python 3.0 版本正式发布。它为了让专业程序员能够进行更随意的应用框架、机器学习运算模块的编写，减少框架和模块使用中出现的语法错误，使用面向对象的编程方法，重新设计了 Python 解释器。Python 解释器对语法树结构进行了大规模修改，在语法中增加了寻括号匹配机制，严格了语法与语义的限制；移除了一些可能导致 32 位、64 位计算机错误响应的旧的程序内存分配、触发和运行机制；根据新的语法匹配机制修改了标准库中的所有函数，引入更新了的异常处理机制；等等。但是由于步子走得太大，旧的 Python 2.x 程序与 Python 3.x 程序不兼容。以往积累的 Python 2.x 用户与代码资源无法无缝转移到 Python 3.x 环境中。这导致了很长一段时间内 Python 2.x 与 Python 3.x 同时存在，并互相争夺人力资源。直至 2020 年 1 月 1 日正式推出 Python 3.0 版本， Python 2.0 资源被逐步迁移替换成 Python 3.x 资源，Python 2.x 才宣布停止更新。

1.4.2 Python 语言源代码文件的执行

当 Python 语言源代码文件 (扩展名为“.py”的文件) 被导入或执行时，解释器会将源文件编译为字节码，并将字节码保存到与源文件对应的“.pyc”文件中。具体来说，生成“.pyc”文件的过程如下。

(1) 当 Python 解释器第一次导入或执行一个“.py”文件时，它会检查是否存在对应的“.pyc” 文件。

(2) 如果存在“.pyc”文件并且其时间戳与对应的“.py”文件一致，那么解释器会加载并执行“.pyc”文件中的字节码。

(3) 如果不存在对应的“.pyc”文件，或者其时间戳不一致，解释器会自动将“.py”文件编译为字节码，并保存到“.pyc”文件中。

(4) 下次再导入或执行相同的“.py”文件时，解释器会直接加载“.pyc”文件中的字节码，而不需要重新编译源文件。

1.4.3 Python 语言解释器

Python 语言有多种解释器，如 CPython、PyPy、JPython 等。

“.pyc”文件是由 CPython 解释器在运行 “.py”文件时生成的。CPython 解释器的主要功能是提高 Python 程序的执行效率。尽管 Python 是解释性语言，但是 Python 解释器中添加的编译功能能够将源代码转换为字节码并保存到编译文件中。这样可以减少运行程序时的准备时间，从而加快程序的运行速度，提高程序的加载速度，实现跨平台移植。

1.4.4 Python 语言的应用

Python 是一种通用语言，其目标是服务编程任务。以下是几个使用 Python 的知名机构与场景。

(1) 美国国家航空航天局 (NASA) 在处理航天器和卫星收集的大量科学数据时使用 Python 进行数据分析和可视化。

(2) 谷歌大量使用 Python 来开发人工智能和机器学习模型，进行自然语言处理。

(3) 雅虎使用 Python 挖掘和分析用户行为、广告效果、搜索模式。

(4) 豆瓣所有的业务都是使用 Python 开发的。

(5) 大家常用的网络下载 BitTorrent 是 Bram Cohen 使用 Python 编写的。

(6) 游戏公司使用 Python 编写游戏，如《战地 2》《文明 4》《模拟人生 4》等。

(7) 美国纽约证交所使用 Python 来编写自动化交易系统和算法交易策略。

由以上例子可以看出，Python 的使用范围是非常广泛的。不同的行业、不同的领域 (数据科学和人工智能、网络应用开发、自动化运维、科学计算和工程计算、游戏开发等多个领域) 都可以使用 Python 作为业务逻辑的编程语言。

1.4.5 Python 语言与其他语言的比较

Python 和 C、C++ 一样，都是高级语言，其区别在于，C、C++ 使用编译器把源程序编译为计算机目标文件，然后让计算机执行这个目标文件，而 Python 使用的则是解释器。Python 源代码完成之后，解释器逐条将源代码翻译为计算机可执行的机器码，并让计算机逐条执行。因此，Python 的运行效率要低于 C、C++。

Python 是一种面向对象的计算机语言。Python 被更多的小型开发者使用进行敏捷开发或小项目框架开发。Python 中的数据类型是基于对象创建的，更接近人类的思维模式。在对象类中，定义有对象属性与对象方法，因此，Python 的可重用性得到了保障。同一类型的多个对象能使用同样的对象方法进行操作。

Python 是一种动态语言，相较于其他编程语言，Python 语法限制不那么严格。Python 是一种脚本语言，具有清晰简洁的语法和结构，易学易用。这使得它容易学习和上手，即使是初学者也能快速掌握。

总之，Python 非常适合对编写程序感兴趣的读者进行入门学习。

1.5 Python 语言编程环境

进行程序设计的时候，需要首先安装编程语言的编程环境，然后安装该语言的集成开发环境。本节将介绍如何安装 Python 的编程环境。

1.5.1　下载与安装 Python

在 Microsoft Edge 浏览器的地址栏中输入 Python 的官网地址，按 Enter 键，进入 Python 官网。本书中使用的 Python 环境为 Python 3.13.5。在 Download 选项卡下即可下载该编程环境的安装包。本书以 Windows 11 系统环境为例，介绍 Python 的下载与安装方法。

