目录 Contents

第 1 章　认识 Python 编程语言

内容要点：

- 认识计算机系统的软硬件组成；
- 理解计算机的基本工作原理；
- 了解 Python 计算机编程语言；
- 安装 Python 语言集成开发环境。

思政目标：

- 尊重科学，了解我国计算机科技的现状和优势，增强民族自豪感和爱国情怀。

1.1　计算机系统的组成

随着科学技术的飞速发展，信息技术已成为当今世界最具潜力的生产力，信息资源已成为国民经济和社会发展的战略资源，信息化水平也已成为一个国家现代化程度的重要标志。计算机作为一种信息处理工具，给我们的学习、工作和生活带来了诸多便利，并将继续推动人类社会向前发展。作为当代大学生，我们掌握计算机的相关基础知识和应用能力是最基本的要求。

计算机俗称电脑，是一种能够按照预先编制的程序自动运行，可高速计算和记忆海量数据的现代化智能电子设备。计算机可分为超级计算机、工业控制计算机、网络计算机、个人计算机、嵌入式计算机等。更先进的计算机还包括生物计算机、光子计算机、量子计算机、神经网络计算机和蛋白质计算机等。

计算机是由硬件系统和软件系统两大部分组成的（见图 1–1）。硬件是计算机的物质组成部分，就

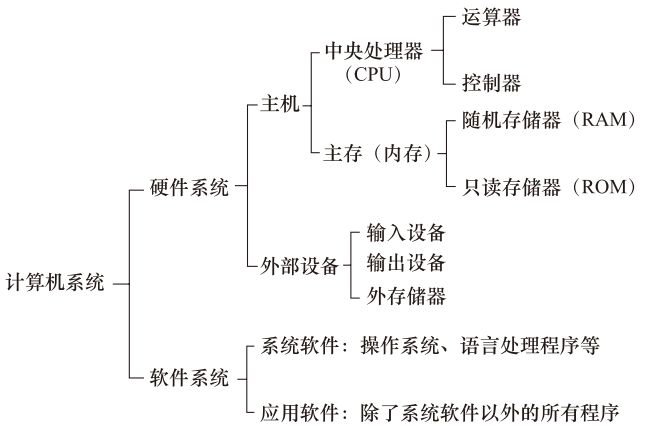

图 1–1　计算机系统组成

像人的躯体；而软件则是数据和指令信息，就像人的灵魂。没有安装任何软件的计算机称为“裸机”。计算机硬件和软件是一个不可分割的完整系统。“裸机”是不能处理任何数据的。下面我们分别来介绍计算机硬件和软件系统。

1.1.1 计算机硬件系统

计算机硬件系统主要由中央处理器（运算器、控制器）、存储器、输入设备、输出设备和各种外部设备组成。中央处理器是对信息进行高速运算处理的主要部件，其处理速度可达每秒数亿次以上。存储器用于存储程序、数据和文件，常由快速的内存储器和慢速海量外存储器组成。各种输入输出外部设备是人机间的信息转换器，由输入 – 输出控制系统管理外部设备与主存储器和中央处理器之间的信息交换。

1. 中央处理器（CPU）

中央处理器即 CPU（Central Processing Unit / Processor），是计算机的运算核心和控制核心。其功能主要是解释计算机指令以及处理计算机软件中的数据。CPU 由运算器、控制器、寄存器、高速缓存及实现它们之间联系的数据、控制及状态的总线构成。作为整个系统的核心，CPU（见图 1–2）也是整个系统中级别最高的执行单元，因此 CPU 已成为决定电脑性能的核心部件，很多用户都以它为标准来判断电脑的档次。

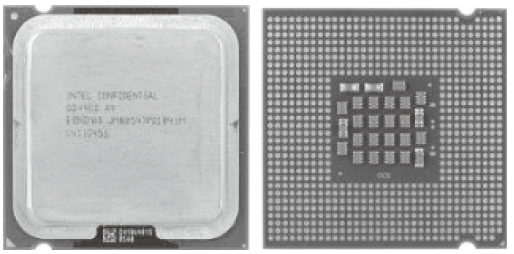

图 1–2　中央处理器（CPU）

（1）运算器：是对数据进行加工处理的部件，它在控制器的作用下与内存交换数据，负责进行各类基本的算术运算、逻辑运算和其他操作。运算器含有暂时存放数据或结果的寄存器。运算器由算术逻辑单元 ALU（Arithmetic Logic Unit）、累加器、状态寄存器和通用寄存器等组成。ALU 是用于完成加、减、乘、除等算术运算，与、或、非等逻辑运算以及移位、求补等操作的部件。

（2）控制器：是整个计算机系统的指挥中心，负责对指令进行分析，并根据指令的要求，有序地、有目的地向各个部件发出控制信号，使计算机的各部件协调一致地工作。控制器由程序计数器、指令译码器、指令寄存器、控制逻辑电路和时钟控制电路等组成。

（3）寄存器：也是 CPU 的一个重要组成部分，是 CPU 内部的临时存储单元。寄存器既可以存放数据和地址，又可以存放控制信息或 CPU 工作的状态信息。

（4）高速缓冲存储器（Cache）：包括一级缓存（L1 Cache）、二级缓存（L2 Cache）、三级缓存（L3 Cache），位于 CPU 与内存之间，是一个读写速度比内存更快的存储器。当 CPU 向内存中写入或读出数据时，这个数据也被存储进高速缓冲存储器中。当 CPU 再次需要这些数据时，CPU 就从高速缓冲存储器读取数据，而不是访问较慢的内存，当然，

如需要的数据在 Cache 中没有，CPU 会再去读取内存中的数据。

2. 主存（内存）

主存（Memory）又叫内部存储器（简称内存），有以下两种：①只读存储器，②随机存储器。它是与 CPU 进行沟通的桥梁。计算机中所有程序的运行都是在内存中进行的，因此内存的性能对计算机的影响非常大。其作用是用于暂时存放 CPU 中的运算数据，以及与硬盘等外部存储器交换的数据。只要计算机在运行中，CPU 就会把需要运算的数据调到内存中进行运算，在运算完成后 CPU 再将结果传送出来，内存的运行也决定了计算机的稳定运行。内存是由内存芯片、电路板、金手指等部分组成的。

（1）只读存储器（Read Only Memory，ROM）。ROM 一般用于存放计算机的基本程序和数据，这些信息在生产厂家在制造 ROM 的时候，就被存入并永久保存在其中，用户对这些信息只能读出，不能写入，即使机器停电，这些数据也不会丢失。如：BIOS（Basic Input Output System）ROM，它是一个固化在计算机内主板上的 ROM 芯片，其通常容量是 1MB 或 2MB 甚至 8MB。该芯片保存着计算机最重要的基本输入输出程序，系统设置信息，开机后自检程序和系统自启动程序。其主要功能是为计算机提供最底层、最直接的硬件设置和控制（见图 1-3）。

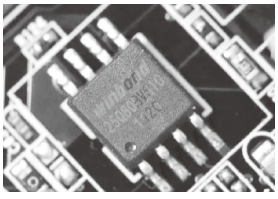

图 1-3　BIOS ROM

（2）随机存储器（Random Access Memory，RAM）。**RAM** 就是我们通常说的内存条，计算机工作时使用的所有程序和数据等都存储在 RAM 中。RAM 属于电子式存储设备，它是由电路板和芯片组成的，特点是体积小，速度快，有电可存，无电清空，即电脑在开机状态时内存中可存储数据，关机后将自动清空其中的所有数据。它允许随机地按任意指定地址向内存单元存入数据或从该单元取出数据，对任意一个地址的存取时间都是相同的。由于信息是通过电信号写入存储器的，所以断电时 RAM 中的信息就会消失。对程序或数据进行了修改之后，应该将它存储到外存储器中，否则关机后信息将丢失。通常所说的内存大小就是指 RAM 的大小。RAM 可分为 DDRAM 内存和 SDRAM 内存（但是 SDRAM 由于容量低，存储速度慢，稳定性差，现在已经被 DDRAM 代替），内存有 DDR，DDR2，DDR3，DDR4 四大类，每一类别又有频率的差异，一般来说，频率越高数据读写速度越快，容量一般为 1 ～ 64GB（见图 1-4）。

