

Python 概述

Python 是一种解释型、跨平台、面向对象的计算机程序设计语言,具有开源性、易学性、可移植性、可扩展性和支持丰富类库等特点。Python 凭借扩展性强、第三方库丰富和免费开源等特点,在机器学习、数据挖掘、人工智能等领域有着很大优势,前景非常值得期待。

1.1 Python 语言简介

目前,Python 是一种知名度高、影响力大、应用广泛的主流编程语言。

1.1.1 Python 的诞生

Python 语言是荷兰首都阿姆斯特丹(Amsterdam)的多·范·罗苏姆(Guido van Rossum)开发的。他出生于 1956 年 1 月 31 日,1982 年他在阿姆斯特丹大学获得数学和计算机科学硕士学位。1989 年 12 月圣诞节期间,33 岁的他在阿姆斯特丹为了打发圣诞节的无趣,决心开发一个新的脚本解释程序。之所以选中 Python(大蟒蛇)作为该编程语言的名字,是取自英国 20 世纪 70 年代首播的电视喜剧片《蒙提·派森的飞行马戏团》(*Monty Python's Flying Circus*)。他希望这个新的叫作 Python 的语言,能实现他的理念,即开发一种介于 C 和 Shell 之间,功能全面、易学易用并可拓展的语言。他用 3 个月的时间做出原型。1991 年,第一个 Python 编译器(同时也是解释器)诞生,它是用 C 语言实现的,并能够调用 C 库(.so 文件)。Python 最初就具有类(class)、函数(function)、异常处理(exception),包括列表(list)和词典(dictionary)在内的核心数据类型,并包括模块(module)为基础的拓展系统。

最初的 Python 完全由 Guido 一人开发。后来 Python 得到 Guido 同事的欢迎,他们也参与进来对 Python 加以改进。Guido 和同事组成了 Python 的核心团队。Python 将许多机器层面的细节隐藏,交给编译器处理,并凸显出逻辑层面的编程思考。Python 程序员可以有更多的时间用于思考程序的逻辑,而不是具体的实现细节。

后来罗苏姆在多个研究机构工作,这些机构包括在荷兰阿姆斯特丹的国家数学和计算机科学研究学会(CWI),在马里兰州 Gaithersburg 的国家标准及技术研究所(NIST),以及维珍尼亚州 Reston 的国家创新研究公司(CNRI)。

2005 年 12 月, Guido 加入 Google。他用 Python 语言为 Google 写了面向网页的代码浏览工具。

2002 年, 在比利时布鲁塞尔举办的自由及开源软件开发欧洲会议上, Guido 获得了由自由软件基金会颁发的 2001 年自由软件进步奖。2003 年 5 月, Guido 获得了荷兰 UNIX 用户小组奖。2006 年, 他被美国计算机协会 (ACM) 认定为著名工程师。

2011 年 1 月, Python 被 TIOBE 编程语言排行榜评为 2010 年度语言。自从 2004 年以后, Python 的使用率呈线性增长。目前 Python 已经成为最受欢迎的计算机语言之一。

2019 年 10 月 30 日, 63 岁的 Guido 宣布退休。

1.1.2 Python 的优缺点

1. Python 的优点

(1) Python 的定位是“优雅”“明确”“简单”, 所以 Python 程序看上去简单易懂, 初学者学 Python 不但入门容易, 而且深入学习后可以编写非常复杂的程序。

(2) 开发效率非常高。Python 有非常强大的第三方库, 基本上想通过计算机实现的各种功能在 Python 官方库里都能找到。开发项目时直接下载并调用一些库文件, 然后在基础库的基础上再进行编程, 从而大幅缩短了开发周期, 避免了重复性劳动。

(3) Python 是面向对象的高级语言, 支持将代码封装在对象中的编程技术。

(4) Python 是解释型语言, 在开发过程中没有编译环节。

(5) Python 是交互式语言, 可以在 Python 提示符后直接互动执行代码。

(6) 免费、开源是 Python 被广泛使用的原因之一, 程序员可以免费使用许多程序, 而不存在版权问题。

(7) 基于其开放源代码的特性, Python 已经被移植(也就是使其工作)到许多平台。

(8) 有较好的可扩展性。如果需要一段运行很快的关键代码, 或者是想要编写一些不愿开放的算法, 可以使用 C 或 C++ 完成那部分程序, 然后在 Python 程序中调用。