(1) 进入 Python 官网的欢迎页面，如图 1-3 所示。

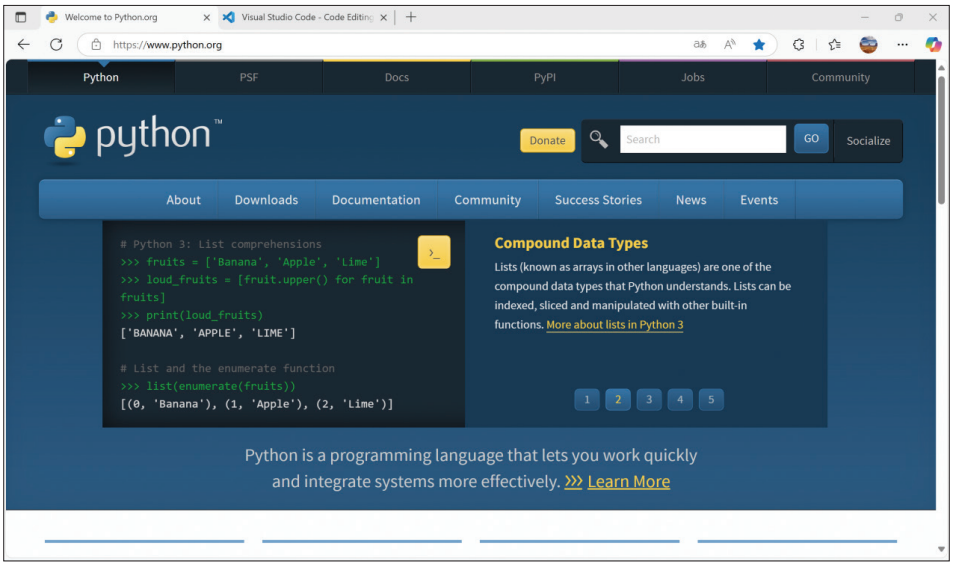

图 1-3　Python 官网的欢迎页面

(2) 在 Downloads 选项卡下选择对应的计算机操作系统。如使用 Windows 操作系统，可以在导航栏下选择“Windows”(如果使用 Mac 系统，可以选择“macOS”)，如图 1-4 所示。

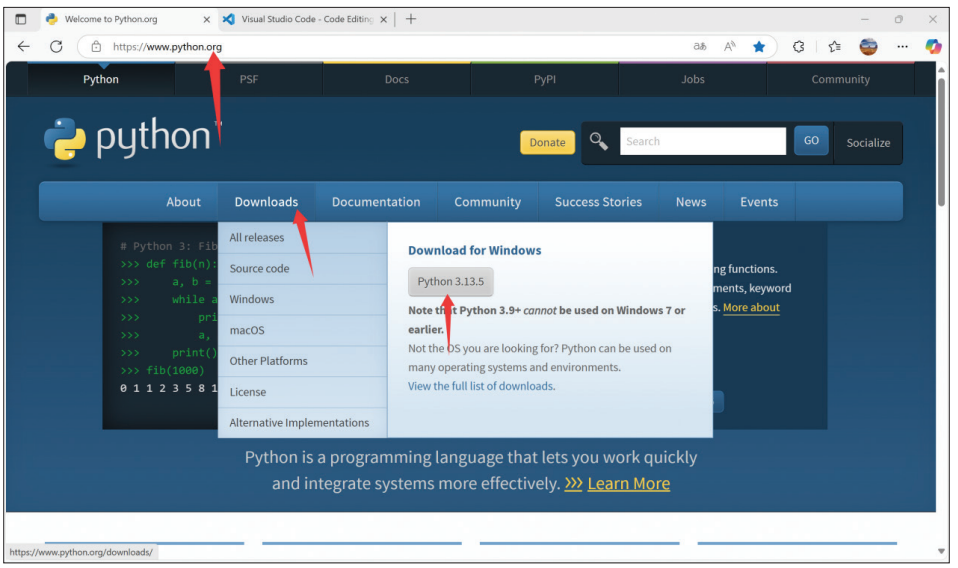

图 1-4　选择下载对应操作系统的文件

注意：

在本书开始编写的时候，Python 最新版本为 Python 3.13.5，这个版本已经无法在 Windows 7 操作系统上使用了。用户可以选择下载最新版的 Python 安装包，或者根据自己的计算机操作系统，选择以前版本的 Python 安装包。

(3) 如果希望下载最新版本的 Python 安装包，单击“Python 3.13.5”按钮 (图 1-5)，开始下载。下载完成后，将在 Edge 中出现图 1-5 所示的下载提示。