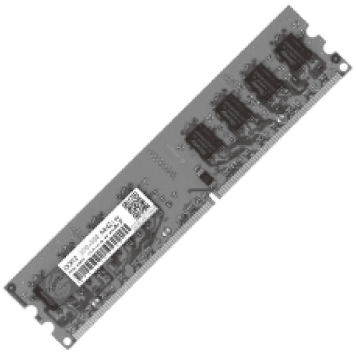

图 1-4　内存条（RAM）

3. 外部存储器

外部存储器简称外存，计算机中常用的外存主要有硬磁盘（简称硬盘）、软磁盘（简称软盘）、光盘、U 盘、移动硬盘等。硬盘是电脑主要的存储媒介之一，主要分为机械硬盘（Hard Disk Drive，HDD）和固态硬盘（Solid

State Drive，SSD），HDD 采用磁性碟片来存储，SSD 采用闪存颗粒来存储。硬盘存储空间比较大，有的硬盘容量已在 2TB 以上。硬盘上保存的数据在不通电状态下是不会丢失的，只要硬盘不坏则可永久保存数据。绝大多数硬盘都是固定硬盘，被永久性地密封固定在硬盘驱动器中。

（1）机械硬盘（HDD）：是由一个或者多个铝制或者玻璃制的碟片组成的，这些碟片外覆盖有铁磁性材料。机械硬盘大多由多个盘片组成，此时，除了每个盘片要分为若干个磁道和扇区以外，多个盘片表面的相应磁道将在空间上形成多个同心圆柱面（见图 1-5）。

（2）固态硬盘（SSD），简称固盘：是用固态电子存储芯片阵列制成的，其特点是数据读写速度快、低功耗、无噪声、抗震动、低热量、体积小。固盘芯片的工作温度范围很宽，商规产品为 0 ～ 70℃，工规产品为 -40 ～ 85℃。因此虽然其成本较高，但也正在逐渐普及到 DIY 市场。由于固态硬盘技术与传统硬盘技术不同，所以产生了不少新兴的存储器厂商。厂商只需购买 NAND 存储器，再配合适当的控制芯片，就可以制造固态硬盘了。新一代的固态硬盘普遍采用 SATA-3 接口、M.2 接口、MSATA 接口、PCI-E 接口、SAS 接口、CFast 接口和 SFF-8639 接口。固态硬盘比机械硬盘更耐用、更低温、更抗震、更便携。因此固态硬盘被广泛应用于军事、车载、工业、医疗、航空等领域（见图 1-6）。

图 1-5　硬盘（HDD）

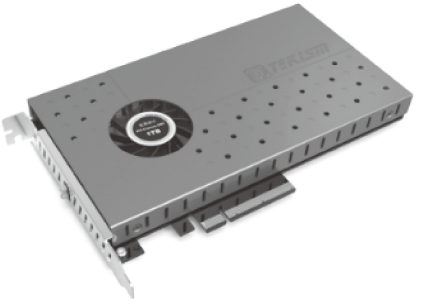

图 1-6　固盘（SSD）

4. 输入设备

输入设备是向计算机输入数据和信息的设备，是计算机与用户或其他设备通信的桥梁，是用户和计算机系统之间进行信息交换的主要装置之一。常用的输入设备有键盘、鼠标、摄像头、扫描仪、光笔、手写输入板、游戏杆、语音输入装置等。

（1）键盘（Keyboard）：是常用的输入设备，由一组开关矩阵组成，包括数字键、字母键、符号键、功能键及控制键等。每一个按键在计算机中都有它的唯一代码。当按下某个键时，键盘接口将该键的二进制代码送入计算机主机中，并将按键字符显示在显示器上。当快速大量输入字符，主机来不及处理时，先将这些字符的代码送往内存的键盘缓冲区，然后再从该缓冲区中取出进行分析处理。键盘接口电路多采用单片微处理器，由它控制整个键盘的工作，如上电时对键盘的自检、键盘扫描、按键代码的产生、发送及与主机

的通信等。常用键盘分为机械键盘、塑料薄膜式键盘、导电橡胶式键盘、无接点静电电容键盘等。

（2）鼠标（Mouse）：是一种手持式屏幕坐标定位设备，它是适应菜单操作的软件和图形处理环境而出现的一种输入设备，特别是在现今流行的 Windows 图形操作系统环境下应用鼠标器方便快捷。常用的鼠标器有两种，一种是机械式鼠标，另一种是光电式鼠标。

5. 输出设备

输出设备是人与计算机交互的一种部件，用于数据的输出。它把各种计算结果数据或信息以数字、字符、图像、声音等形式表示出来。常见的有显示器、打印机、绘图仪、影像输出系统、语音输出系统、磁记录设备等。

显示器（Display）是计算机必备的输出设备，常见显示器分为 CRT 显示器、LCD 显示器、LED 显示器。

（1）CRT 显示器：是一种使用阴极射线管（Cathode Ray Tube）的显示器，通常是一台电脑的标准设备之一。早期的 CRT 显示器只有绿色的一小块，而如今 20 寸的 CRT 显示器都司空见惯了。随着尺寸的增加，CRT 显示器的显示效果也在提高。

（2）LCD 显示器，也称为液晶显示器。液晶是一种介于固体和液体之间的特殊物质，它是一种有机化合物，常态下呈液态，但是它的分子排列却和固体晶体一样非常规则，因此取名液晶。它的另一个特殊性质在于，如果给液晶施加一个电场，会改变它的分子排列。这时如果给它配合偏振光片，它就具有阻止光线通过的作用（在不施加电场时，光线可以顺利透过），如果再配合彩色滤光片，改变加给液晶电压大小，就能改变某一颜色透光量的多少，也可以形象地说改变液晶两端的电压就能改变它的透光度（但实际中这必须和偏光板配合）。

（3）LED 显示器：通过控制半导体发光二极管显示，用来显示文字、图形、图像、动画、行情、视频、录像信号等各种信息的显示屏幕。通过发光二极管芯片的适当连接（包括串联和并联）和适当的光学结构，可构成发光显示器的发光段或发光点。由这些发光段或发光点可以组成数码管、符号管、米字管、矩阵管、电平显示器管等。通常把数码管、符号管、米字管统称为笔画显示器，而把笔画显示器和矩阵管统称为字符显示器。

6. 主板

主板（Motherboard, Mainboard，简称 Mobo）：又称主机板、系统板、逻辑板、母板、底板等。它安装在机箱内，是个人计算机最基本最重要的部件之一，上面安装了组成计算机的主要电路系统，一般有 BIOS 芯片、I/O 控制芯片、键盘和面板控制开关接口、指示灯插接件、扩充插槽、主板及插卡的直流电源供电接插件等元件。主板上大都有 6 ～ 15 个扩展插槽，供 PC 机处理器、显卡、声效卡、硬盘、存储器等设备的控制卡（适配器）

插接。用户通过更换这些插卡，可以对微机的相应子系统进行局部升级，使厂家和用户在配置机型方面有更大的灵活性。总之，主板在整个微机系统中扮演着举足轻重的角色。可以说，主板的类型和档次决定着整个微机系统的类型和档次，主板的性能影响着整个微机系统的性能（见图 1-7）。

7. 电源设备

电源是把 220V 交流电，转换成直流电，并专门为电脑配件，如 CPU、主板、硬盘、内存条、显卡、光盘驱动器等供电的设备，是电脑各部件供电的枢纽，是电脑的重要组成部分。计算机电源主要由电磁滤波器、保护器、变压器、散热器、滤波电路、保护电路等部件组成（见图 1-8）。

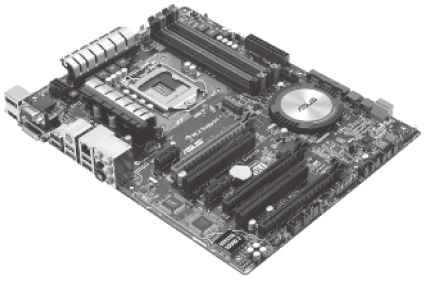

图 1-7　主板

图 1-8　电源

1.1.2　计算机软件系统

软件（Software）是一系列按照特定顺序组织的电脑数据和指令的集合。一般来说，软件被划分为系统软件和应用软件。其中系统软件为计算机使用提供最基本的功能，但是并不针对某一特定应用领域。而应用软件则恰好相反，不同的应用软件根据用户和所服务的领域提供不同的功能。软件并不只是包括可以在计算机上运行的电脑程序，还包括与这些电脑程序相关的文档。简单来说，软件就是程序加文档的集合体。软件被应用于社会的各个领域，对人们的生活和工作都产生了深远的影响。