(9) Python 提供所有主要商业数据库的接口。

(10) Python 支持 GUI(graphical user interface, 图形用户界面), 可以创建并移植到许多系统中进行调用。

2. Python 的缺点

(1) 代码不能加密。因为 Python 是解释型语言, 它的源代码都是以明文形式存放的(不过这也许不能算是一个缺点)。

(2) 对多线程支持度不高。

(3) 速度慢。Python 相对 C 语言要慢得多, 但大多数用户是感觉不到的。

(4) Python 作为解释型动态语言, 通常需要程序员自查代码安全。而 Java 和 C 等静态类型语言, 这些问题会在编译时检查出来。

(5) 缺少其他高级语言具备的一些功能。为了增加相关功能, Python 通过变通的方式实现, 从而导致代码变得复杂。

(6) 由于 Python 是由许多业余人士逐渐丰富并完善的语言,相对 Java、C# 等语言,缺乏整体设计,造成规范繁杂、混乱。

1.1.3 Python 的应用领域

(1) 系统编程。Python 提供 API(application program interface,应用程序接口),能方便地进行系统维护和管理,很多系统管理员认为它是理想的编程工具。

(2) 图形界面开发。Python 在图形界面开发方面很强大,可以用 Tkinter/PyQT 框架开发各种桌面软件。

(3) 科学计算。Python 是一门很适合做科学计算的编程语言。随着 NumPy、SciPy、Matplotlib、Enthought librarys 等众多程序库的开发,使 Python 越来越适合做科学计算并绘制高质量的二维和三维图像。

(4) 文本处理。Python 提供的 re 模块支持正则表达式,还提供 SGML、XML 分析模块,许多程序员利用 Python 进行 XML 程序的开发。

(5) 数据库编程。可通过 Python DB-API(数据库应用程序接口)规范的模块与 Microsoft SQL Server、Oracle、Sybase、DB2、MySQL、SQLite 等数据库通信。另外,Python 自带一个 Gadget 模块,提供了一个完整的 SQL 环境。

(6) 网络编程。Python 提供丰富的模块支持 Socket 编程,能方便快速地开发分布式应用程序。

(7) Web 开发。Python 拥有很多免费函数库、免费 Web 网页模板系统,以及与 Web 服务器进行交互的库,可以实现 Web 开发,搭建 Web 框架。

(8) 自动化运维。Python 是运维人员广泛使用的语言,能满足绝大部分自动化运维需求,包括前端和后端。

(9) 金融分析。利用 Numpy、Pandas、Scipy 等数据分析模块,可快速完成金融分析工作。目前,Python 是金融分析、量化交易领域使用最多的语言。

(10) 多媒体应用。Python 的 PyOpenGL 模块封装了“OpenGL 应用程序编程接口”,能进行二维和三维图像处理。

(11) 网络爬虫。在爬虫领域,Python 几乎处于霸主地位,提供了 Scrapy、Request、BeautifulSoup、urllib 等工具库,将网络中的一切数据作为资源,通过自动化程序进行有针对性的数据采集以及处理。

(12) 游戏开发。Python 在网络游戏开发中也有很多应用。Python 非常适合编写超过 1 万行代码的项目,而且能够很好地把网游项目的规模控制在 10 万行代码以内。

(13) 人工智能。NASA 和 Google 早期大量使用 Python,为 Python 积累了丰富的科学运算库。当 AI(artificial intelligence,人工智能)时代来临后,Python 从众多编程语言中脱颖而出,各种 AI 算法都基于 Python 编写。在神经网络、深度学习方面,Python 都能够找到比较成熟的程序包来加以调用。另外,Python 是面向对象的动态语言,适用于科学计算,使其在人工智能方面备受青睐。

1.1.4 Python 发展历史

1989 年, Guido 开始编写 Python 语言的编译器, 他希望这个叫作 Python 的语言能符合他的理想。

1991 年, 第一个 Python 编译器诞生。它是用 C 语言来实现的, 并能够调用 C 语言的库文件。Python 从一出现便具有类、函数、异常处理等功能, 包含表和词典在内的核心数据类型, 以及以模块为基础的拓展系统。