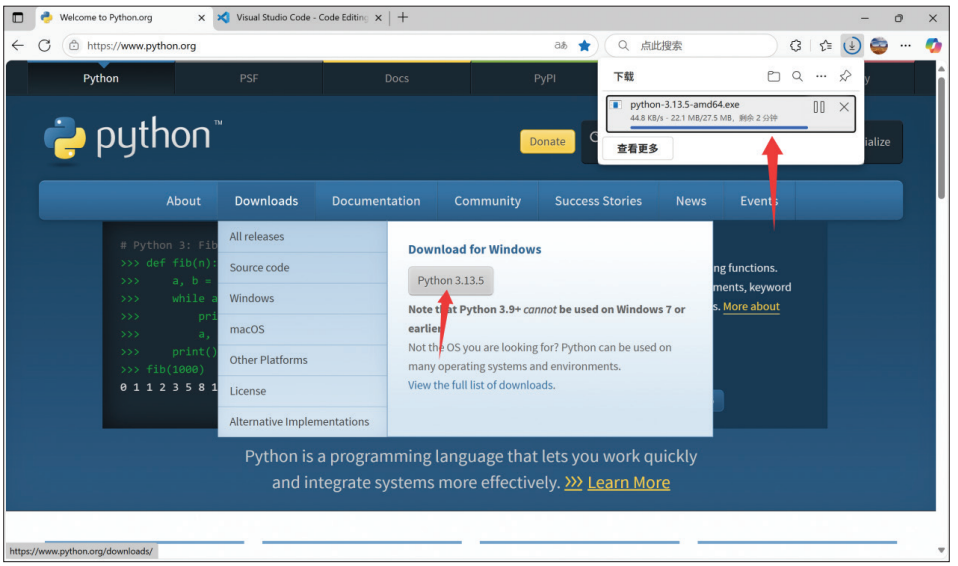

图 1-5　下载 Python 编程环境

(4) 单击“打开文件夹”图标，即可进入下载文件夹，如图 1-6 所示；或者直接单击“打开文件”，开始安装 Python 编程环境。

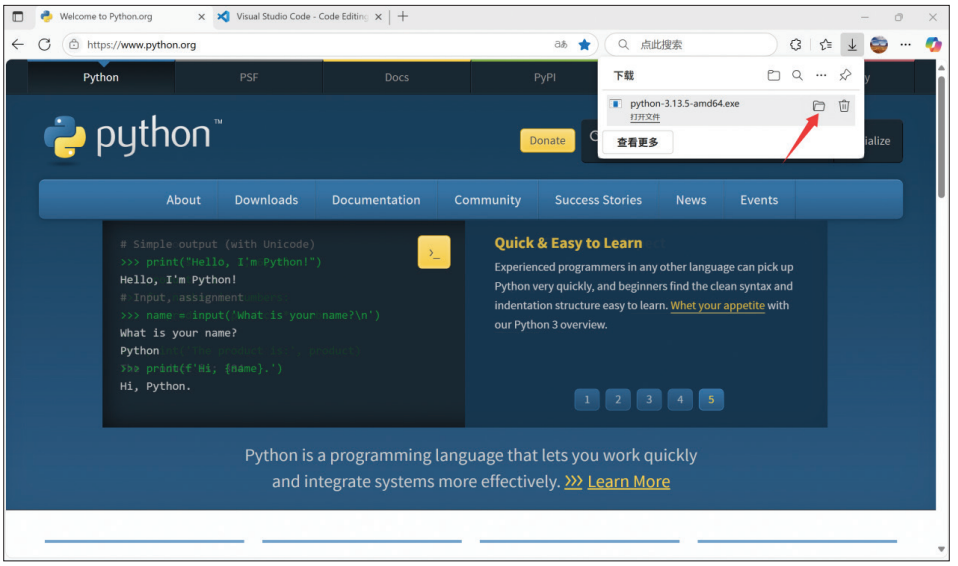

图 1-6　下载完成

进入下载的文件夹后，双击已经下载好的 Python 安装文件进行安装，如图 1-7 所示。

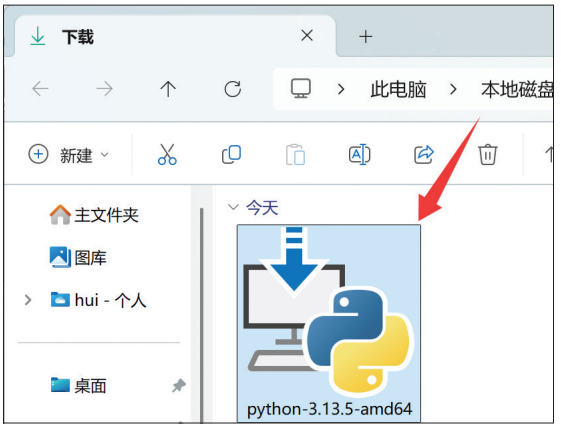

图 1-7 保存在计算机硬盘上，下载文件夹中的 Python 安装文件

如果想下载以往的 Windows 系统的旧版本 Python 语言程序开发环境，可以访问 Python 官网，在网页上选择希望使用的旧版本 Python 安装文件下载安装，如图 1-8 所示。

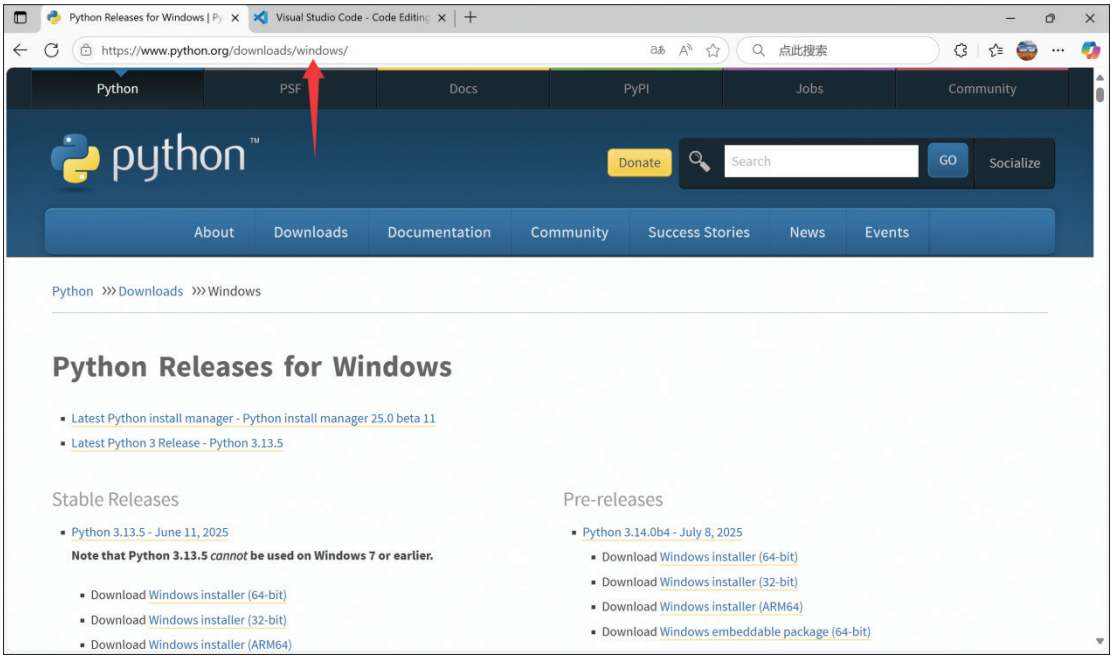

图 1-8 其他已经发布的旧版本的 Python 安装文件

如果想下载适合在苹果系统 MacOS 上使用的各种 Python 版本，可以访问 Python 官网，在下载页面上选择希望使用的 Python 版本安装文件下载安装，如图 1-9 所示。

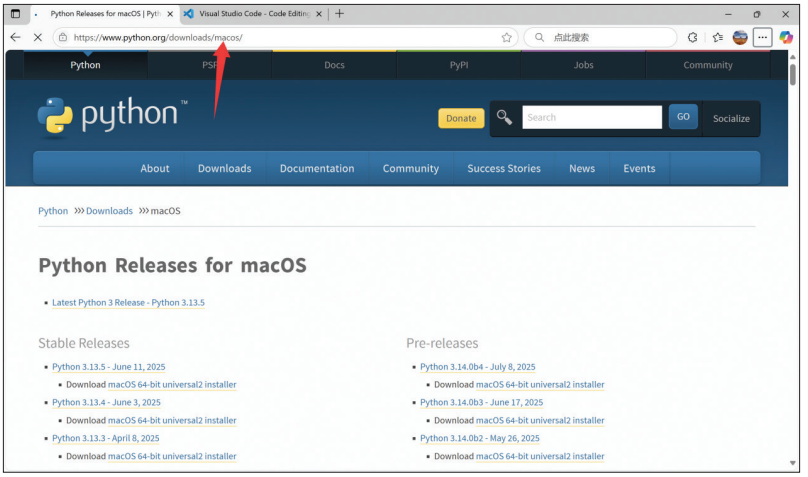

图 1-9　各种版本的可以在 macOS 上安装的 Python

(5) 在本地计算机上安装 Python 编程环境。双击已经下载好的应用程序图标开始安装，如图 1-10 所示。

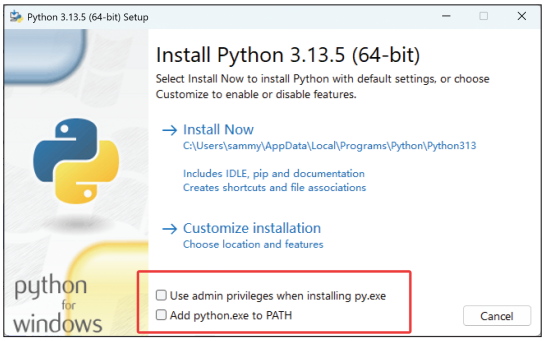

图 1-10　开始安装 Python 编程环境

(6) 安装的第一步非常关键，请务必将下面两个复选框全部勾选，特别是“Add python.exe to PATH”，如图 1-11 所示。勾选后，Windows 系统会自动配置好环境变量。设置好之后，在使用命令行窗口命令的时候，Windows 会自动查看 Python 安装路径文件夹中的文件。