（1）系统软件：是负责管理计算机系统中各种独立的硬件，使得它们可以协调工作的软件。系统软件使得计算机使用者和其他软件将计算机当作一个整体而不需要顾及底层每个硬件是如何工作的。一般来讲，系统软件包括操作系统和一系列基本的工具（比如编译器，数据库管理，存储器格式化，文件系统管理，用户身份验证，驱动管理，网络连接，等等）。常用的操作系统有 Windows、Linux、UNIX 等。

（2）应用软件：是为了某种特定的用途而被开发的软件。它可以是一个特定的程序，比如一个图像浏览器；可以是一组功能联系紧密，互相协作的程序的集合，比如微软的 Office 软件；也可以是一个由众多独立程序组成的庞大的软件系统，比如信息管理系统。

1.2 计算机工作原理

计算机根据人们预定的安排，自动地进行数据的快速计算和加工处理。人们预定的安排是通过一连串指令（操作者的命令）来表达的，这个指令序列就是程序。一个指令规定计算机执行一个基本操作。一个程序规定计算机完成一个完整的任务。一种计算机所能识别的一组不同指令的集合，称为该种计算机的指令集合或指令系统。在微机的指令系统中，主要使用了单地址和二地址指令，其中，第 1 个字节是操作码，规定计算机要执行的基本操作，第 2 个字节是操作数。计算机指令包括以下类型：数据处理指令（加、减、乘、除等）、数据传送指令、程序控制指令、状态管理指令。整个内存被分成若干个存储单元，每个存储单元一般可存放 8 位二进制数（字节编址）。每个单元可以存放数据或程序代码，为了能有效地存取该单元内存储的内容，每个单元都给出了一个唯一的编号来标识，即地址。

计算机在运行时，先从内存中取出第一条指令，通过控制器的译码，按指令的要求，从存储器中取出数据进行指定的运算和逻辑操作等加工，然后再按地址把结果送到内存中去，接下来，再取出第二条指令，在控制器的指挥下完成规定操作，依此进行下去，直至遇到停止指令。

程序与数据一样存储，按程序编排的顺序，一步一步地取出指令，自动地完成指令规定的操作是计算机最基本的工作原理。这一原理最初是由美籍匈牙利数学家冯 • 诺依曼于 1945 年提出来的，故称为冯 • 诺依曼原理。冯 • 诺依曼原理的核心是“存储程序控制”，其内容包括以下几项：

（1）采用二进制（0，1）形式表示数据和指令。

（2）将程序（数据和指令序列）预先存放在主存储器中，使计算机在工作时能够自动高速地从存储器中取出指令，并加以执行。

（3）由运算器、存储器、控制器、输入设备和输出设备五大基本部件组成计算机系统，并规定了这五大部件的基本功能。冯 • 诺依曼思想实际上是电子计算机设计的基本思想，奠定了现代电子计算机的基本结构（见图 1–9）。

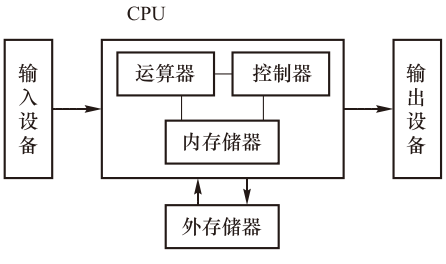

图 1–9　冯 • 诺依曼原理

1.3 计算机编程语言

上面我们已经了解到计算机是通过执行程序指令来完成工作的，这些程序的指令必须要遵循该计算机能识别的指令规则来编写，不同的计算机其指令系统是不同的，这主要取决于所用到的 CPU。这就是 Windows 操作系统不能安装在苹果电脑上运行的原因。

计算机程序指令可以用各种符号来表示，可以是 0 或 1、英语单词甚至是汉字。但这些符号必须按规定的语法规则来编写，不同的符号加上不同的语法规则就代表不同的编程语言。因此，计算机编程语言就是编写程序的工具。打个比方，就如我们可以用汉语来写文章，还可以用英语等其他语言来写文章，但不同的语言会有不同的语法规则。从计算机诞生至今，计算机编程语言经历了“机器语言”“汇编语言”和“高级语言”几个阶段。

（1）机器语言。根据冯 • 诺依曼原理，我们知道计算机只能够识别 0 和 1 两种符号。所有程序指令最终都要转换为 0 和 1 来表示，计算机才能运行。用 0 和 1 表示的编程语言我们则称之为“机器语言”。

用机器语言编写程序，编程人员要首先熟记所用计算机的全部指令代码和代码的涵义。编写程序时，程序员得自己处理每条指令和每一数据的存储分配和输入输出，还得记住编程过程中每步所使用的工作单元处在何种状态。这是一件十分繁琐的工作，编写程序花费的时间往往是实际运行时间的数十倍或数百倍。而且，编出的程序全是些 0 和 1 的指令代码，直观性差，还容易出错。除了计算机生产厂家的专业人员外，绝大多数程序员已经不再学习机器语言。

（2）汇编语言。为了克服机器语言难读、难编、难记和易出错的缺点，人们就用与代码指令实际含义相近的英文缩写词、字母和数字等符号来取代指令代码（如用 ADD 表示运算符号“+”的机器代码），于是就产生了汇编语言。所以说，汇编语言是一种用助记符表示的仍然面向机器的计算机编程语言，汇编语言亦称符号语言。

汇编语言由于采用了助记符号来编写程序，比用机器语言的二进制代码编程要方便些，在一定程度上简化了编程过程。汇编语言的特点是用符号代替了机器指令代码，而且助记符与指令代码一一对应，基本保留了机器语言的灵活性。使用汇编语言能面向机器并较好地发挥机器的特性，得到质量较高的程序。

汇编语言中由于使用了助记符号，用汇编语言编制的程序送入计算机，计算机不能像用机器语言编写的程序一样直接识别和执行，必须通过预先放入计算机的“汇编程序”的加工和翻译，才能变成能够被计算机识别和处理的二进制代码程序。用汇编语言等非机器

语言书写好的符号程序称“源程序”，运行时汇编程序要将源程序翻译成“目标程序”。目标程序是机器语言程序，它一经被安置在内存的预定位置上，就能被计算机的 CPU 处理和执行。

汇编语言像机器指令一样，是硬件操作的控制信息，因而仍然是面向机器的语言，使用起来还是比较烦琐费时，通用性也差。汇编语言是“低级语言”。但是，汇编语言用来编制系统软件和过程控制软件，其目标程序占用内存空间少，运行速度快，有着高级语言不可替代的用途。

（3）高级语言。不论是机器语言还是汇编语言都是面向硬件具体操作的，语言对机器的过分依赖，要求使用者必须对硬件结构及其工作原理都十分熟悉，这对非计算机专业人员是难以做到的，对于计算机的推广应用是不利的。计算机行业的发展，促使人们去寻求一些与人类自然语言相接近且能为计算机所接受的语意确定、规则明确、自然直观和通用易学的计算机语言。这种与自然语言（如英语）相近并为计算机所接受和执行的计算机语言称为“高级语言”。高级语言是面向用户的语言。无论何种机型的计算机，只要配备上相应的高级语言的编译或解释程序，则用该高级语言编写的程序就可以通用。如今被广泛使用的高级语言很多，如 C，C++，C#，JAVA，Python 等。每一种高级（程序设计）语言，都有自己人为规定的专用符号、英文单词、语法规则和语句结构（书写格式）。高级语言与自然语言（英语）更接近，而与硬件功能相分离（彻底脱离了具体的指令系统），便于广大用户掌握和使用。高级语言的通用性强，兼容性好，便于移植。

计算机不能直接地接收和执行用高级语言编写的源程序，源程序在输入计算机时，通过“翻译程序”翻译成机器语言形式的目标程序，计算机才能识别和执行。这种“翻译”通常有两种方式，即“编译方式”和“解释方式”（见图 1-10）。

