巍巍交大　百年书香
www.jiaodapress.com.cn
bookinfo@sjtu.edu.cn

丛书策划　张荣昌
责任编辑　王　清　孟海江
封面设计　唐韵设计

软件开发类人才培养系列教材

- .NET开发综合实战
- PHP开发综合实战
- Web前端开发综合实战
- Java开发综合实战
- Java Web应用与开发
- PHP应用与开发
- Bootstrap应用与开发
- Node.js应用与开发
- Vue.js应用与开发
- ASP.NET应用与开发
- 软件测试基础
- 项目管理
- 软件工程与测试
- C语言程序设计
- C#程序设计
- R语言程序设计
- Go语言程序设计
- Android程序设计
- **Python程序设计项目化教程**
- Python 3程序设计实战教程
- Java程序设计
- JavaScript基础教程
- PHP网络编程入门与进阶
- JavaScript+JQuery入门与进阶
- UI交互设计入门与进阶
- HTML+CSS入门与进阶
- HTML5+CSS3 Web前端设计案例教程
- Java程序设计实用案例教程

扫描二维码
关注上海交通大学出版社
官方微信

ISBN 978-7-313-28818-9
9 787313 288189 >
定价：59.80元

软件开发类人才培养系列教材

Python 程序设计项目化教程

主编　刘君尧　王辉静　李坤颖

上海交通大学出版社
SHANGHAI JIAO TONG UNIVERSITY PRESS

软件开发类人才培养系列教材
“互联网+”新形态一体化教材

Python 程序设计项目化教程

主编　刘君尧　王辉静　李坤颖

Python
CHENGXU SHEJI
XIANGMUHUA
JIAOCHENG

本书配套资源

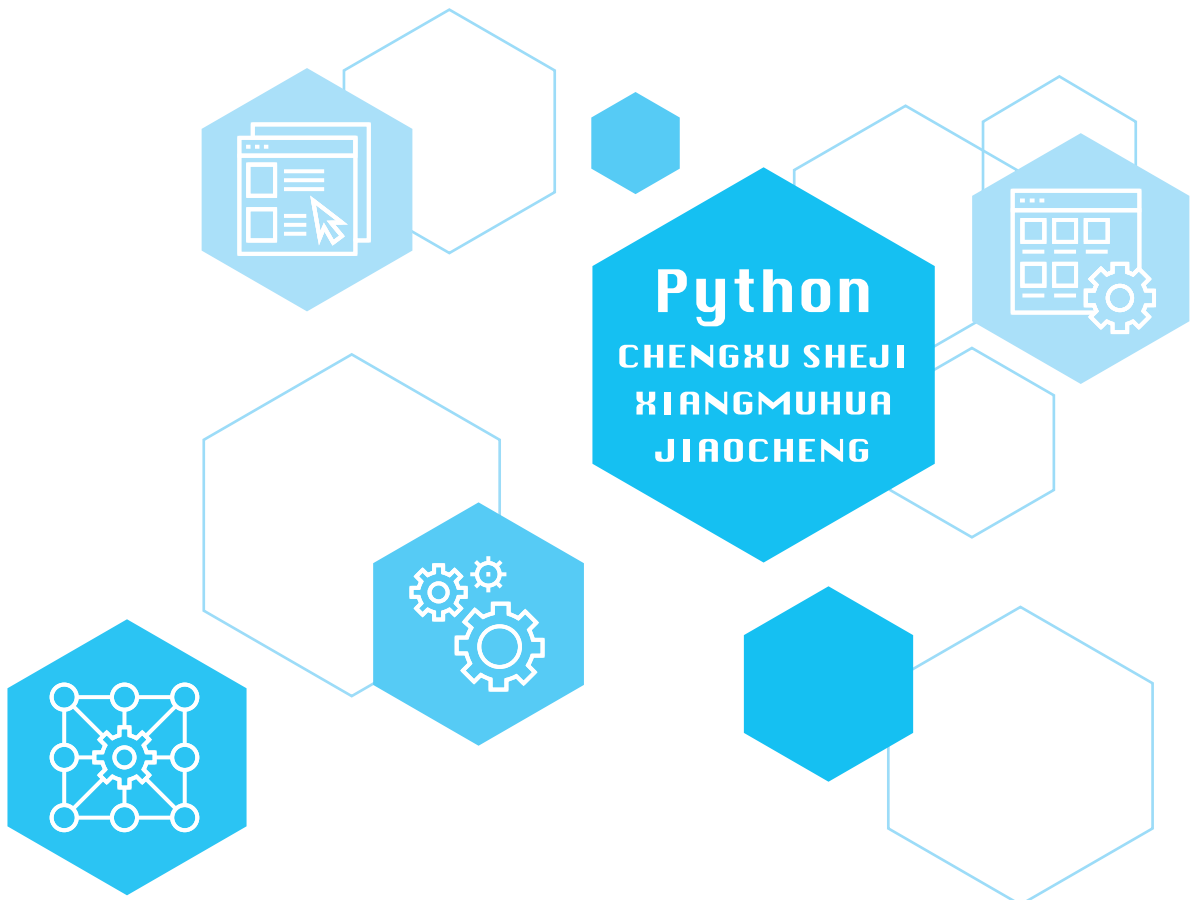
- 微课视频
- 素材源码
- 教学课件
- 电子教案
- 习题答案

上海交通大学出版社
SHANGHAI JIAO TONG UNIVERSITY PRESS

软件开发类人才培养系列教材
“互联网+”新形态一体化教材

Python
程序设计项目化教程

主编　刘君尧　王辉静　李坤颖



上海交通大學出版社
SHANGHAI JIAO TONG UNIVERSITY PRESS

内容提要

本书参考最新教学标准和课程改革成果，结合计算机等级考试要求和计算机相关专业的能力需求编写而成。通过 12 个项目的具体实施详细讲解了 Python 程序设计的相关知识，包括编程输出“社会主义核心价值观”、实现找零换算、跳水单人项目计分、显示杨辉三角、实现用户登录功能，实现通讯录功能，绘制正弦函数、创建客户信息簿、分词处理器、模拟智能机房管理、动态时钟和系统信息采集。本书提供了丰富的数字化教学资源，包括相关的课程标准，授课计划、教学课件、微课视频和习题解答，可作为计算机，软件和通信类相关专业的教材，也可作为计算机等级考试的培训用书。