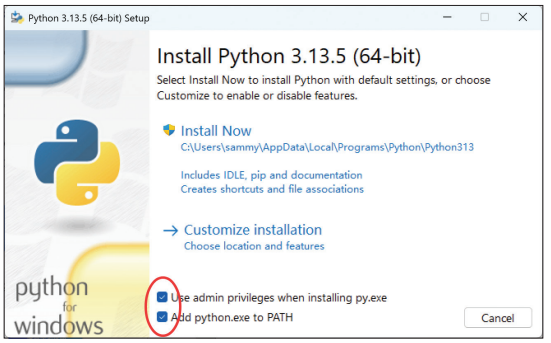

图 1-11　勾选程序的复选框

(7) 选择“Install Now”选项，就可以将 Python 安装至默认文件夹。如果需要安装到自己指定的文件夹中，则需要选择“Customize installation”，此处建议选择默认目录安装，如图 1-12 所示。

图 1-12　选择“Install Now”选项进行安装

(8) 在 Windows 系统中安装 Python 的进程如图 1-13 所示。

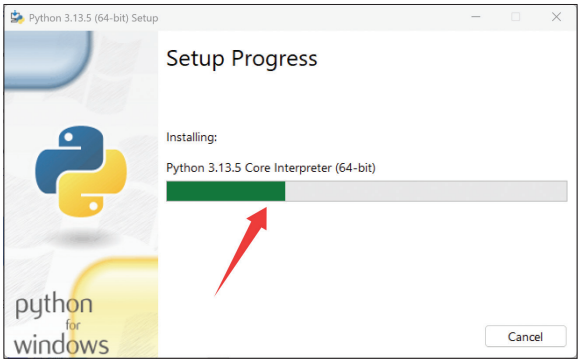

图 1-13　安装进程

(9) 完成安装后，自动弹出安装成功的提示对话框，如图 1-14 所示。如果出现了要求取消路径最大长度限制的选项，可根据需要，单击该选项。

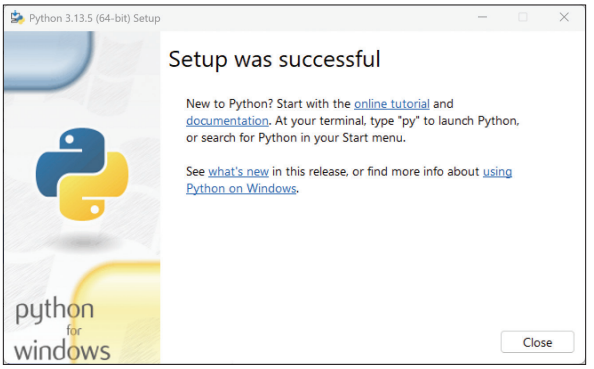

图 1-14　“安装成功”对话框

这时，单击 Windows 的“开始”菜单的“全部”按钮就会出现安装好的 Python 3.13 快

捷方式的文件夹。“全部”按钮如图 1-15 所示。

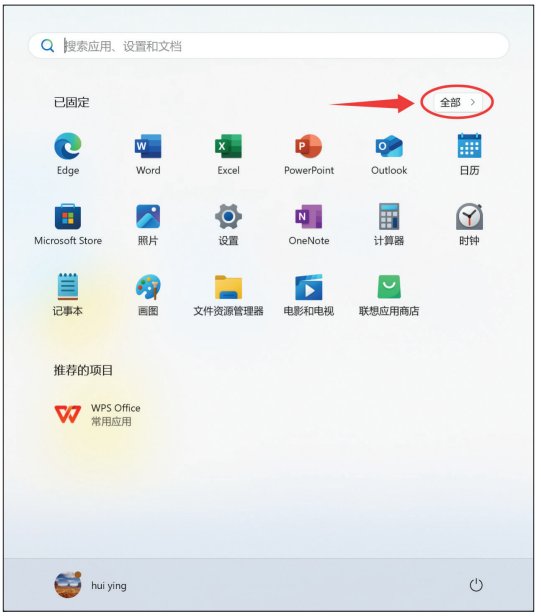

图 1-15　安装完成后单击“开始”菜单的“全部”按钮

Python 3.13 文件夹中存在 4 个文件的快捷方式，如图 1-16 所示。其具体说明如下。

① IDLE：Python 自带的简易开发环境。

② Python 3.13：运行 Python 环境。

③ Python 3.13 Manuals：Python 手册。

④ Python 3.13 Module Docs：模块说明文档。

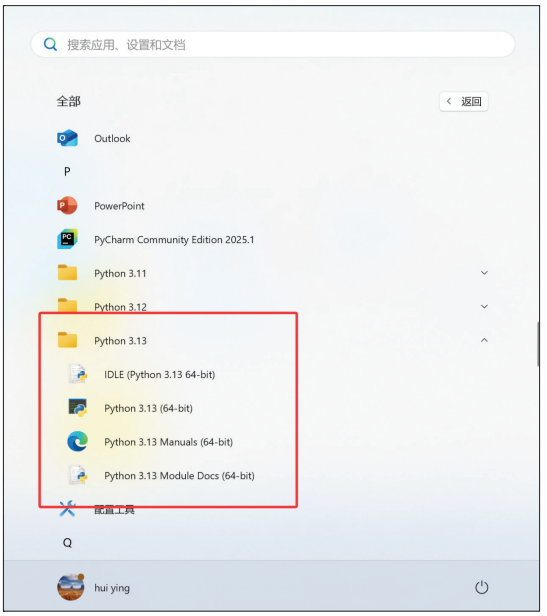

图 1-16　安装完成后，所有应用中出现的 Python 3.13 文件夹

1.5.2　运行 Python

单击“Python 3.13(64-bit)”快捷方式，运行 Python 环境后，会出现图 1-17 所示的命令行窗口。

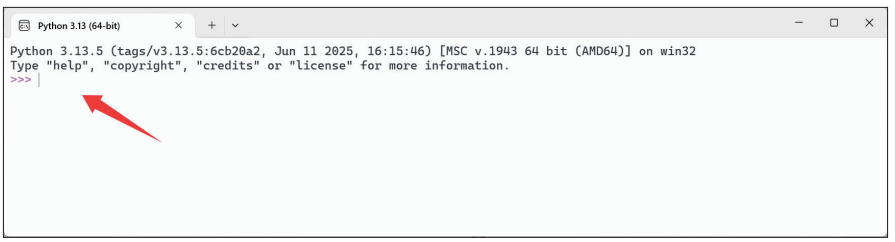