编译方式：是事先编好一个称为“编译程序”的机器语言程序，作为系统软件存放在计算机内，在用户将由高级语言编写的源程序输入计算机后，“编译程序”便把源程序整个地翻译成用机器语言表示的与之等价的目标程序，然后计算机再执行该目标程序，以完成源程序要处理的运算并取得结果。

采用编译方式执行的编程语言称之为“编译型语言”，例如 C，C++ 等就属于编译型语言。使用编译型语言开发完成后需要将所

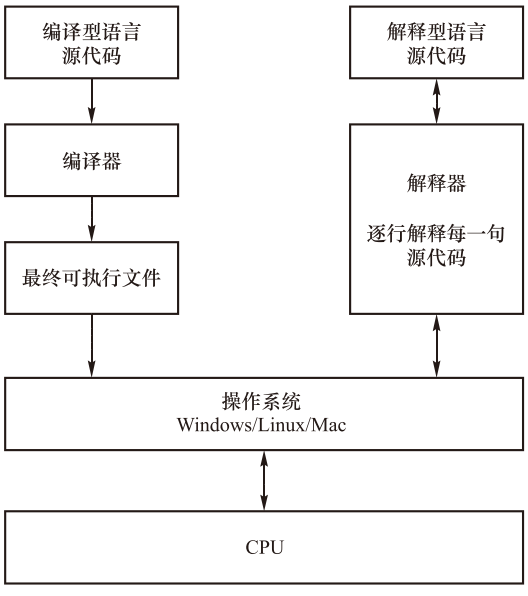

图 1-10　编译型语言和解释型语言的执行流程

有的源代码都转换成“可执行程序”，比如 Windows 下的 .exe 文件，可执行程序就是计算机能直接识别的二进制机器码。只要我们拥有可执行程序，就可以随时运行，不用再重新编译了，也就是“一次编译，无限次运行”。在运行的时候，我们只需要直接运行生成的可执行程序，不再需要源代码和编译器了，所以说编译型语言可以脱离开发环境运行。

但编译型语言最大的问题是其开发的程序一般是不能跨平台的。具体表现在两方面：

1）可执行程序不能跨平台。可执行程序不能跨平台很容易理解，因为不同操作系统对可执行文件的内部结构有着截然不同的要求，彼此之间也不能兼容。比如，不能将 Windows 操作系统下的可执行程序拿到 Linux 操作系统下运行，也不能将 Linux 下的可执行程序拿到 Mac OS 下运行。

另外，相同操作系统的不同版本之间也不一定兼容，比如，不能将 X64 程序（Windows 64 位程序）拿到 X86 平台（Windows 32 位平台）下运行。但是反之一般可行，因为 64 位 Windows 对 32 位程序作了很好的兼容性处理。

2）源代码不能跨平台。不同平台支持的函数、类型、变量等都可能不同，基于某个平台编写的源代码一般不能拿到另一个平台下编译。例如，在 C 语言中要想让程序暂停可以使用“睡眠”函数，在 Windows 平台下该函数是 Sleep( )，在 Linux 平台下该函数是 sleep( )，首字母大小写不同。其次，Sleep( ) 的参数是毫秒，sleep( ) 的参数是秒，单位也不一样。

解释方式：是源程序输入计算机时，“解释程序”边扫描边解释，逐句输入逐句翻译成计算机能直接识别的二进制机器码，然后计算机一句一句地执行，并不产生目标程序。采用解释方式执行的编程语言称之为“解释型语言”，例如 Python，JavaScript 等就属于解释型语言。

对于解释型语言，每次执行程序都需要一边转换一边执行，用到哪些源代码就将哪些源代码转换成机器码，用不到的不进行任何处理。每次执行程序时可能使用不同的功能，这个时候需要转换的源代码也不一样。

因为每次解释程序都需要重新转换源代码，所以解释型语言的执行效率天生就低于编译型语言。因此，计算机的一些底层功能，或者关键算法，一般使用 C 或 C++ 实现。而解释型语言主要用于开发一些应用层面的软件（比如网站、批处理、小工具等）。

在运行解释型语言的时候，我们始终都需要“源代码”和“解释程序”（解释器），所以说它无法脱离开发环境。当我们“下载一个程序（软件）”时，不同类型的语言有不同的含义：

对于编译型语言，我们下载到的是可执行文件，源代码被作者保留，所以编译型语言的程序一般是“闭源”的。

对于解释型语言，我们下载到的是所有的源代码，因为作者不给源代码就没法运行，

所以解释型语言的程序一般是“开源”的。

相比于编译型语言来说，解释型语言开发的软件最大的优势就在于能跨平台运行“一次编写，到处运行”。那么，为什么解释型语言就能跨平台呢？

这就要归功于解释器，我们所说的跨平台，是指源代码跨平台，而不是解释器跨平台，解释器用来将源代码转换成机器码，它就是一个可执行程序，是不能跨平台的。官方需要针对不同的平台开发不同的解释器，这些解释器必须要能够遵守同样的语法，识别同样的函数，完成同样的功能，只有这样，同样的代码在不同平台的执行结果才是相同的。

1.4 Python 语言

Python 是目前公认的全球五大流行语言之一。Python 已有 30 多年的历史，最近几年迅速火爆，除了因为它简洁、容易上手之外，还因为它在人工智能、数据分析和爬虫等多个领域提供了非常优秀的开发库（模块）。从云计算、大数据到人工智能，Python 无处不在，百度、阿里巴巴、腾讯等一系列大公司都在使用 Python 完成各种任务。

1. Python 语言的诞生与发展

Python 是一门高级计算机编程语言，其作者是 Guido von Rossum（吉多 • 范 • 罗苏姆，中国 Python 程序员都叫他“龟叔”），荷兰人。1982 年，“龟叔”从阿姆斯特丹大学获得了数学和计算机硕士学位。然而，尽管他算得上是一位数学家，但他更加享受计算机带来的乐趣。用他的话说，虽然拥有数学和计算机双料资质，他总趋向于做计算机相关的工作，并热衷于做任何和编程相关的事情。

1989 年，为了打发圣诞节假期，“龟叔”开始写 Python 语言的编译器。Python 这个名字，来自“龟叔”所挚爱的电视剧 *Monty Python's Flying Circus*。他希望这个新的叫作 Python 的语言，能符合他的理想：创造一种 C 和 Shell 之间，功能全面，易学易用，可拓展的语言。“龟叔”作为一个语言设计爱好者，已经有过设计语言的尝试。这一次，也不过是一次纯粹的 hacking 行为。

1991 年，第一个 Python 编译器诞生。它是用 C 语言实现的，并能够调用 C 语言的库文件。Python 从一开始就特别在意可拓展性。Python 可以在多个层次上拓展。从高层上，可以直接引入 .py 文件。在底层上，可以引用 C 语言的库。最初的 Python 完全由“龟叔”本人开发，后来 Python 慢慢得到“龟叔”同事的欢迎。他们迅速地反馈使用意见，并参与到 Python 的改进中来。“龟叔”和一些同事构成 Python 的核心团队。他们将自己大部分的业余时间用

于 hack Python。随后，Python 拓展到研究所之外。Python 将许多机器层面上的细节隐藏，交给解释器处理，并凸显出逻辑层面的编程思考。Python 程序员可以花更多的时间用于思考程序的逻辑，而不是具体的实现细节。这一特征吸引了广大的程序员，Python 开始流行。

1994 年 1 月 Python 1.0 发布了，这个版本的主要新功能是 lambda, map, filter 和 reduce，但是“龟叔”不喜欢这个版本。

2000 年 10 月 Python 2.0 发布了，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。然而，尤为重要的变化是开发流程的改变，Python 此时有了一个更透明的社区。然而，Python 2.x 仍然存在很多缺陷和错误。

2008 年 12 月 Python 3.0 发布了，为了解决 Python 2.x 存在的问题，必须对 Python 3.x 进行全新的设计，所以 Python 3.x 是不兼容 Python 2.x 的，这意味着 Python 3.x 无法运行 Python 2.x 的代码。Python 的社区也一直在蓬勃发展，当我们提出一个有关的 Python 问题，几乎总是有人遇到了同样的问题并已经解决了。因此，学习 Python 并不是很难，只需要安装好环境——开始敲代码——遇到问题——解决问题，就是这么简单。