图书在版编目（CIP）数据

Pyhon 程序设计项目化教程 / 刘君尧，王辉静，李坤颖主编 . — 上海：上海交通大学出版社，2023.7（2025.1 重印）

ISBN 978-7-313-28818-9

Ⅰ. ① P… Ⅱ. ①刘… ②王… ③李… Ⅲ. ①软件工具—程序设计—教材 Ⅳ. ① TP311.561

中国国家版本馆 CIP 数据核字（2023）第 102042 号

Python 程序设计项目化教程

Python CHENGXU SHEJI XIANGMUHUA JIAOCHENG

主　　编：刘君尧　王辉静　李坤颖

出版发行：上海交通大学出版社

邮政编码：200030

印　　制：北京荣玉印刷有限公司

开　　本：889 mm × 1194 mm　1/16

字　　数：576 千字

版　　次：2023 年 7 月第 1 版

书　　号：ISBN 978-7-313-28818-9

定　　价：59.80 元

地　　址：上海市番禺路 951 号

电　　话：6407 1208

经　　销：全国新华书店

印　　张：19

印　　次：2025 年 1 月第 3 次印刷

前　言

Python 语言问世于 1990 年。它自诞生之日起，就以“优雅”“明确”“简单”的设计哲学和良好的可阅读性吸引着来自世界各地的开发人员。Python 提倡的开源理念也为其流行奠定了坚实的群众基础。实际中，Python 已经广泛应用于计算机的各个领域，成为最流行、最具价值的编程语言之一。

2018 年，全国计算机等级考试将 Python 纳入考试科目。同年 9 月，全国计算机等级考试二级 Python 语言程序设计科目开考。这几年，考生踊跃报名，考试需求旺盛，很多院校纷纷开设 Python 课程。

本书参考最新的教学标准和课程改革成果，结合计算机等级考试要求和计算机相关专业的能力需求进行设计规划，以 12 个项目组织教学内容，将 Python 的各个知识点合理地分布在各个项目中。这 12 个项目分别是编程输出“社会主义核心价值观”、实现找零换算、跳水单人项目计分、显示杨辉三角、实现用户登录功能、实现通讯录功能、绘制正弦函数、创建客户信息簿、分词处理器、模拟智能机房管理、动态时钟和系统信息采集。

本书紧贴《全国计算机等级考试二级 Python 语言程序设计考试大纲（2022 年版）》（以下简称《二级考试》），涵盖《二级考试》全部内容，同步训练和巩固练习均采用《二级考试》题型，包括选择题、程序填空题和编程题，完全满足《二级考试》的需求。同时，根据计算机应用技术和运维等相关专业的需求，本书选用 time、datetime、platform、sys、os、shutil 和 psutil 等核心运维模块作为延伸学习内容，为专业后续运维课程的开展奠定坚实的编程基础。

本书由刘君尧、王辉静和李坤颖任主编，由冯海军、汪卫明、刘星明和桂荣枝任副主编。高维春提出了编写意见，在此表示感谢。

加快推进人才强国战略，健全现代职业教育体系，培养造就一大批具有高超技艺和精湛技能的高技能人才，是新时期国家对职业教育的新要求。本书落实校企合作双元育人的创新人才培养要求，面向工作岗位组织教材内容，全面提高学生的技能水平。同时，本书落实立德树人根本任务，贯彻《高等学校课程思政建设指导纲要》和党的二十大精神，将专业知识与思政教育有机结合，实现价值引领、知识传授和能力培养紧密结合。

本书提供了丰富的数字化教学资源，包括相关的课程标准、授课计划、教学课件、微课视频和习题解答等，有需要者可致电 13810412048 或发邮件至 2393867076@qq.com 领取。

本书可作为计算机、软件和通信等相关专业的教材，也可作为 Python 二级考试的培训用书。由于时间仓促及编者水平有限，书中存在的疏漏之处，敬请广大读者批评指正。

编　者

2023 年 1 月

目　录

项目 1　编程输出“社会主义核心价值观”

项目导航

知识目标

（1）了解 Python 的发展史及其特点。

（2）掌握 Python 编程的基本语法规范。

（3）初步了解 PEP 8 规范。

（4）了解程序的错误分类及其对应的处理方式。

（5）熟悉 Python IDLE 和 PyCharm 集成开发环境。

能力目标

（1）能在官网下载、安装 Python 和 PyCharm。

（2）能在 Python IDLE 中新建、编写、运行和调试程序。

（3）能在 PyCharm 中新建项目并新建、编写、运行和调试程序。

（4）能编写基础的 print( )语句。

素养目标

（1）深刻理解“社会主义核心价值观”的内涵和意义，践行“社会主义核心价值观”，做社会主义接班人。

（2）关注科技发展，深刻理解发展开源软件的战略意义，学好 Python 语言，努力为祖国建设贡献自己的力量。

（3）态度决定水平、细节决定成败，培养认真严谨的工作态度。

项目描述

“社会主义核心价值观”的基本内容是“富强、民主、文明、和谐、自由、平等、公正、法治、爱国、敬业、诚信、友善”。前面四个是国家层面的价值目标，中间四个是社会层面的价值取向，后面四个是公民个人层面的价值准则。

在 PyCharm 中建立项目 PythonStudy，并在该项目中编写程序 core. py，输出“社会主义核心价值观”。运行效果如图 1-1 所示。

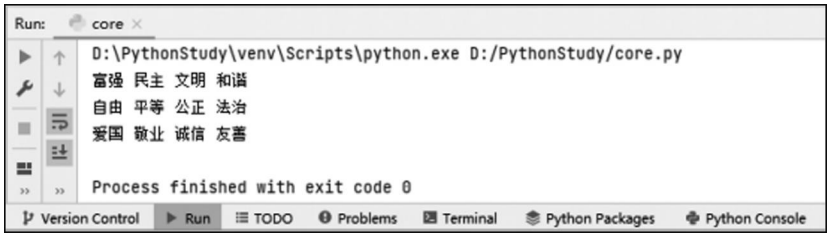


图 1-1　core. py 程序在 PyCharm 中的运行效果

项目分析

（1）项目要求使用 PyCharm 软件进行 Python 程序的开发，该软件的下载、安装和使用是项目开发的必备知识。

（2）编写 Python 程序要掌握 Python 语言的基本概念，信息的输出需要掌握 print( ) 函数的使用。

（3）编程过程中可能出现错误，而程序的调试是开发软件的关键步骤。

1.1　认识 Python

教学视频：开源和Python

1.1.1　开源和 Python

软件是新一代信息技术的灵魂，是数字经济发展的基础，是制造强国、网络强国、数字中国建设的关键支撑，对于加快建设现代产业体系具有重要意义。开放、平等、协作、共享的开源模式已成为全球软件产业创新的主导模式。“软件定义世界，开源定义软件。”