图 1-17　运行 Python 的命令行窗口

图 1-17 显示了安装的 Python 版本号，以及“>>>”，代表执行 Python 3.13 命令成功。这时 Python 编程环境已经可以使用了。

例如，在尖角号顶端输入“1+2”后，按回车键，出现图 1-18 所示的运行结果，说明 Python 已经处于可用状态。

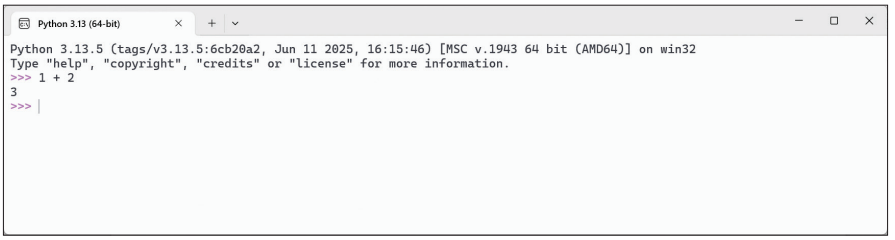

图 1-18　运行 Python 的命令行窗口

若命令行窗口中的命令运行失败，极大可能是在安装的时候未勾选“Add python.exe to PATH”，则需要重新安装一次，或者百度搜索失败原因，在百度经验的指导下，修改 PATH 环境变量。对于初学者来说，卸载后重新安装，并在重新安装的过程中特别注意图 1-12 所提示的问题，解决问题的速度更快。

以上是 Windows 环境下的安装过程。Mac 机上的安装过程大致相同，不再赘述。

1.6　VSCode 集成开发环境的安装

安装完 Python 语言编程环境之后，需要继续安装集成开发环境。如果想要体验 AI 程序员，必须完成本章节的内容。

VSCode 是一个比较好用的 Python 语言编程集成开发环境。初学者可以按照以下步骤完成 VSCode 集成开发环境的安装。

(1) 进入 VSCode 的官方页面，如图 1-19 所示。

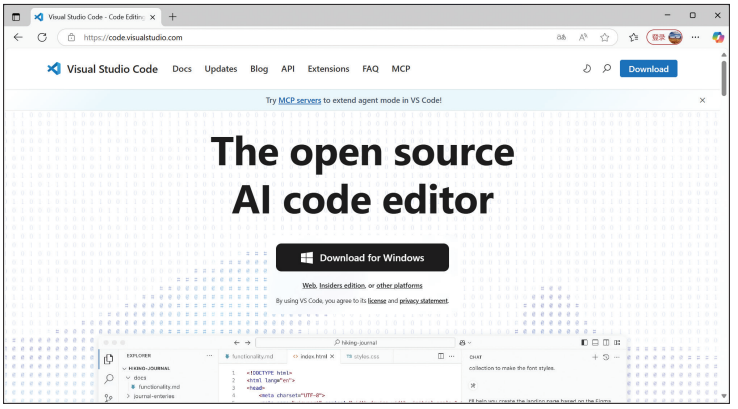

图 1-19　VSCode 官方网页

(2) 单击“Download for Windows”按钮进入下载页面，如图 1-20 所示。

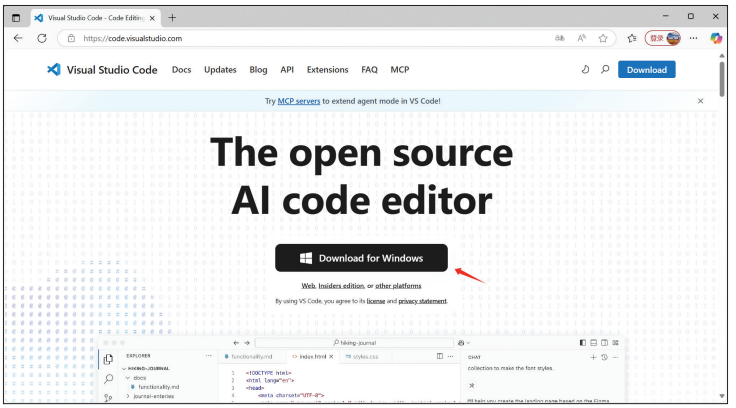

图 1-20　单击“Download for Windows”按钮

(3) 这时页面会跳转，不用做什么操作，等待下载开始即可，如图 1-21 所示。

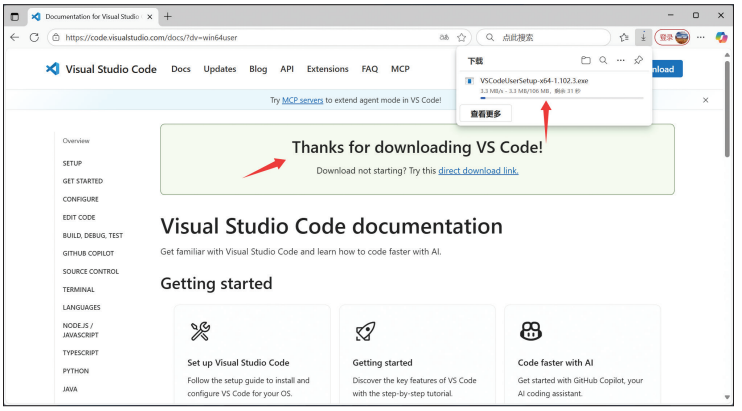

图 1-21　开始下载

(4) 下载完成后，就可以开始安装工作了。选择浏览器的“打开文件”，即可开始安装过程，如图 1-22 所示；或者双击已经下载好的 VSCode 安装文件进行安装。