Python 3 相对 Python 早期的版本是一个较大的升级。Python 3 在设计的时候没有考虑向下兼容, 所以很多早期版本的 Python 程序无法在 Python 3 上运行。

Python 2.7 是 2.x 系列的最后一个版本, 已经停止开发, 不再增加新功能, 2020 年终止支持。对所有最新标准库的更新改进, 只会在 3.x 的版本里出现。Python 3 比 Python 2 更规范统一, 去掉了一些不必要的关键字, 且 Python 3.x 还在持续改进, 所以推荐读者使用 Python 3.x。

据 2020 年 4 月 TIOBE 公布的编程语言排行榜看, 前 5 位的是 Java(16.73%)、C(16.72%)、Python(9.31%)、C++(6.78%)、C#(4.74%), 括号中的百分数是使用占比。

1.2 安装与配置 Python 程序开发环境

1.2.1 IDLE 的下载与安装

IDLE 是开发 Python 程序的基本集成开发环境(IDE), 具备基本 IDE 的功能。IDLE 可以被看作是简易版的集成开发环境, 是学习 Python 不错的选择。下面介绍 Windows 下 IDLE 的安装。

(1) 在浏览器地址栏中输入 Python 官网的下载地址 <https://www.python.org/downloads/>, 显示下载网页, 如图 1-1 所示。由于网页经常改变, 大家看到的网页可能不同。单击“Download Python 3.8.3”, 将 Python 安装文件下载到指定的文件夹中。

(2) 单击“运行”按钮, 打开“打开文件-安全警告”对话框, 单击“运行”按钮。

(3) 显示 Python 安装向导, 如图 1-2 所示, 选中“Add Python 3.8 to PATH”复选框, 会自动配置环境变量。Install Now 是默认安装选项, Customize installation 是安装时自定义安装的文件。选择 Install Now 的安装路径为 C:\Users\Administrator\AppData\Local\Programs\Python\Python38-32, 这个安装路径在安装其他模块时不容易出错; 再单击 Install Now。