开源是软件赋能经济高质量发展的重要路径。全球 97% 的软件开发者和 99% 的企业均使用开源软件，开源软件在很大程度上提升了供给和需求两侧的技术产品创新能力，开源生态已成为软件产业国际竞争的战略制高点。开源助推软件业生产方式升级、生产关系变革，加速企业从“使用者”向“创造者”转变，并正在以群智模式应对数字化发展“长尾”需求，以应用为牵引赋能产业数字化转型。

作为开源语言，Python 目前由 Python 软件基金会管理，可以免费获取、使用和分发。

教学视频：Python 历史

1.1.2　Python 历史

Guido van Rossum 在 1990 年创建了一种新的编程语言，并命名为 Python。

作为一种解释型、面向对象、动态数据类型的高级程序设计语言，Python 兼具简洁性、易读性和可拓展性，已成为最受欢迎的程序设计语言之一。从 2004 年开始，Python 的使用率呈线性增长。2018—2022 年，IEEE Spectrum 发布的年度编程语言排行榜上，Python 位居第一名。

市面上 Python 的版本主要包括 Python 2 和 Python 3。Python 2 发布于 2000 年，已于 2020 年 1 月 1 日正式停止维护。Python 3 发布于 2008 年，是目前市场上的主流版本。Python 3 和 Python 2 不完全兼容。Python 2 程序无法在 Python 3 中正常运行。

1.1.3　Python 集成开发环境

教学视频：Python 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等工具，集成了代码编写功能、分析功能、编译功能和调试功能等一体化服务。所有具备这一特性的软件或软件套（组）都可以叫集成开发环境。Python 集成开发环境是指支持 Python 代码编写、调试和运行的集成开发环境。

Python IDLE 是一个 Python 自带的、简洁的集成开发环境，是 Python 标准发行版内置的简单的 IDE，从 Python 官网下载安装 Python，Python IDLE 就会自动安装好。

PyCharm 是由 JetBrains 打造的一款 Python IDE，是目前市场上应用最广泛的 Python 开发软件。它带有一整套可以帮助用户在使用 Python 语言开发时提高效率的工具，如调试、语法高亮、Project 管理、代码跳转、智能提示、自动完成、单元测试、版本控制等，并支持 Django、Flask 等多种流行的 Python Web 框架。

Eclipse 作为著名的自由集成开发环境，提供 Pydev 插件以支持 Python 编程。微软 Visual Studio 提供了 Python 相关的开发插件 PTVS。IPython、PythonWin、WingIDE、SPE、NINJA－IDE 等也都是不错的 Python IDE。

1.2　在 Python IDLE 中开发 Python 程序

2018 年 9 月，全国计算机等级考试科目中加入二级 Python 语言程序设计，考试环境要求为 Python 3.5.2 以上的 IDLE。下面以 Python 3.10.7 为例介绍 Python 软件的下载、安装和使用。

1.2.1　Python 的下载和安装

教学视频：Python 的下载和安装

进入 Python 官网下载页面 https://www.python.org/downloads/，查看最新版本的下载链接。本书选择 Python 3.10.7 版本（见图 1－2），读者也可选择当前最新版本。单击下载，获得 Python－3.10.7－amd64.exe 文件。

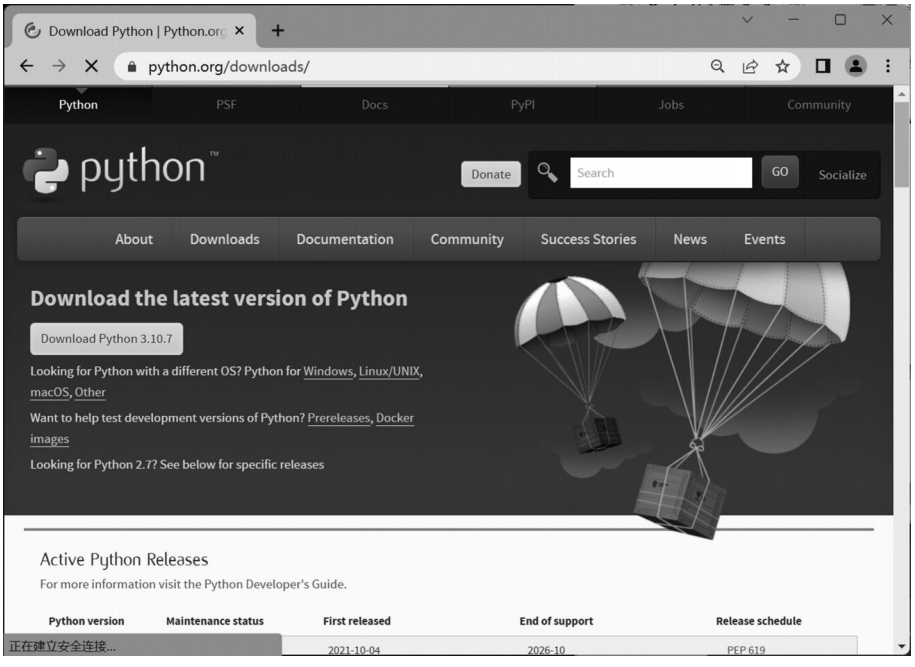


图 1－2　Python 官网下载页面

双击 Python－3.10.7－amd64.exe，进入 Python 安装界面，如图 1－3 所示。

图 1-3　Python 安装界面

勾选“Add Python 3.10 to PATH”，将 Python 3.10 解释器添加到 Windows 的环境变量 PATH 中。“Install Now”是使用默认配置安装。单击“Customize installation”进入自定义安装。在 Optional Features 页面不做修改，然后进入“Advanced Options”页面，如图 1-4 所示。