2. Python 语言的特点

（1）解释性脚本语言：不需要编译就可以直接运行，由于 Python 是一种解释执行的计算机语言，因此它的应用程序运行起来要会比编译式的计算机语言慢一些，这也是 Python 的缺点。但现在的计算机硬件性能越来越强大，这个缺点已不再是问题。

（2）面向对象的语言：在 Python 中一切都是对象。

（3）动态语言：变量的类型可以在运行时发生变化。

（4）强类型：某个变量在某个特定时刻类型确定，不能将字符串对象当作整数来使用，与之相对的是弱类型语言，如 PHP。

（5）语法简单：这降低了入门门槛，使得 Python 非常容易上手。

（6）易于扩展：可以方便地将其他语言开发的模块加入到 Python 中。

（7）开源免费：Python 解释器都可以免费获得和使用。Python 语言也是免费的，任何人都可以开发自己的 Python 解释器，不用给任何人交专利费用。

（8）可移植性强：Python 解释器在目前主流硬件架构和操作系统上都获得了支持，而且绝大多数的 Python 代码可以在这些平台上无差别地运行。

（9）丰富的库：这个决定了 Python 语言的应用领域。目前 Python 在互联网、人工智能、手机应用开发等领域都有各种丰富的库可以使用。Python 语言现在可以算是一种通用开发语言了，在各个领域中都得到了应用。

1.5 安装 Python 语言集成开发环境

集成开发环境（Integrated Development Environment，IDE）是用于提供程序开发环境的应用程序，一般包括代码编辑器、编译器、调试器和图形用户界面工具，具备代码编写功能、分析功能、编译功能（解释功能）、调试功能等一体化的开发软件服务包。Python IDE 就是给 Python 语言的编写、调试、解释提供环境的一种程序。

1. IDLE（Integrated Development Environment）

因为 Python 语言是开放的，任何人、任何公司或者组织都可以自己做解释器，所以网上可以找到很多版本的 Python IDE。但最正宗的是 python.org 提供的用 C 语言实现的 Python IDE，它可以在 Python 官网上进行免费下载（见图 1-11）。

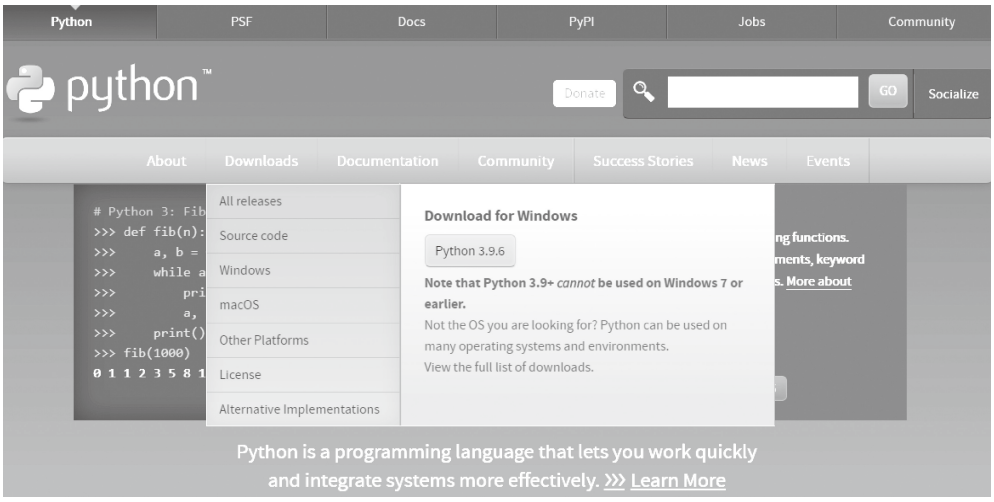

图 1-11　Python 官网

下载的安装包中不仅有 Python 解释器，还包含了一个叫 IDLE 的集成开发环境。下载安装过程如下：

（1）进入 Python 官网（https://www.python.org/），找到适合自己电脑操作系统的 Python 版本，然后点击下载。

（2）双击所下载的程序，进行 Python 的安装，在安装过程中建议选择自定义安装到自己指定的目录位置。

（3）在安装过程中，建议在复选框中勾选上 Install launcher for all users 和 Add Python 3.9 to PATH，这样安装完成后该计算机的所有用户都可以使用，并且安装程序会自动配置

好环境变量（见图 1-12）。

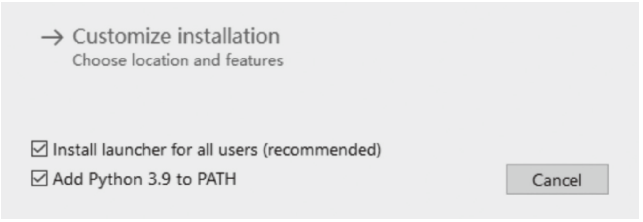

图 1-12　Python 安装过程

（4）安装成功后，打开命令提示符窗口（win+R，再输入 cmd 回车），敲入 python-V 后，出现 Python 版本就说明安装成功。

2. PyCharm

PyCharm 是一款第三方公司开发的常用 Python IDE，它由一家捷克的软件开发公司 JetBrains 所开发。PyCharm 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，比如调试、语法高亮、项目管理、代码跳转、智能提示、自动完成、单元测试和版本控制等。此外，该 IDE 提供了一些高级功能，以用于支持 Django 框架下的专业 Web 开发，使 PyCharm 已成为 Python 专业开发人员和刚起步人员使用的有力工具。下载安装过程如下（注意：安装 PyCharm 前，要先安装 Python 解释器）：

（1）进入 PyCharm 官网（https://www.jetbrains.com/pycharm/）下载 PyCharm 社区版（免费版），PyCharm 专业版是收费的，建议初学者使用社区版（见图 1-13）。

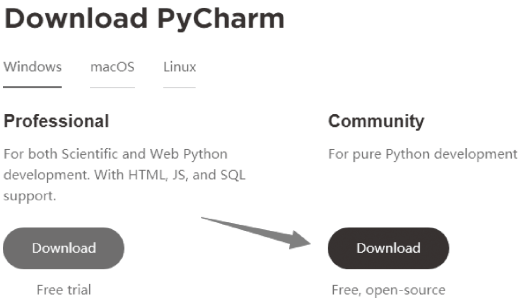

图 1-13　下载 PyCharm 社区版

（2）下载好以后，双击安装，可修改安装路径，然后点击 Next（见图 1-14）。

（3）继续点击 Next 进入图 1-15 界面，建议把复选框都选上。

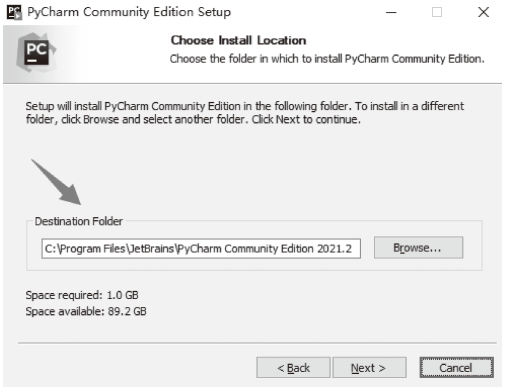

图 1-14　PyCharm 安装向导

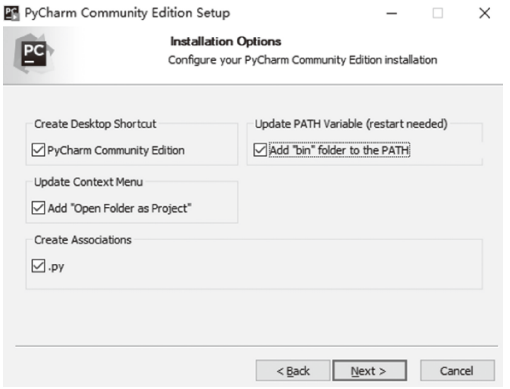

图 1-15　PyCharm 安装向导

（4）再次点击 Next 后，点击 Install 按钮，等待自动安装完成。

本节中，我们介绍了 IDLE 和 PyCharm 两种集成开发环境的下载与安装。IDLE 是 Python 自带的一种轻量级的 IDE，适合 Python 初学者使用。而 PyCharm 的功能更加强大，适合工程人员进行软件项目开发。