(4) 显示 Setup Progress 对话框, 如图 1-3 所示。安装过程需要几分钟。

(5) 显示 Setup was successful 对话框, 如图 1-4 所示, 表示安装完成。单击 Close 按钮, 关闭安装向导对话框。

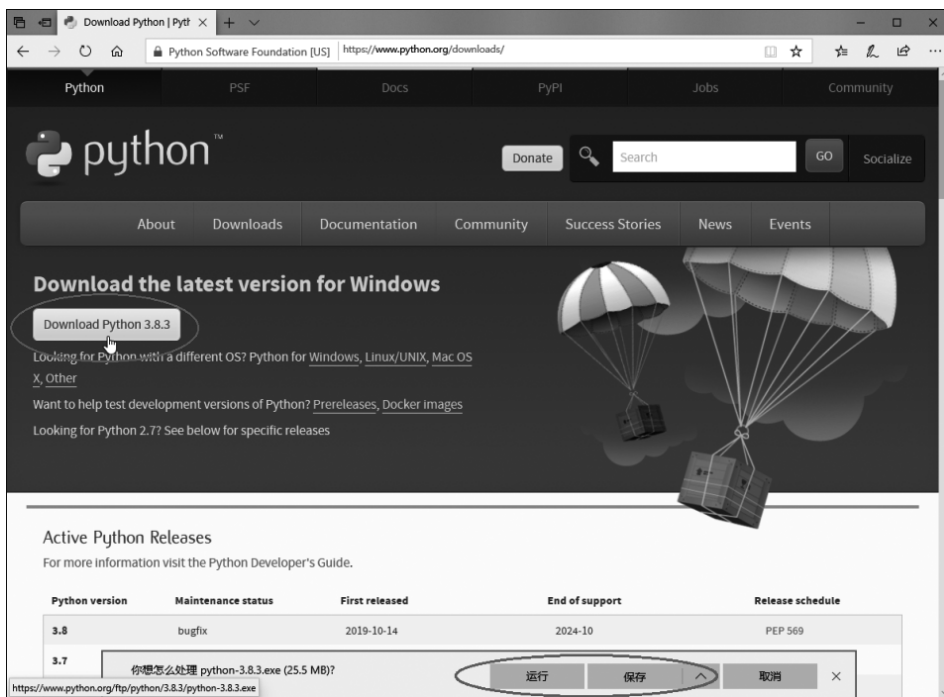


图 1-1 选择要下载的版本

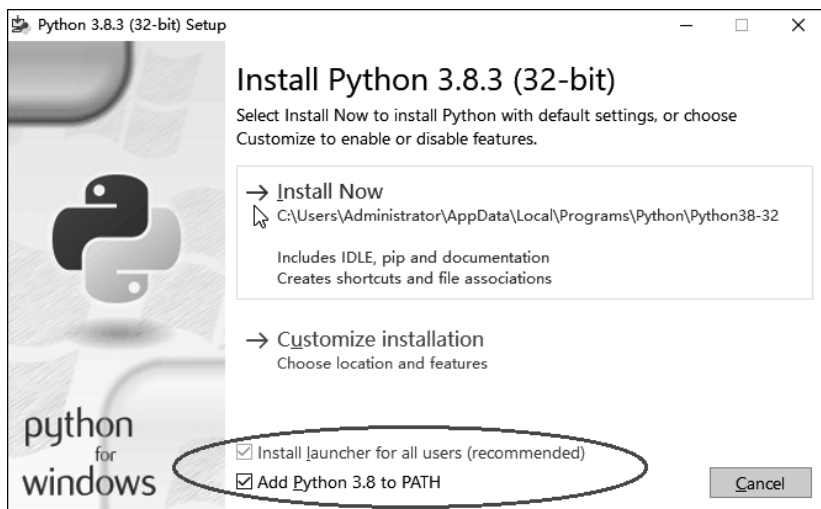


图 1-2 安装向导对话框

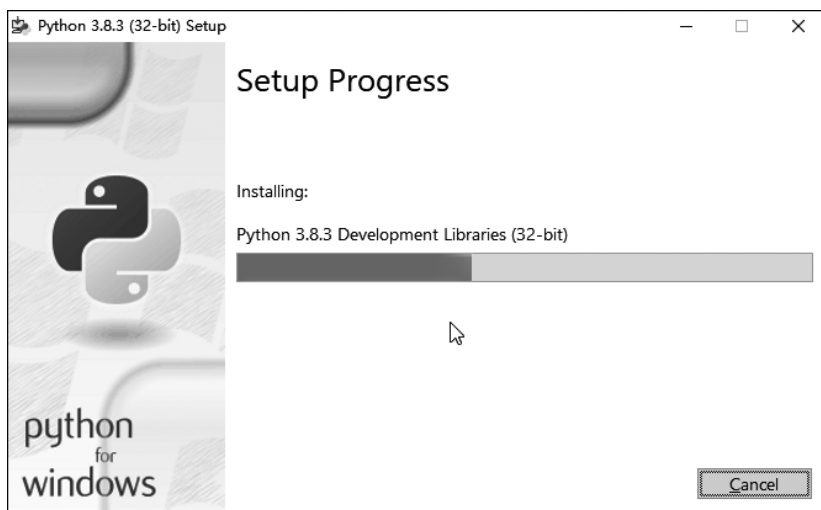


图 1-3 显示 Setup Progress 对话框



图 1-4 Setup was successful 对话框

1.2.2 Python 的菜单选择和启动

Python 安装成功以后,在 Windows 的“开始”菜单中展开“Python 3.8”文件夹,显示的菜单选项如图 1-5 所示。

1. IDLE(Python 3.8 32-bit)

用“IDLE (Python 3.8 32-bit)”选项可以打开 Python 自带的集成开发环境,它是一个 Python Shell



图 1-5 “开始”菜单中的“Python 3.8”文件夹

程序。Shell 是“外壳”的意思,是一个通过输入文本与系统进行交互的程序。

单击 IDLE(Python 3.8 32-bit)选项,显示如图 1-6 所示的窗口。可以在 IDLE 的“>>>”提示符后输入 Python 指令,也可以创建 Python 程序文件。对于初学者,学习 Python 通常使用这个 IDLE 窗口。稍后将介绍该 IDLE 的使用方法。