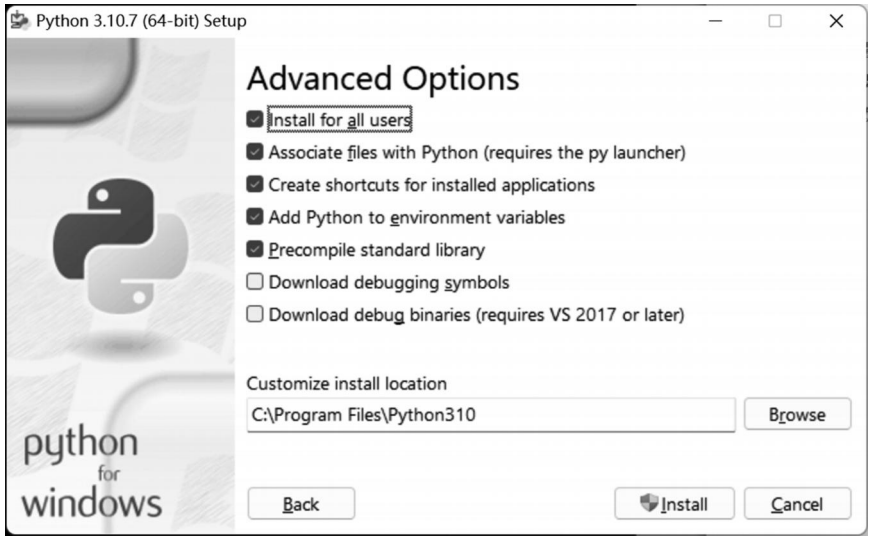


图 1-4　Python 安装之 Advanced Options 界面

如果勾选“Install for all users”，则安装路径就不再是当前用户的个人路径，而转为系统标准路径“C:\Program Files\Python310”。单击“Install”按钮，进入正式安装。

安装结束后会弹出“Setup was successful”信息框，即表示 Python 3.10.7 安装成功。

1.2.2　IDLE 的基本使用

安装 Python 后，Windows 开始菜单中新增了“Python 3.10”选项，共有四个子菜单。第一个子菜单“IDLE（Python 3.10 64-bit）”就是 Python 自带的 IDLE 集成开发环境。

IDLE 可以使用交互模式或文件模式来执行 Python 语句。

1. 交互模式

单击子菜单“IDLE（Python 3.10 64-bit）”，进入 Python Shell，也就是控制台。“>>>”是命令提示符，输入 Python 命令后立即显示运行结果，如图 1-5 所示。

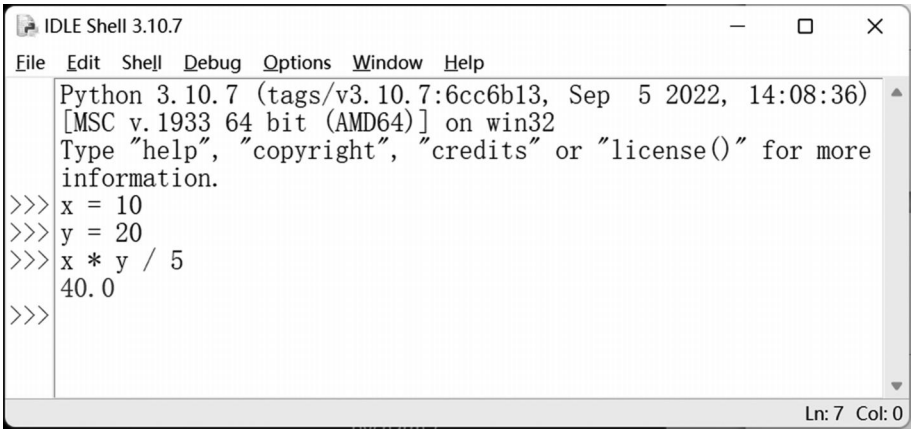


图 1-5　使用 Python IDLE 的交互模式

2. 文件模式

在 Python IDLE 中新建 Python 程序并进行编写和运行。单击菜单 “File” → “New File”，打开一个空白的编辑窗口，默认文件名为 “untitled.py”。

[注意] Python 程序的后缀名是 “.py”。

在空白编辑窗口中输入如图 1-6 所示的代码内容，单击菜单 “File” → “Save” 保存程序，这里保存在 “D:\Chapter1” 目录下，文件名为 “e_1.py”。

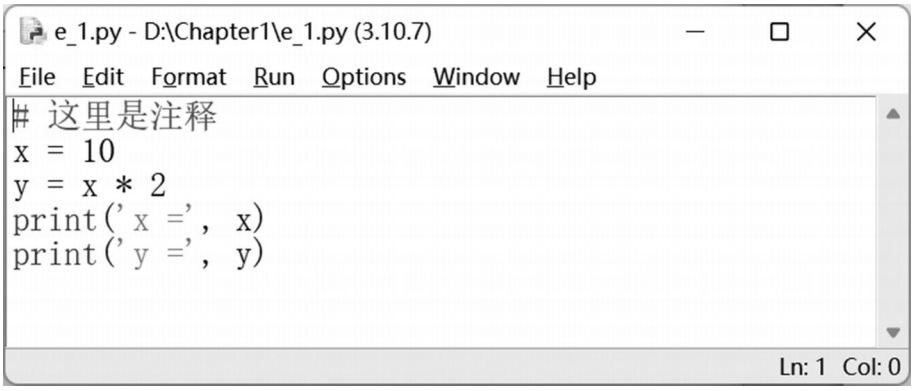


图 1-6　在 Python IDLE 中编写程序 e_1.py

单击菜单 “Run” → “Run Module F5”，或直接按 F5 键，均可运行程序。运行时，Python IDLE 会打开与当前程序关联的 Shell 窗口并显示运行效果，如图 1-7 所示。