1.6　编写第一个 Python 程序

1. 使用 IDLE 编写第一个 Python 程序

（1）命令行编程方式。在安装 Python 后，会自动安装一个 IDLE，它是一个 Python Shell（可以在打开的 IDLE 窗口的标题栏上看到），程序开发人员可以利用 Python Shell 输入程序代码并执行。例如：我们在命令提示符后输入 print（" 你好 Python!"），然后敲回车就会输出“你好 Python!”（见图 1-16）。

图 1-16　IDLE 命令行窗口

（2）文件编程方式。IDLE 命令行编程方式多用于实时交互，不便于程序源代码的长期保存。因此 IDLE 还可以单独创建一个文件来编写程序代码，在全部代码编写完成后一起执行并可长期保存源代码。

1）在 IDLE 主窗口的菜单栏上，选择“File >> New File”菜单项，将打开一个新窗口，在该窗口中直接输入 Python 程序代码（见图 1-17）。

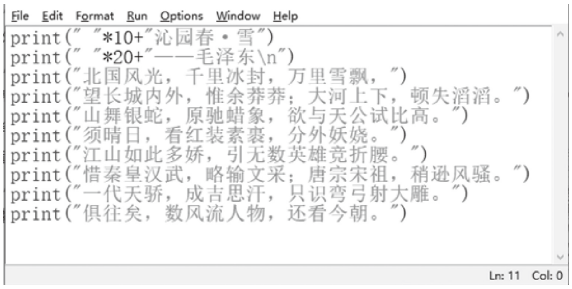

图 1-17　IDLE 文件编辑窗口

2）程序代码输入完成后，在菜单栏中选择“Run >> Run Module”进行程序的调试和运行（也可以直接按下快捷键 <F5>）。如果是新建的文件，这时会弹出一个“保存文件”的对话框，则需进行文件的保存操作。需要注意的是，保存的文件扩展名应是 .py（如保存为 demo.py），确定后将在 IDLE Shell 窗口中输出程序结果（见图 1-18）。

图 1-18　程序运行结果

（3）Python IDLE 常用快捷键。在程序开发过程中，合理使用快捷键不但可以减少代码的错误率，还可以提高程序开发效率，IDLE 快捷键见表 1-1 所示。用户也可通过选择“Options >> Configure IDLE”菜单项，在打开的“Settings”对话框的“Keys”选项卡中查看。

表 1-1　IDLE 快捷键

快提键	说明	适用范围
F1	打开 Python 帮助文档	Python 文件窗口和 Shell 均可用
Alt+P	浏览历史命令（上一条）	仅 Python Shell 窗口可用
Alt+N	浏览历史命令（下一条）	仅 Python Shell 窗口可用
Alt+/	自动补全前面曾经出现过的单词，如果之前有多个单词具有相同前缀，可以连续按下该快捷键，在多个单词中间循环选择	Python 文件窗口和 Shell 窗口均可用
Alt+3	注释代码块	仅 Python 文件窗口可用
Alt+4	取消代码块注释	仅 Python 文件窗口可用
Alt+g	转到某一行	仅 Python 文件窗口可用
Ctrl+Z	撤销一步操作	Python 文件窗口和 Shell 窗口均可用
Ctrl+Shift+Z	恢复上一次的撤销操作	Python 文件窗口和 Shell 窗口均可用
Ctrl+S	保存文件	Python 文件窗口和 Shell 窗口均可用
Ctrl+]	缩进代码块	仅 Python 文件窗口可用
Ctrl+[	取消代码块缩进	仅 Python 文件窗口可用
Ctrl+F6	重新启动 Python Shell	仅 Python Shell 窗口可用

2. 使用 PyCharm 编写第一个 Python 程序

（1）双击打开安装好的 PyCharm 软件，进入欢迎页面，选择 New Project 新建项目按钮（见图 1-19）。

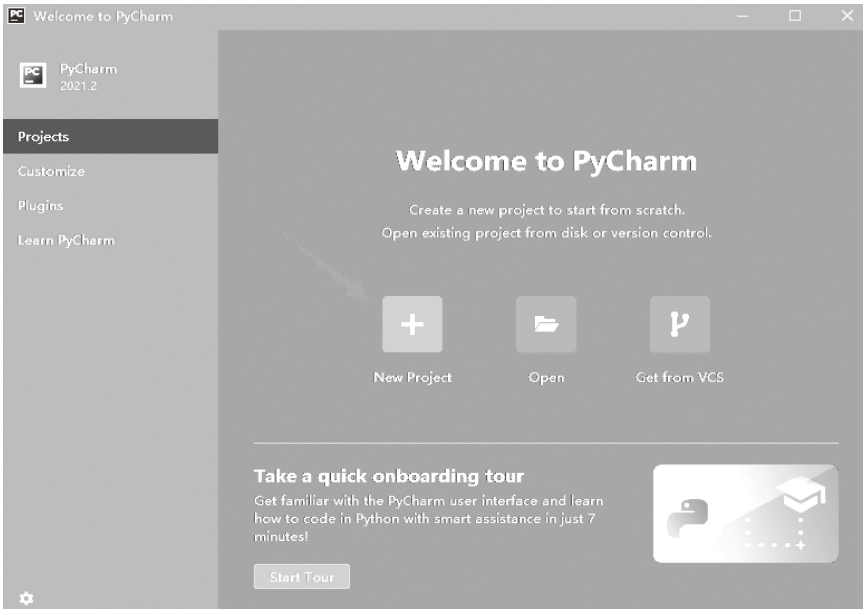

图 1-19　PyCharm 欢迎页面

（2）在新建项目页面中 Location 文本框中可修改项目名称和存放位置，其他内容使用默认值即可，修改完成后点击 Create 按钮自动创建项目（见图 1-20）。

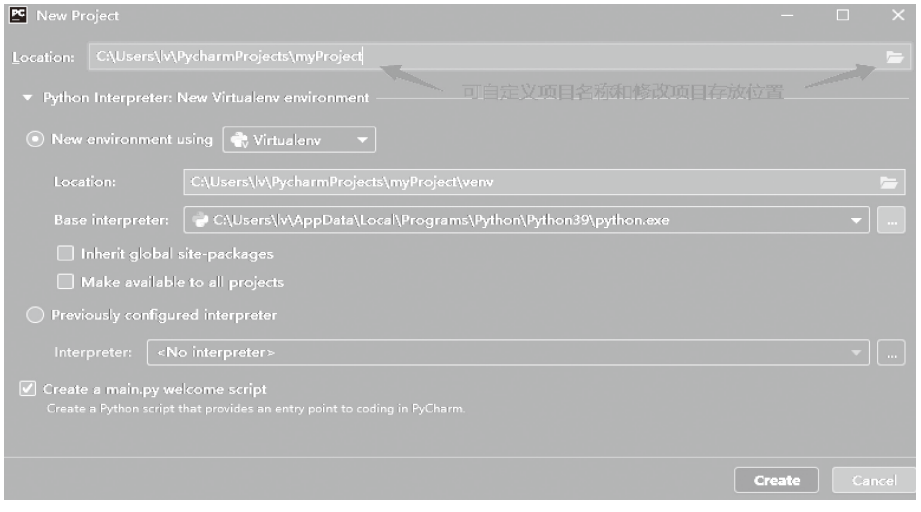

图 1-20　PyCharm 创建项目

（3）项目新建成功后，自动进入 PyCharm 主窗口，PyCharm 主窗口与绝大多数 Windows 窗口的布局一样，上边是菜单栏，左边是项目目录结构，右边是代码编辑区，下边是控制台

输出结果区。我们在代码编辑区中输入程序代码，然后点击菜单栏中的 Run 按钮运行程序，如果程序代码没有语法错误，则可在窗口下方的控制台查看程序输出结果（见图 1-21）。