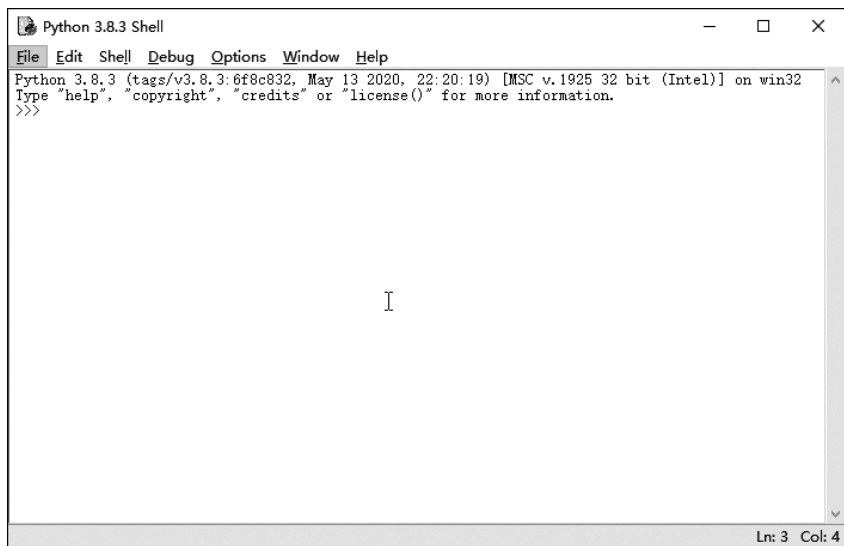


图 1-6 Python 的 IDLE 窗口

2. Python 3.8(32-bit)

单击 Python 3.8(32-bit)选项,可以打开 Python 命令行窗口,如图 1-7 所示。可以在“>>>”提示符后输入 Python 命令。

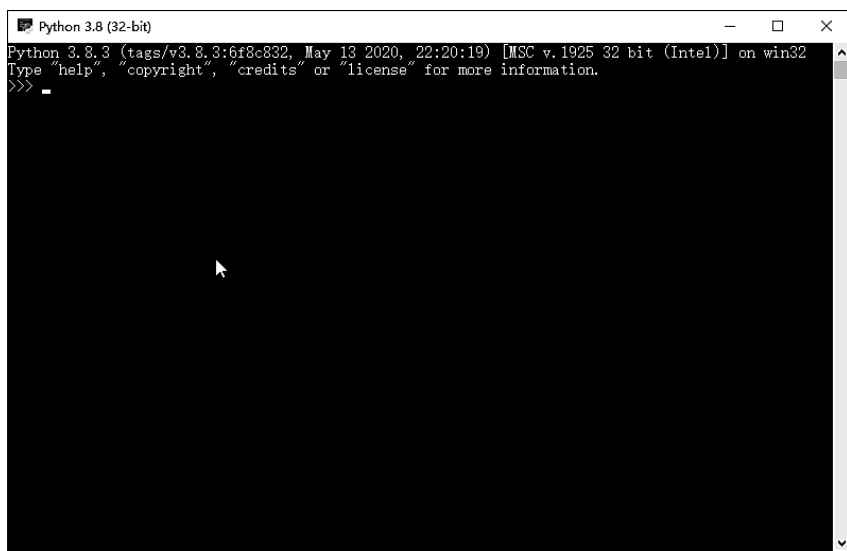


图 1-7 命令行窗口

3. Python 3.8 Manuals(32-bit)

单击 Python 3.8 Manuals(32-bit)选项,可以打开 Python 帮助文档,显示如图 1-8 所示的窗口。

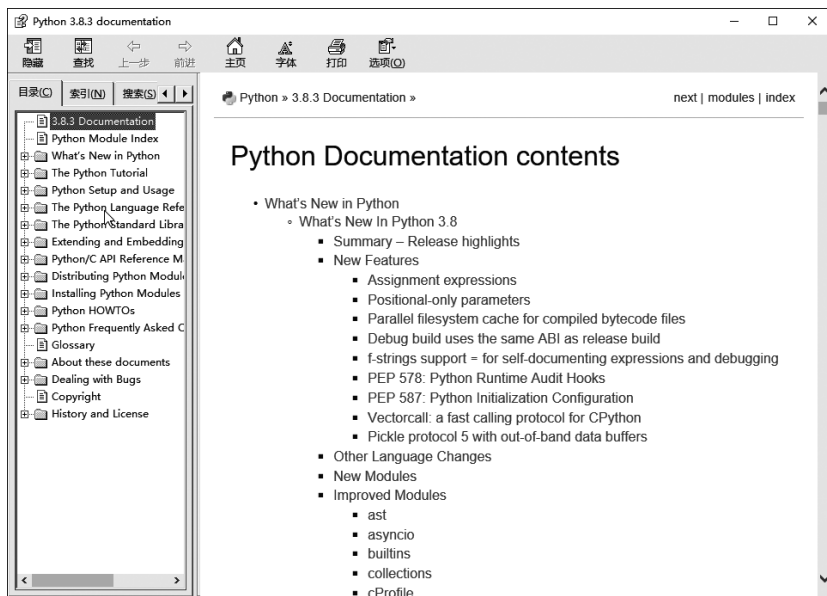


图 1-8 Python 帮助文档