图 1-7　程序 e_1.py 在 Python IDLE 中的运行效果

[提示] 交互模式下输入变量会显示其值，在文件中则需要使用 print( )语句来显示变量。

教学视频：IDLE的高级使用

1.2.3 IDLE 的高级使用

1. 语法高亮

IDLE 默认的配色方案如图 1-8 所示。单行注释是红色的，函数是紫色的，字符串是绿色的，变量是黑色的，保留字是橙色的。程序运行时其内部输出是蓝色的，用户交互输入是黑色的。

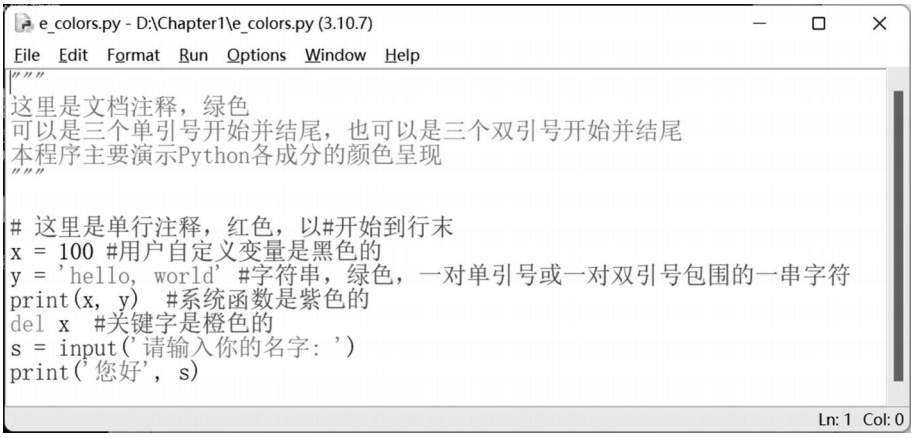


图 1-8 IDLE 默认的配色方案

2. 语法提示

IDLE 可以显示语法提示，如图 1-9 所示。

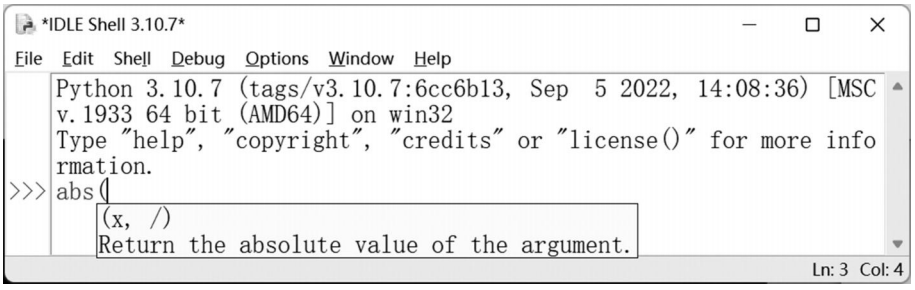


图 1-9 IDLE 的语法提示

教学视频：IDLE的程序调试

1.2.4 IDLE 的程序调试

程序调试是指在编写的程序实际运行前，用手工或编译程序等方法进行测试，修正程序错误的过程。这是保证软件项目正确性必不可少的步骤。

这里以 e_area.py（见图 1-10）为例讲解程序调试。该程序的功能是计算半径为 10 的圆的面积，并保留两位小数。当前代码存在错误。

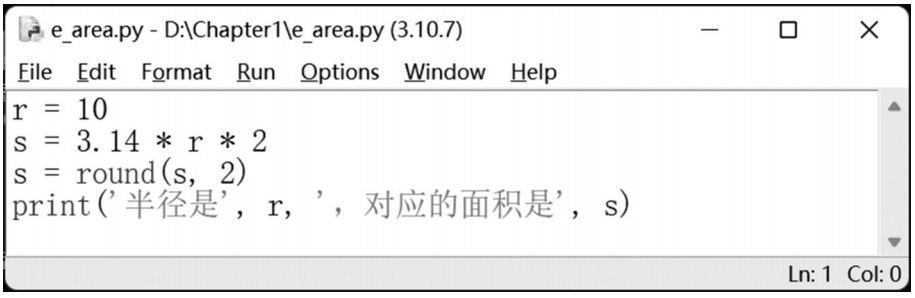


图 1-10 程序 e_area.py

1. 打开调试器

运行程序后将打开与之关联的 Shell 窗口，单击该窗口下的菜单 “Debug” → “Debugger”，弹出 Debugger 调试器窗口，如图 1-11 所示。

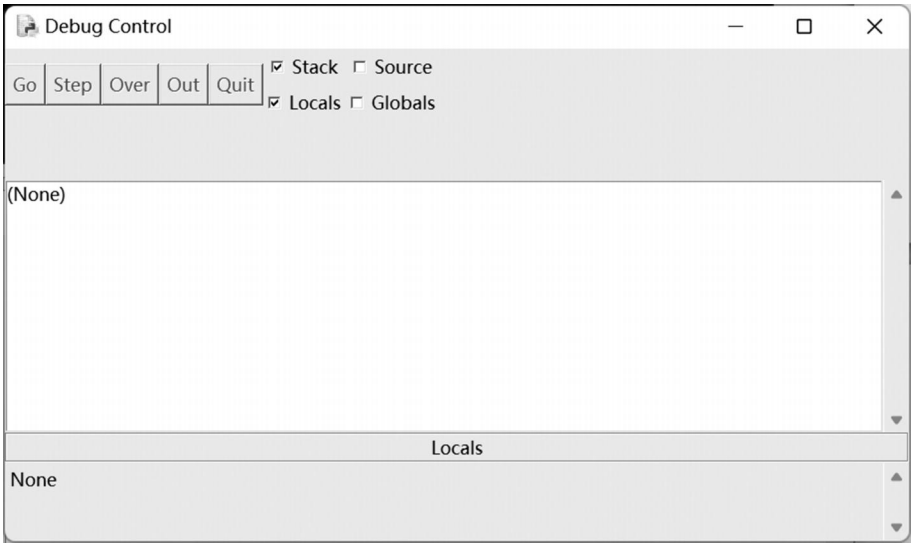


图 1-11　IDLE 的 Debugger 调试器窗口

Debugger 中 “Quit” 按钮的右边有四个复选框，具体含义如下。

(1) “Stack” 表示堆栈，用于跟踪代码执行信息，信息显示在白色窗口，默认显示 “(None)”。

(2) “Source” 表示在调试过程中即将运行的代码行，用灰色背景标注。

(3) “Locals” 表示局部空间，显示在 “Stack” 的下方。

(4) “Globals” 表示全局空间，显示在 “Locals” 的下方。

目前不涉及局部空间，所以暂时跳过 “Locals”，只勾选其他三个选项。

2. 设置断点

调试代码时可以设置断点 (Breakpoint)。断点的主要作用是控制程序运行到断点所在行时暂时挂起，以方便程序员进行观察分析。鼠标右击第 3 行，在快捷菜单中单击 “Set Breakpoint” 设置断点，此时第 3 行的背景色变成黄色，如图 1-12 所示。取消断点时需要在快捷菜单中单击 “Clear Breakpoint”。