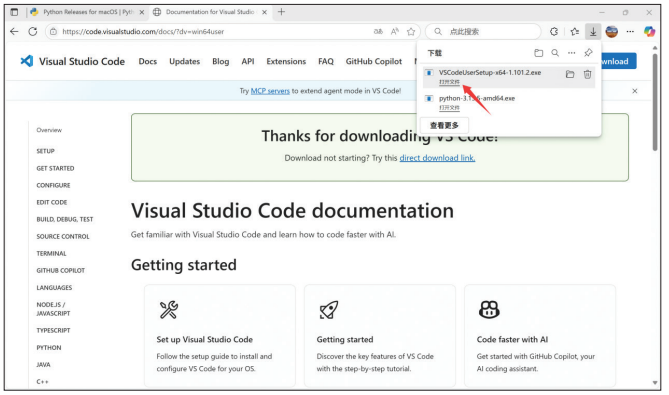

图 1-22　下载完成

(5) 开始安装过程，选择“我同意此协议 (A)”后，单击“下一步 (N)”按钮，如图 1-23 所示，开始安装。

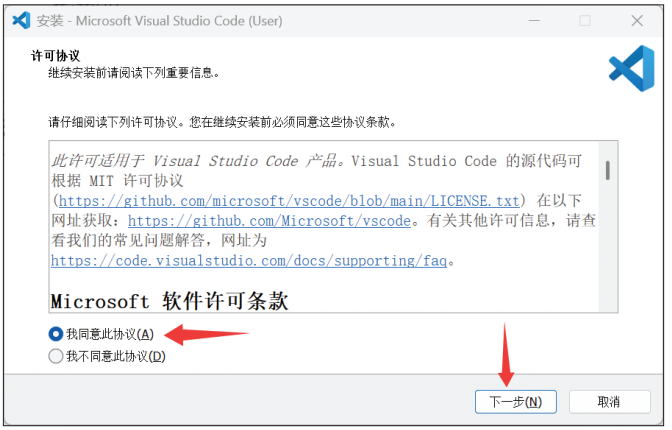

图 1-23　安装 PyCharm 的运行按钮

(6) 在安装向导中单击“下一步 (N)”按钮，在安装过程中可以自定义安装路径。在图 1-24 所示位置更改安装路径，也可以使用默认路径。

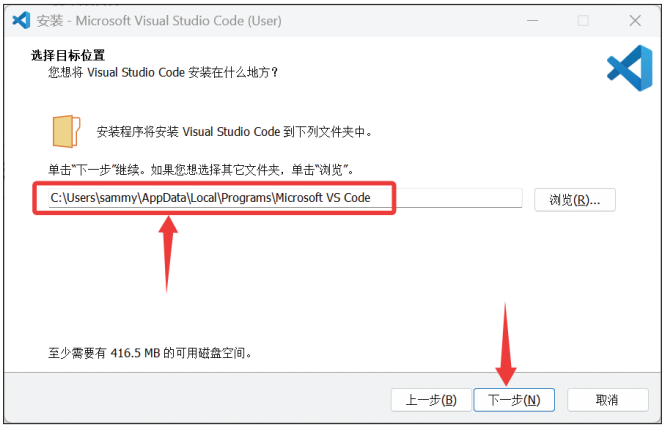

图 1-24　可以更改 VSCode 的安装路径

(7) 在对话框中单击“下一步 (N)”按钮。在选择安装选项时，可以在开始菜单文件夹中放置 VSCode 的快捷方式，如图 1-25 所示。

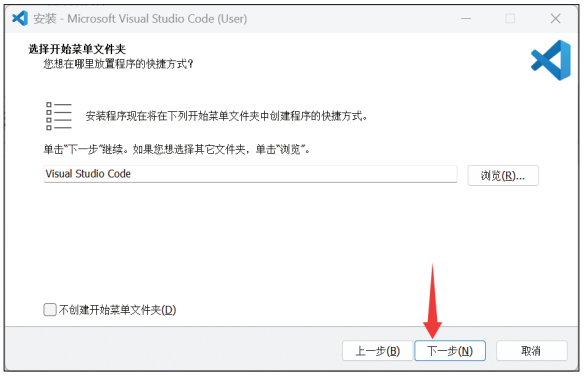

图 1-25　VSCode 的快捷方式安放路径安装选项

(8) 勾选所有的复选框，单击“下一步 (N)”按钮，如图 1-26 所示。

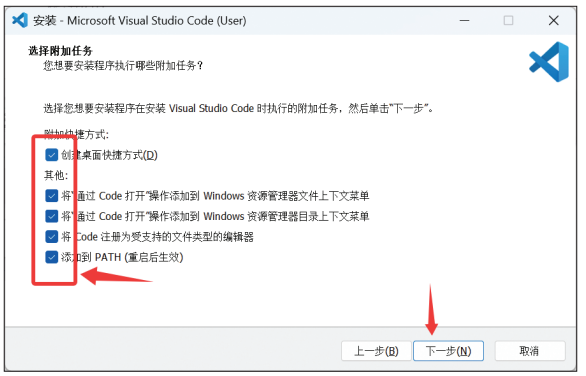

图 1-26　开始安装 PyCharm

(9) 安装完成后，显示图 1-27 所示界面，这时可以单击“安装 (I)”按钮。

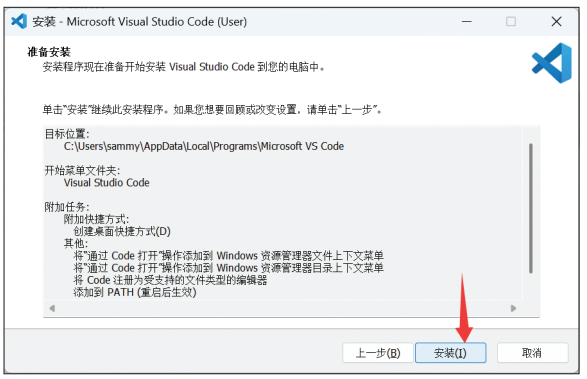

图 1-27　开始 VSCode 的安装

以上是 VSCode 集成开发环境的安装过程。

1.7　LLM 程序设计实验

大语言模型 (LLM) 是基于深度学习技术的自然语言处理工具，通过海量数据训练实现对人类语言的理解、生成和推理。这类模型的核心在于利用多层神经网络捕捉语言规律，广泛应用于文本生成、问答系统、翻译等领域，正在重塑人机交互方式。