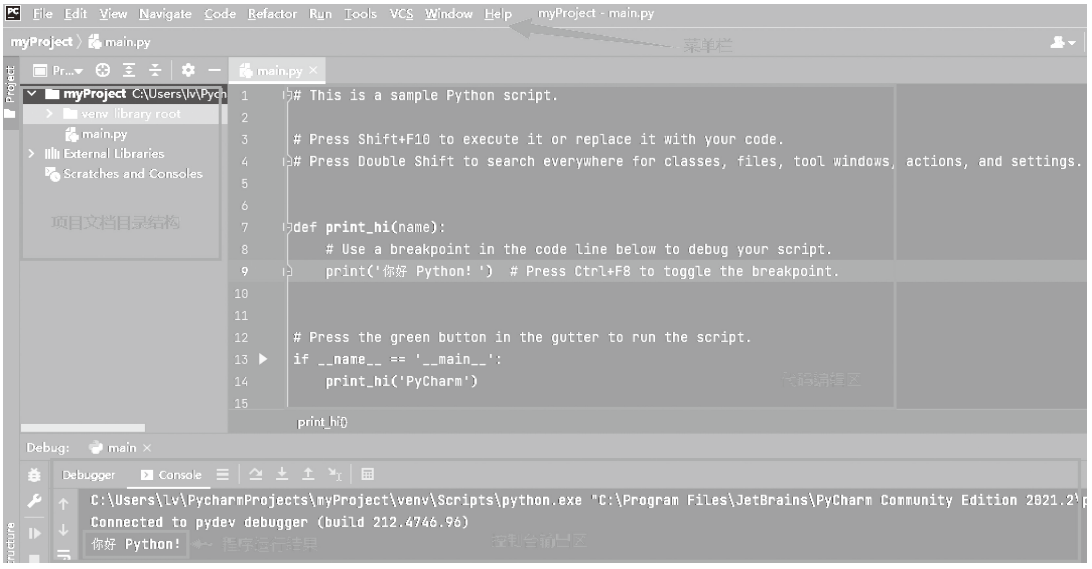

图 1-21　PyCharm 主窗口

总　结

- 了解计算机是由哪些硬件和软件组成的；
- 理解计算机的工作原理（冯・诺依曼原理）；
- 认识什么是机器语言、汇编语言和高级语言；
- 了解 Python 语言的诞生与发展，以及其特点；
- 掌握 Python IDLE 和 PyCharm 的下载与安装。

拓展阅读

习近平总书记指出，新时代抓发展，必须更加突出发展理念，坚定不移贯彻创新、协调、绿色、开放、共享的新发展理念。Python 拥有丰富的标准库和第三方库，支持各种编程范式和领域，例如 Web 开发、数据分析、人工智能等。此外，Python 还支持高级特性，如列表推导式、生成器、装饰器等，为程序员提供了丰富的工具和技术，它的开源性也为开发者提供了更开放、灵活和自由的编程环境。

习　题

一、选择题

1. 计算机系统中执行算术运算和逻辑运算的功能部件是（　　）。

A. ALT　　B. CPU　　C. ALU　　D. DBMS

2. 所谓“裸机”是指（　　）。

A. 单片机　　B. 没安装任何软件的计算机

C. 单板机　　D. 只安装操作系统的计算机

3. 计算机软件通常分为（　　）。

A. 高级软件和一般软件　　B. 管理软件和控制软件

C. 系统软件和应用软件　　D. 专业软件和大众软件

4. Python 语言属于（　　）。

A. 机器语言　　B. 汇编语言　　C. 高级语言　　D. 以上都不是

5. 用 Python 语言编写的程序，需要经过（　　）后计算机才能识别。

A. 汇编　　B. 编译　　C. 解释　　D. 连接

6. Python 程序文件的扩展名是（　　）。

A. .exe　　B. .php　　C. .doc　　D. .py

7. Python 内置的集成开发工具是（　　）。

A. Python Win　　B. Pydev　　C. PyCharm　　D. IDLE

8. Python 解释器的命令提示符是（　　）。

A. >　　B. >>　　C. >>>　　D. #

二、解答题

1. 冯 • 诺依曼原理的内容是什么？

2. 程序的编译和解释有什么不同？

3. Python 程序为什么能够跨平台运行？

三、操作题

下载安装 Python IDLE 和 PyCharm 后，编写第一个 Python 程序输出一行文字。

第 2 章　Python 语言基础知识

内容要点：

- Python 语言程序编写规范；
- Python 语言的保留字与标识符；
- Python 语言中变量的赋值与使用；
- Python 语言中的基本数据类型（数字、布尔值、字符串）；
- Python 中有关字符串常用函数和方法；
- Python 语言中的各种运算符。

思政目标：

- 通过本章的学习，强化规范意识。

2.1　Python 语言程序编写规范

每一门编程语言都有自己的语法规则和编写规范，语法就是编写程序时需要遵循的一些规则约定。掌握编程语言的语法规则是学习该门语言的第一步，也是最简单的环节。规范的代码可以减少程序 Bug，提高程序可读性，降低维护成本以及有利于团队合作开发。本节详细介绍 Python 语言的语法规范。

2.1.1　Python 语言程序书写规范

1. 代码行

在代码编辑器中可以一行写多条代码，代码之间用分号（;）隔开，但不建议这样做。建议每行只写一条代码，且每行不要超过 80 个字符，如果超过，建议使用小括号将多行内容隐式地连接起来，而不推荐使用反斜杠“\”进行连接。代码如下：

```
if (width == 0 and height == 0 and
          color == 'red' and emphasis == 'strong'):
x = (‘这是一个很长很长很长很长很长很长很长很长’
      ‘很长很长很长很长很长很长很长很长的字符串’)
```

注意：此编程规范适用于绝对大多数情况，但以下两种情况除外：

- 导入模块的语句过长时，请写成一行。
- 注释里的长 URL 地址时，请写成一行。

2. 缩进

Python 使用缩进来表示代码的包含关系，比如，在连续的代码行中，缩进相同的行被认为是同一级别的程序块。代码如下：

```
class PythonLanguage:                    # 代码前没有空格
    def __init__(self,name):            # 代码前有 4 个空格
        self.name = name                 # 代码前有 8 个空格
    def setname(self,name):
        self.name = name
    def getname(self):
        return self.name
pl= PythonLanguage ("Python 语言 ")     # 代码前没有空格
print(pl.getname())
pl.setname("Python 程序设计 ")
print(pl.getname())
print(pl.getname())
```

Python 没有强制要求用 Tab 键缩进还是用空格键缩进，甚至空格按几个都没有强制要求。建议大家用 4 个空格来缩进代码，不要用 Tab 键，绝对不允许 Tab 和空格混用。原因是 Tab 键在 ASCII 码中，编码是 9，而空格是 32，当我们按下一个 Tab 键的时候，有的编辑器中它表示 4 个空格，而另外的编辑器有可能表示的是 8 个空格，所以在一个编辑器里混用 Tab 和空格键设置缩进后，在其他编辑器里有可能缩进就乱了。如果统一使用空格键就不会出现这个问题，因为一个空格就占一个字符的位置。

3. 空行

可以用空行来从形式上对程序代码块表示间隔，有助于提高代码的可读性。代码如下：

```
foo = 1000
long_name = 2
                                         # 空行
dictionary = {
     "foo": 1,
     "long_name": 2,
     }
```

4. 注释

用来向用户提示或解释某些代码的作用和功能，提高程序的可读性，在调试（Debug）程序的过程中，注释还可以用来临时移除无用的代码。它可以出现在代码中的任何位置。Python 解释器在执行代码时会忽略注释，不做任何处理，就好像它不存在一样。注释往往容易被初学者所忽视，但每个成熟的程序员都能深刻理解注释的重要性，所以一般软件企业都会有明确的规定，要求必须要写注释，合理的代码注释应该占源代码的 1/3 左右。Python 支持两种类型的注释，分别是“单行注释”和“多行注释”。

（1）Python 使用井号（#）作为单行注释的符号，语法格式为：

 # 注释内容

从井号（#）开始，直到这行结束为止的所有内容都是注释。Python 解释器遇到 # 时，会忽略它后面的整行内容。用于注释单行代码的功能时一般将注释放在代码的右侧。代码如下：

```
print("Hello Python!")        # 使用 print 输出 Hello Python!
print(100)                    # 使用 print 输出数字
```