图 1-12　在 e_area.py 中设置断点

3. 启动调试

单击菜单 “Run” → “Run Module” 进入程序调试。e_area.py 的第 1 行将呈现灰色背景，意思是即将执行该行代码。此时，调试器窗口也发生变化。如图 1-13 所示，Stack 区域给出即将执行的代码行信息，“>'__main__'.<module>()，line 1:r = 10” 表示即将执行第 1 行代码。Globals 全局空间出现几个与当前程序相关的全局参数，例如：“__file__” 是正在执行的程序名称。

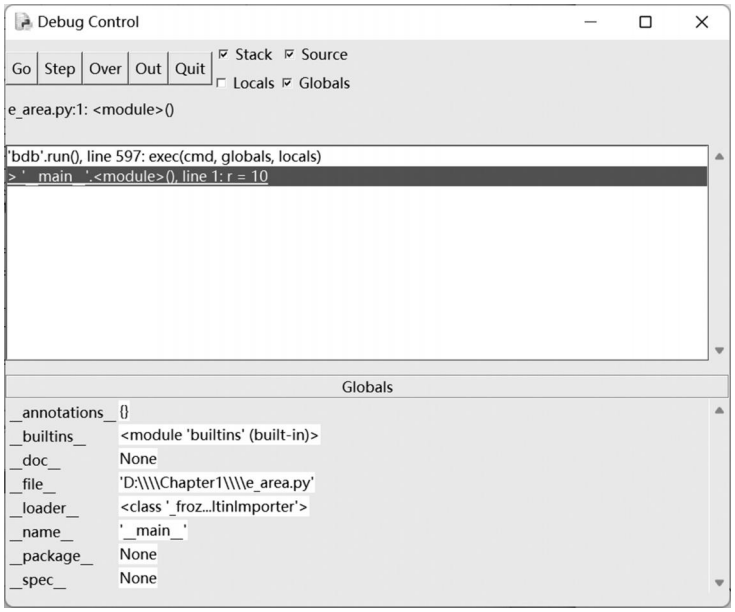


图 1-13　刚进入调试状态的 Debug Control

进入调试后，调试器左上角的五个按钮转为可使用状态，它们的作用如下。

（1）Go。以正常速度执行程序的剩余部分，直到结束或者遇到一个断点。

（2）Step。一次让程序执行一行代码，如果当前行是一个函数调用，则调试器会跳入函数内部。

（3）Over。一次让程序执行一行代码，如果当前行是一个函数调用，则调试器不会跳入函数内部，而是直接运行当前的整行代码并移动到下一行。

（4）Out。当调试已进入某一个函数内部时，Out 可以直接跑完整个函数的剩余部分并跳出这个函数，返回它被调用的代码处；当并未在函数内部时，Out 与 Go 作用相同。

（5）Quit。退出调试。

4. 继续调试

单击 Over 按钮，灰色背景下移到第 2 行，这意味着第 1 行执行完毕。观察调试器窗口的变化。Stack 区域显示即将执行“line 2:s = 3.14 * r * 2”。Globals 区域的变化是我们关注的重点，变量 r[①]被创建且值为 10。继续单击 Over 按钮，第 2 行被执行。第 2 行执行后的调试状态，如图 1-14 所示，此时全局空间中会新增变量 s。

①本教材代码中的注释和输入用斜体表示，为避免正斜体格式混乱，同时考虑到正文与代码格式的一致性，本教材涉及的变量等科技符号统一用正体表示。

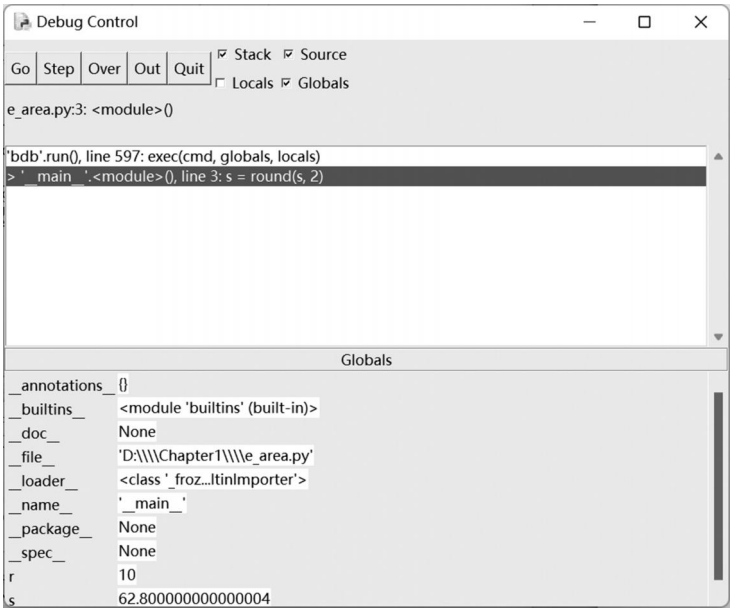


图 1-14　第 2 行执行后的调试状态

运行到这里就定位到错误了。变量 s 并不是预期的 314.0，显然是面积计算公式出错了。此时，若还要继续调试，单击 Over 继续单行执行，单击 Go 将直接运行到程序结束，单击 Quit 则直接退出调试。