我国在 LLM 自主研发中成果显著，其中典型的有文心一言 (百度)、通义千问 (阿里云) 和星火大模型 (科大讯飞) 等。

- 文心一言：融合知识增强与行业知识图谱，支持企业级智能服务。
- 通义千问：集成多任务学习框架，应用于客服、内容创作等场景。
- 星火大模型：强化语音交互与教育领域适配，如智能批改、个性化学习方案生成。

通义灵码是阿里云开发的一款基于通义大模型的人工智能辅助程序设计工具，对于解决简单编程问题具有良好的适应性。只要对通义灵码下达准确的指令，就可以使其生成适应性较好的代码，帮助程序员完成编程任务。

在 AI 程序员的帮助下，读者会发现，就算没有系统地学习过编写程序，也能写出结构良好的代码。随着技术的迭代发展，AI 程序员的功能会越来越强大。笔者曾经在初、高中的学生中做过实验，在搜集了素材之后，初、高中的学生能够以提示词的方式，完成大学三年级下学期的计算机专业课程教材中提及的大部分课程作业和课程项目。大部分学生能够以提示词的方式构建自己的 Blog 网站。一部分学生能够通过提示词方式自主开发、编写物联网硬件编码，参加信息技术大赛。

AI 程序员给希望进行自主程序编写，而又未学过计算机专业的学习者打开了一扇门。

在 VSCode 集成开发环境中使用通义灵码插件，需要按照以下步骤安装该插件。

(1) 在开始菜单栏找到 Visual Studio Code 的快捷方式，单击该快捷方式，运行 VSCode，如图 1-28 所示。

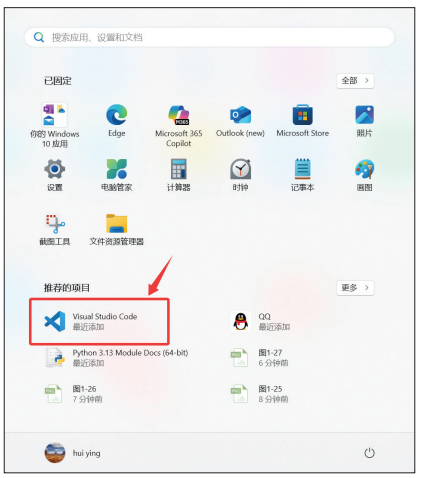

图 1-28　单击 Visual Studio Code 的快捷方式

(2) 启动 VSCode，显示“Welcome”界面，如图 1-29 所示。

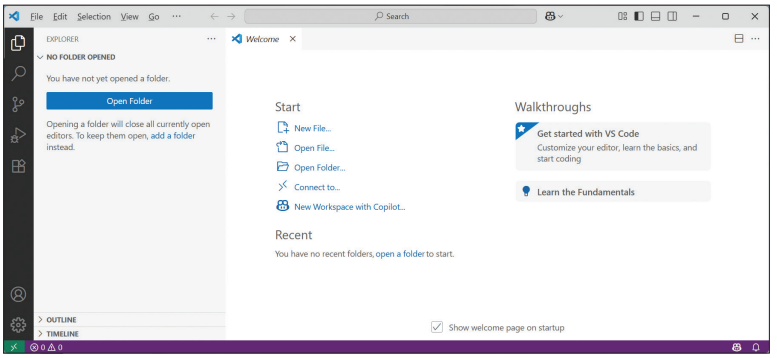

图 1-29　VSCode 的“Welcome”界面

(3) 如果希望将 VSCode 的工作语言改为中文，需要先找到“Command Palette”命令，如图 1-30 所示。

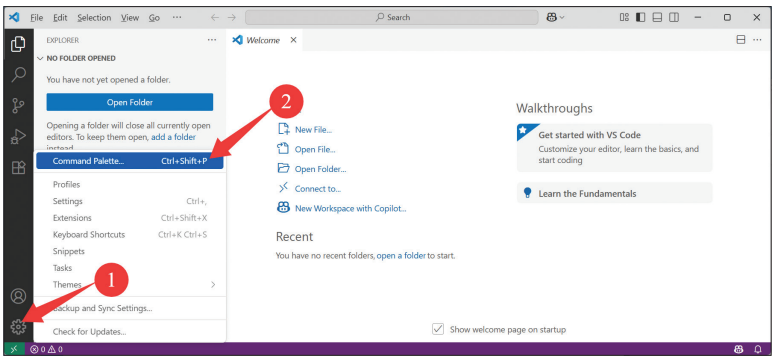

图 1-30　找到“Command Palette”命令

(4) 单击“Extensions”按钮后，在搜索栏输入“Chinese”；然后选择执行“Install”命令，安装中文 (简体) 扩展，如图 1-31 所示。如果选择英文版本，本步骤可以忽略。