如果希望查看中文帮助文档,请在浏览器地址栏中输入 <https://docs.python.org/zh-cn/3/>,显示的内容如图 1-9 所示。



图 1-9 Python 的在线中文帮助文档

4. Python 3.8 Module Docs(32-bit)

单击 Python 3.8 Module Docs(32-bit)选项,将以内置服务器的方式打开 Python 的模块文档,显示如图 1-10 所示的窗口。

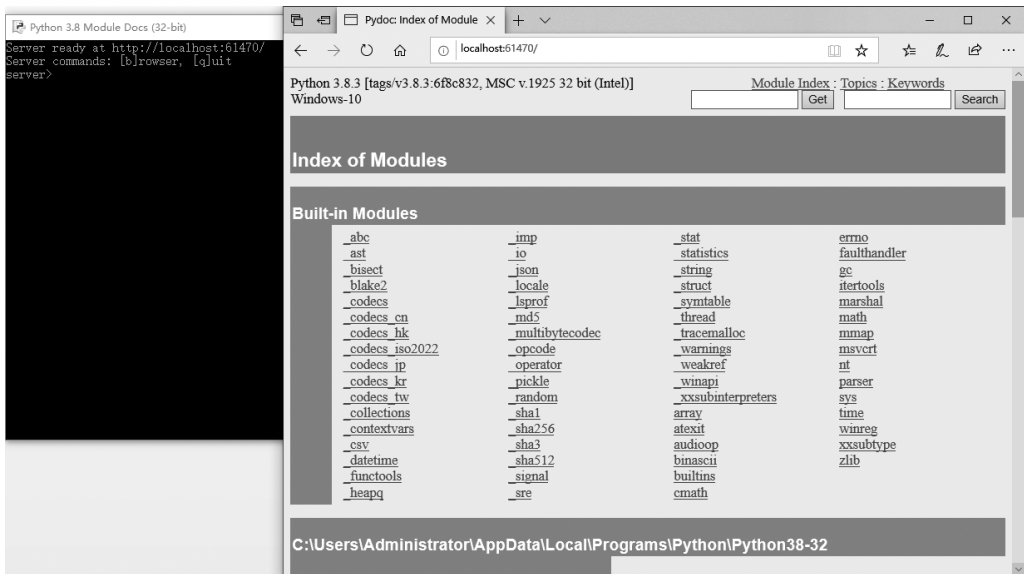


图 1-10 Python 的模块文档

1.2.3 在 IDLE 中编辑、运行程序

IDLE 的基本功能有语法加亮、段落缩进、基本文本编辑、Tab 键控制、调试程序等,可以方便地创建、运行、调试 Python 程序。下面介绍 Python IDLE 的基本用法。