退出调试后修改程序，将第 2 行代码调整为 “s = 3.14 * r * r”。再次运行程序，则输出 “半径是 10，对应的面积是 314.0”，运行正确。

[思考] 3.14 * 10 * 2 的计算结果为什么不是 62.8？“s = round(s,2)” 的作用是什么？

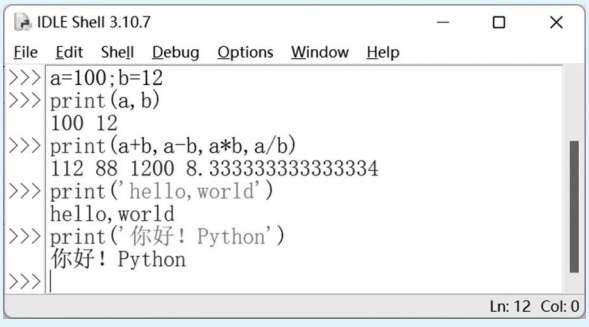
同步训练

（1）从 Python 官网下载和安装最新版 Python 软件。

（2）打开 Python IDLE，新建 e_area.py 并输入修改后的正确内容，调试并运行。

（3）打开 IDLE shell 窗口，录入如图 1-15 所示的交互命令并理解其内容。

教学视频：IDLE Shell的使用



教学视频：编写交互命令

图 1-15　Python IDLE 命令模式的使用练习

（4）打开 IDLE Shell 窗口，编写交互命令依次实现：创建变量 i、j 和 k，分别取值 3、4 和 5；显示 i + j + k 的值；修改 i 为 10；再次显示 i + j + k 的值。

1.3　在 PyCharm 中开发 Python 程序

教学视频：Pycharm的下载和安装

1.3.1　PyCharm 的下载和安装

进入 PyCharm 官网 https://www.jetbrains.com/pycharm/，单击 “Download” 进入 PyCharm 官网的下载页面，如图1-16 所示。PyCharm 发行版包括 Professional（专业

版）和 Community（社区版）。

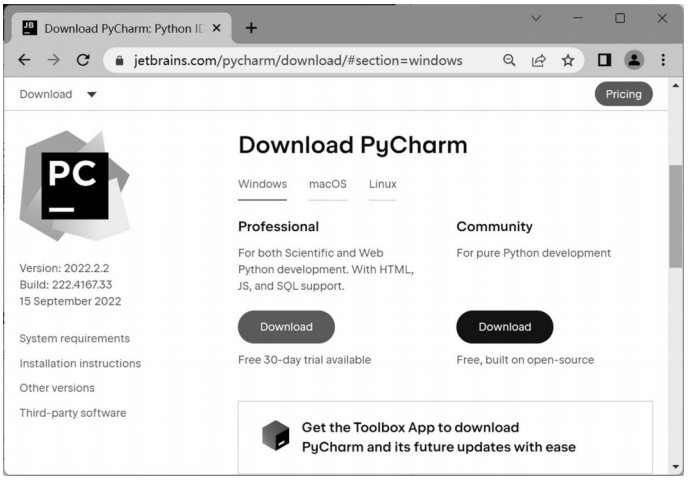


图 1-16　PyCharm 官网的下载页面

PyCharm 社区版是完全免费的，能够满足多数 Python 开发人员的需求；PyCharm 专业版更强大，增加了 Web 开发、Python Web 框架、Python 分析器、远程开发、数据库与 SQL 支持等功能。这里，单击社区版下的“Download”按钮，得到“pycharm-community-2022.2.2.exe”（后续可能会有版本更新，读者直接使用最新版本即可，操作是一样的）。

双击该文件，进入安装界面。“Choose Install Location”页面用于指定 PyCharm 的安装路径，通常不做修改。“Installation Options”页面用于可选项配置。“Create Desktop Shortcut”表示建立桌面快捷方式。“Create Association”表示在 Windows 系统注册.py 程序与 PyCharm 的关联。“Choose Start Menu Folder”页面用于选择 PyCharm 在开始菜单中出现的位置，一般不做修改。安装结束后，Windows 开始菜单里新增 JetBrains 选项，其子项 JetBrains PyCharm 即为 PyCharm 的启动菜单。

教学视频：Pycharm的基本使用

1.3.2　PyCharm 的基本使用

首次启动 PyCharm 时要求用户继续完成安装，选择“Do not import settings”，单击“OK”。紧接着进入 PyCharm 的启动界面，在“Customize”选项页的“Color theme”中可以设置主题颜色，如图 1-17 所示。其中“Darcula”以深灰色为底色，“IntelliJ Light”则以白色为底色。这里，我们选择“IntelliJ Light”。

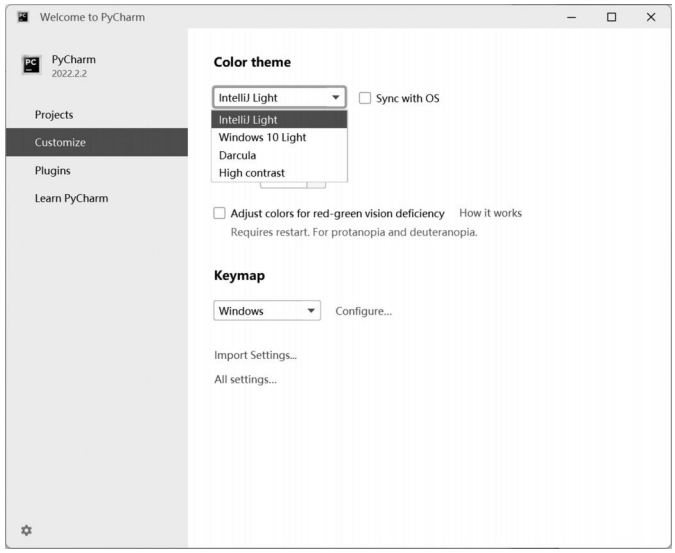


图 1-17　PyCharm Customize 选项页

在 “Projects” 选项页中选择 “New Project”，即新建项目。在 D 盘下新建 PythonStudy 项目，然后单击 “Create” 即可，如图 1-18 所示。