用于注释多行代码的功能时一般将注释放在代码的上一行。代码如下：

```
# 使用 print 输出字符串
print("Hello Python!")
print(" 欢迎进入 Python 的世界！ ")
# 使用 print 输出数字
print(100)
print( 3 + 100 * 2)
```

（2）多行注释指的是一次性注释程序中多行的内容（包含一行）。多行注释通常用来为 Python 文件、模块、类或者函数等添加版权或者功能描述信息。Python 使用三个连续的单引号（'''）或者三个连续的双引号（"""）注释多行内容。格式如下：

```
def max（input_list）:
    """ 计算输入列表中元素的最大值，首先判断输入的是否是列表，如果是列表，则通过
for 循环语句计算出最大值并返回，如果输入的不是列表，则返回 None。
    """
        if isinstance(input_list, list):
            ret = input_list[0]
            for x in input_list:
                if x > ret:
```

```
                    ret = x
            return ret
        else:
            return None
```

（3）需要注意的是不管是单行注释还是多行注释，当注释符作为“字符串”的一部分出现时，就不能再将它们视为注释标记，而应该看做正常代码的一部分。代码如下：

```
print('''Hello Python!''')
print("""# 是单行注释的开始 """)
```

执行结果是：

```
Hello Python!
# 是单行注释的开始
```

5. 空格分隔

在运算符两侧、函数参数之间以及逗号两侧，可使用空格进行分隔。

6. 符号

除了符号本身作为了字符串的情况下，在 Python 语言中所有的符号都必须使用英文符号。

2.1.2 Python 保留字与标识符

1. Python 保留字

保留字又可称为关键字，是在 Python 语言中已经被赋予特定意义的 33 个单词，开发者在开发程序时，不能用这些保留字作为标识符给变量、函数、类、模板以及其他对象命名。所有 Python 的关键字只包含小写字母。可以在 IDLE Shell 命令行中执行如下命令查看 Python 包含的保留字：

```
>>> import keyword
>>> keyword.kwlist
```

运行结果如下：

```
    and、as、assert、break、class、continue、def、del、elif、else、
except、
    finally、for、from、False、global、if、import、in、is、lambda、
nonlocal、
    not、None、or、pass、raise、return、try、True、while、with、yield
```

2. Python 内置函数

函数就是将需要重复使用的代码封装起来，并给这段封装的代码起一个名字，以后需要使用该段代码功能的时候只要调用名字就可以了，这样有助于大大提高编程效率。Python 语言中的函数包括：内置函数、标准函数、扩展函数（自定义函数）。

（1）内置函数。Python 解释器自带的函数叫做内置函数，Python 解释器启动以后，内置函数也生效了，这些函数可以直接拿来使用，不需要导入某个模块。

（2）标准函数。Python 标准库相当于解释器的外部扩展，它并不会随着解释器的启动而启动，要想使用这些外部扩展，必须提前导入。Python 标准函数库非常庞大，包含了很多模块，要想使用某个函数，必须提前导入对应的模块，否则函数是无效的。

一般来说，内置函数的执行效率要高于标准库函数。但内置函数的数量必须被严格控制，否则 Python 解释器会变得庞大和臃肿。只有那些使用频繁或者和语言本身绑定比较紧密的函数，才会被提升为内置函数。例如，在屏幕上输出文本是使用最频繁的功能之一，因此 print() 是 Python 的内置函数。除了 print() 函数以外，Python 解释器还提供了更多的内置函数，表 2-1 列出了 Python 3.x 中的所有内置函数。

表 2-1　Python 3.x 内置函数

abs()	delattr()	hash()	memoryview()	set()	all()
dict()	help()	min()	setattr()	any()	dir()
hex()	next()	slicea()	ascii()	divmod()	id()
object()	sorted()	bin()	enumerate()	input()	oct()
staticmethod()	bool()	eval()	int()	open()	str()
breakpoint()	exec()	isinstance()	ord()	sum()	bytearray()
filter()	issubclass()	pow()	super()	bytes()	float()
iter()	print()	tuple()	callable()	format()	len()
property()	type()	chr()	frozenset()	list()	range()
vars()	classmethod()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	__import__()	complex()
hasattr()	max()	round()			

3. Python 标识符

标识符就是一个名字，就像我们每个人都有属于自己的名字，它的主要作用就是作为变量、函数、类、模块以及其他对象的名称。Python 中标识符的命名要遵守下述命令规则：

（1）标识符是由字母（A ～ Z 和 a ～ z）、下划线和数字组成，但第一个字符不能是数字。

（2）不能使用 Python 中的保留字和内置函数名作为标识符。

（3）Python 中的标识符中，不能包含空格、@、% 以及 $ 等特殊字符。例如合法标识符：UserID，name，mode12，user_age；非法标识符：try，12mode，$money，user age。

（4）在 Python 中标识符的字母是严格区分大小写的，也就是说，两个同样的单词，如果大小格式不一样，代表的意义是完全不同的。例如：

number = 100

Number = 100

NUMBER = 100

表示 3 个不同的变量。

（5）在 Python 语言中，以下划线开头的标识符有特殊含义（后面章节中有专门介绍），应避免使用以下划线开头的标识符。例如：

1）以单下划线开头的标识符（如： _width），表示不能直接访问的类属性，其无法通过 from...import * 的方式导入；

2）以双下划线开头的标识符（如：__add）表示类的私有成员；

3）以双下划线作为开头和结尾的标识符（如： __init__），是专用标识符。

（6）Python 允许使用汉字作为标识符，例如：姓名 = “LiLei”，但为了避免一些汉字编码错误，我们应尽量避免使用汉字作为标识符。

除了要遵守以上这几条规则外，我们还应该养成良好的标识符的命名习惯。例如：

（7）当标识符用作模块名、函数名、类中的属性名和方法名时，应尽量短小，并且全部使用小写字母，如果名称由多个单词组成，可以使用下划线进行分割，例如：user_name、user_age 等。

（8）当标识符用作包的名称时，应尽量短小，也全部使用小写字母，不推荐使用下划线。

（9）当标识符用作类名时，应采用单词首字母大写的形式。例如，定义一个图书类，可以命名为 Book。

2.1.3 Python 之禅

前面小节介绍了 Python 程序的编写规范，关于 Python 语言程序开发的原则 Tim Peters 还专门进行了总结，被称为”Zen of Python”，也被称为“Python 之禅”。

这些编码的原则被 Python 社区广泛接受，因此最后被放入到各个 Python 解释器中。我们可以在 Python 解释器中输入 import this 即可看到这个“Python 之禅”的具体内容（见图 2-1）。当然它是用英文编写的，这里做了一下简单的翻译和解释。

```
>>> import this
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```

图 2-1　Python 之禅

“Python 之禅”的内容翻译成中文如下：

（1）优美漂亮的代码优于丑陋的代码（就是说我们不仅要求代码能够正常工作，而且还希望代码看起来优美）。

（2）明确优于隐含（简单来说就是我们的代码要明确说明其用法，不要让用户根据他们自己的理解来猜）。

（3）简单优于复杂（能用简单方法就一定不要故意给自己找麻烦，最简单的方法就是最好的方法）。

（4）复杂胜于凌乱（如果功能很复杂，则希望能够将其分割成功能单一的多个模块；希望保持模块间接口函数简洁，保证各个模块功能单一）。

（5）扁平优于嵌套（就是尽量不要使用嵌套，毕竟嵌套代码在调试时，定位问题比较麻烦，不知道是在哪一层嵌套时出的问题）。

（6）宽松优于紧凑（各个代码模块之间的联系要简单，不能过于依赖某些模块，不要不同模块之间的联系过于复杂而形成蜘蛛网状）。

（7）代码可读性很重要（变量名、函数名、类名最好有明确的含义。注释也是很重要的，注释可以帮助我们和他人来理解代码）。

（8）即便是特例，也不可违背上述规则（所谓的特例就是这样一些情况，如果我们不遵守这些规则，看起来在目前更加划算。但是如果我们的代码会长期服务于我们，那么遵守这些规则最终会让我们受益）。

（9）虽然现实往往不那么完美，但是不应该放过任何异常（对异常的处理非常重要，90% 的问题就发生在那些边角用例中）。

（10）对异常处理不可马虎（虽然多数异常出现概率很低，但是我们不能掉以轻心，