IDLE 启动后的初始窗口中,窗口标题栏显示“Python 3.8.4 Shell”,通过它可以在 IDLE 内部执行 Python 命令。除此之外, IDLE 带有一个交互式解释器,用来解释执行 Python 语句;有一个编辑器,用来编辑 Python 程序(或者脚本);有一个调试器,用来调试 Python 程序脚本。

IDLE(Python 3.8 32-bit)是以 GUI 及菜单方式来执行 Python 命令。对于喜欢以命令行方式来执行命令的程序员,可以启动 Python 3.8(32-bit)。

IDLE 和 Python 命令行两种窗口中都以 3 个大于号“>>>”作为提示符,可以在该提示符后输入命令或语句。在交互方式下,在这两种窗口中执行命令的方式是一样的。本书使用 IDLE 环境来介绍 Python 程序的编程。

提示: 本书约定,在所有章节给出的示例代码中,“>>>”符号都不需要输入,仅表示该代码是在交互方式下运行;而不带该提示符的代码则表示是以程序的方式运行。

1. 交互方式运行命令

如果使用交互式编程模式,直接在提示符“>>>”后输入相应的命令或语句,并按 Enter

键。如果正确执行了命令,在下一行将显示运行结果;否则将抛出异常,给出错误提示。

下面采用交互方式输入简单的语句,计算两个变量相加的值,输出内容显示在提示符下方。

在“>>>”提示符后输入 $x=20$,按 Enter 键;在下一行的“>>>”后输入 $y=30$,按 Enter 键;在新的下一行的“>>>”后输入 $x*y$,按 Enter 键,则下一行显示 $x*y$ 的计算结果 600。

在“>>>”后输入 "Hello,world!",按 Enter 键,则在下一行显示“Hello,world!”。

在“>>>”后输入“2/0”,则给出错误提示,如图 1-11 所示。

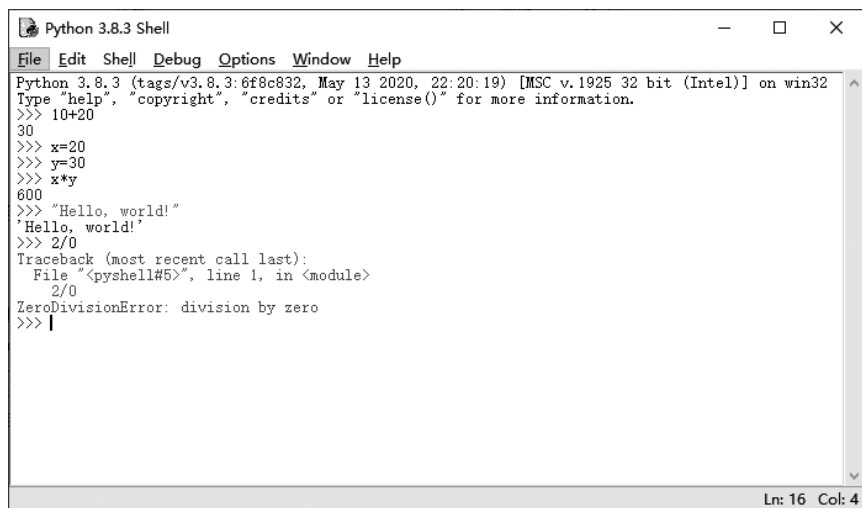


图 1-11 交互方式的运行

在交互方式下,在提示符“>>>”后输入相应的命令或语句,并按 Enter 键,就能显示结果,非常简单。但是输入的命令和语句不能重复利用(可以选中语句后,按 Ctrl+C 组合键复制,然后在最下面一行的“>>>”后按 Ctrl+V 组合键粘贴),也不能保存到硬盘等外存上。因此,交互方式仅仅用于执行最简单的代码,很少用于程序的输入和运行。

2. 程序方式运行

把 Python 语句、命令等程序代码按照一定的业务逻辑编写成程序,并以文件的形式保存到硬盘等外存中,这样的程序文件可以重复调用和编辑。在下面的例题中,介绍在 IDLE 中如何进行程序的新建、输入、编辑、保存和运行。

【例 1-1】 编写第一个程序,输出“Hello, world!”。

```
#Filename: helloworld.py
print("Hello, World!")
x=20
y=30
z=x * y+100
print(z)
```