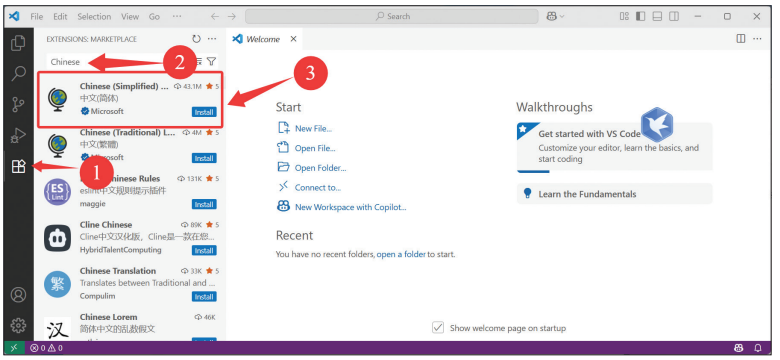

图 1-31　安装中文 (简体) 扩展

(5) 中文 (简体) 扩展安装后，按钮变成“uninstall”，此时需要重启 VSCode，如图 1-32 所示。

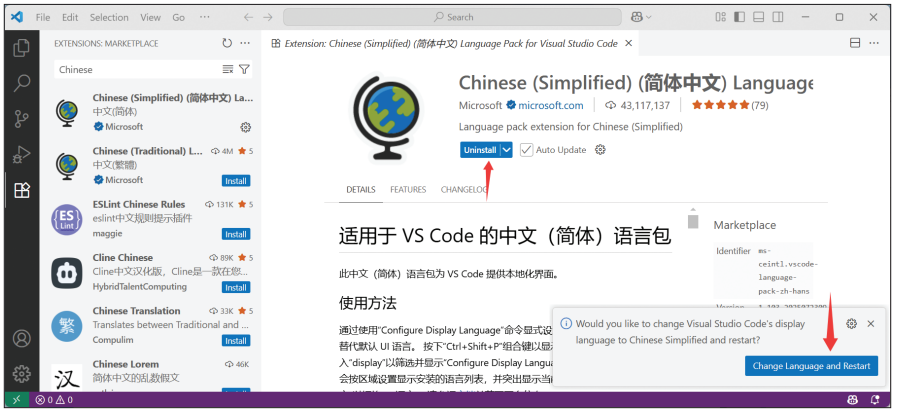

图 1-32　完成中文 (简体) 语言包扩展的安装

(6) 下载完成后，重新启动 VSCode，初始的英文工作界面变为中文 (简体) 的工作界面，如图 1-33 所示。

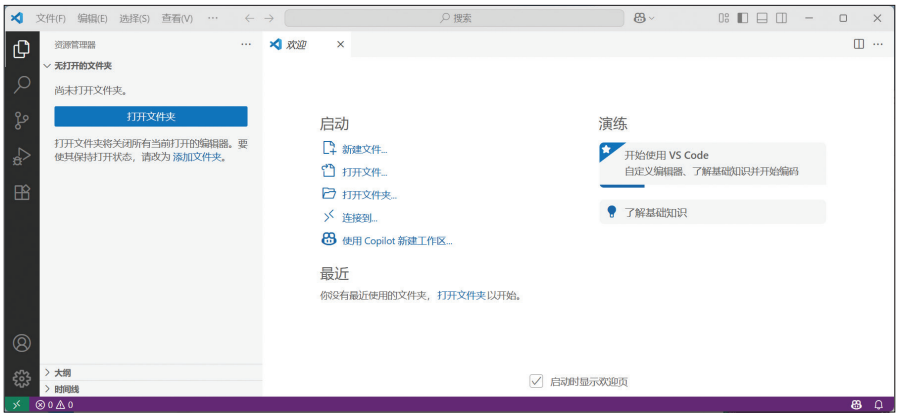

图 1-33　中文 (简体) 工作界面

(7) 在拓展商店中单击“扩展”按钮后，搜索“Lingma”，然后选择执行“安装”命令，安装“通义灵码”扩展，如图 1-34 所示。

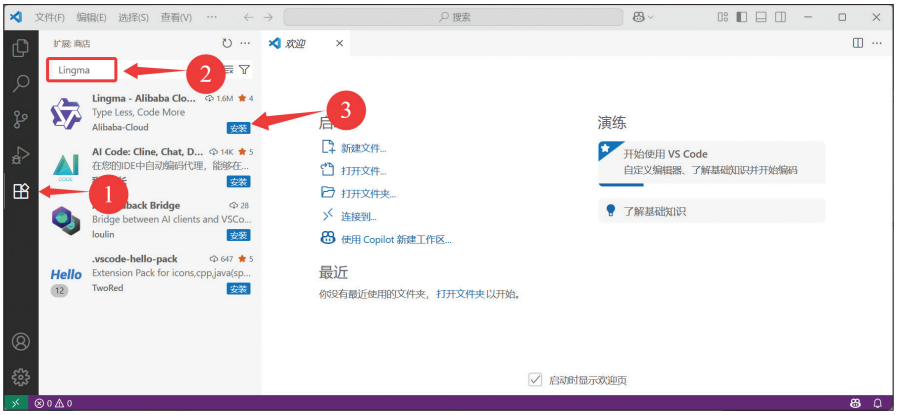

图 1-34　安装“通义灵码”扩展

(8) 单击“信任发布者和安装”按钮，如图 1-35 所示。

图 1-35　单击“信任发布者和安装”按钮

(9) 完成“通义灵码”的安装后，界面如图 1-36 所示，出现通义灵码的“插件激活”按钮。

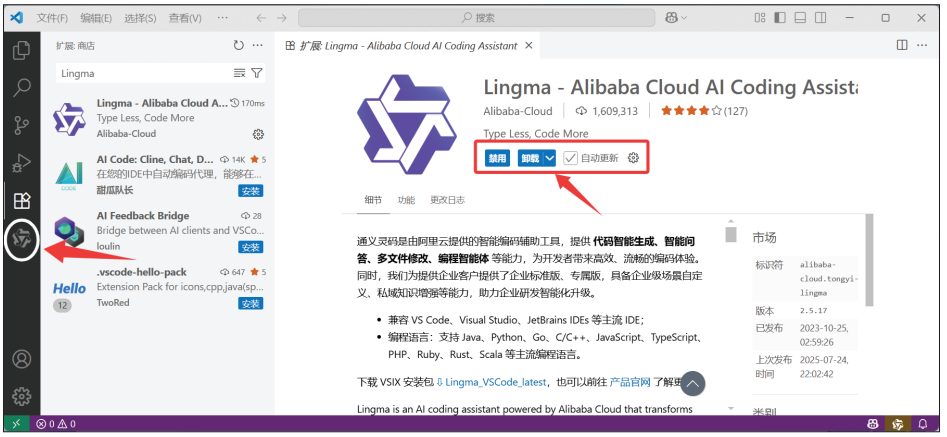

图 1-36　完成了通义灵码插件的安装

(10) 关闭“扩展商店安装”选项卡后，单击“通义灵码”按钮，如图 1-37 所示。

图 1-37　单击“通义灵码”按钮

(11) 带通义灵码扩展的 VSCode 工作界面如图 1-38 所示。

图 1-38　带通义灵码扩展的 VSCode 工作界面

1.8　本章小结

本章介绍了程序设计语言的基础知识。程序设计语言历经了二进制的机器语言、助记符的汇编语言和接近自然语言的高级语言三个阶段。AI 程序员基于深度学习，经大规模代码训练，可以实现从自然语言到代码的转换，具备代码生成、补全、错误检测等功能。目前，比较适用于人工智能程序设计的语言是 Python。在搭配 VSCode 集成开发环境后，Python 语言能够比较好地满足程序设计需求。后续章节将使用阿里云的通义灵码作为 AI 程序员，进行课程学习。