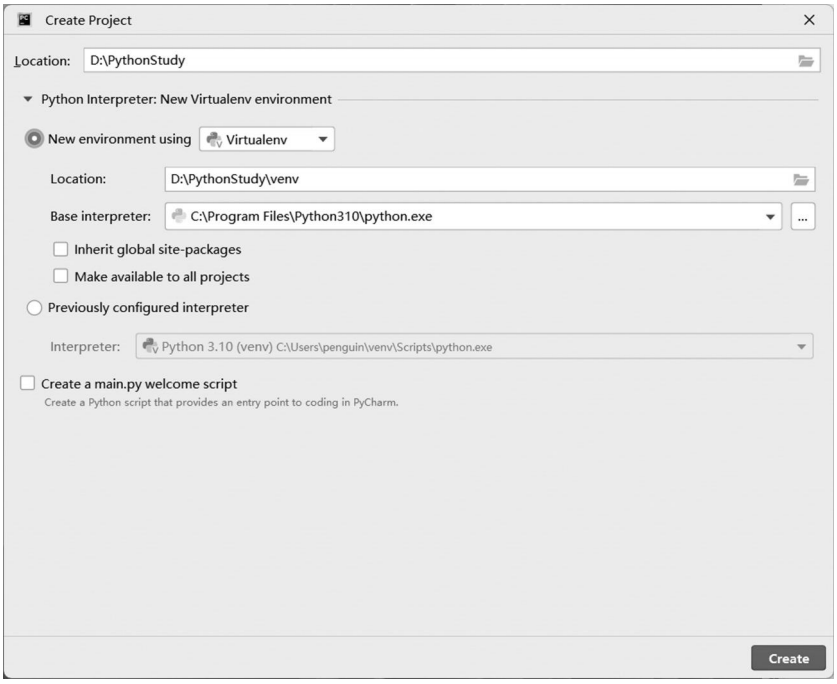


图 1-18　在 PyCharm 中新建项目 PythonStudy

现在正式进入 PyCharm 开发环境。虽然此时用户尚未开始写代码，但从左上方的 Project（项目）区域可以看到，PythonStudy 项目中已经有一些子目录和文件，这些内容可以帮助用户进行项目的整体管理，如图 1-19 所示。

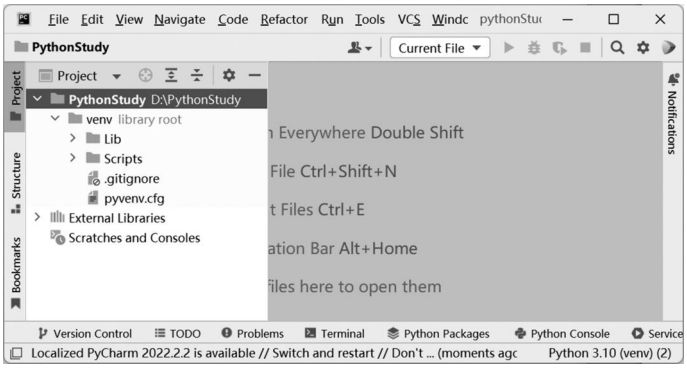


图 1-19　PythonStudy 项目中已有的子目录和文件

右击 “PythonStudy”，选择 “New” → “Python File”，输入文件名 “test”，即可在项目中新建程序 test.py。输入如图 1-20 所示的代码内容，右击程序 “test.py” → “Run test” 或使用快捷键 “Ctrl + Shift + F10” 运行程序，运行结果显示在窗口下方。

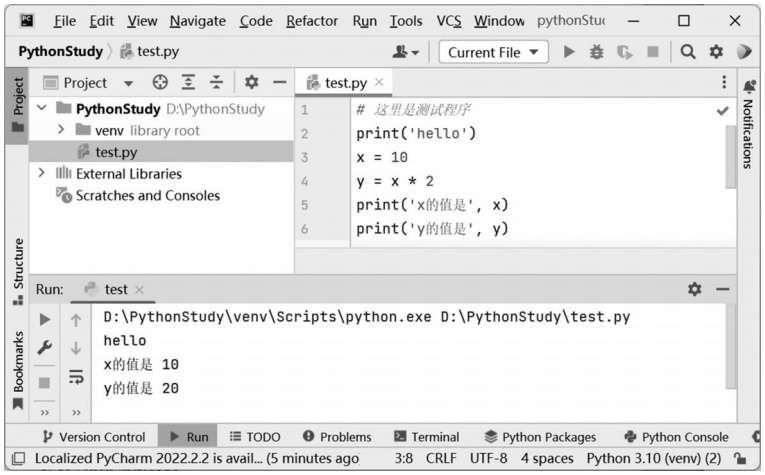


图 1-20　程序 test. py 的代码和运行结果

1.3.3　PyCharm 的程序调试

在 PyCharm 中单击程序左侧的行号和代码之间的灰色区域，会出现一个红点，同时当前行的背景改为粉色，这表示在该行设置断点，如图 1-21 所示。再次单击后，断点取消。

教学视频：Pycharm的程序调试

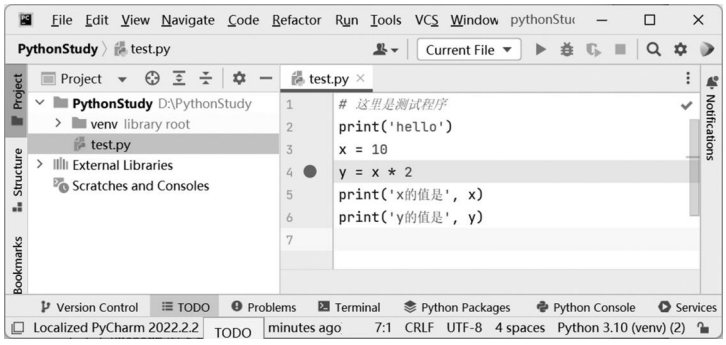


图 1-21　在 test. py 中设置断点

右击 test. py，选择“Debug test”进入调试状态。

IDLE 启动调试时会进入第 1 行并等待，而 PyCharm 将直接运行到第一个断点处，如果没有设置断点，则直接跑完整个程序。因为第 4 行已设断点，程序将运行完前 3 行，在第 4 行进行等待。观察软件下方的 Debugger 窗口，<module>堆栈（全局空间）生成变量 x，类型为 int，数值为 10，如图 1-22 所示。

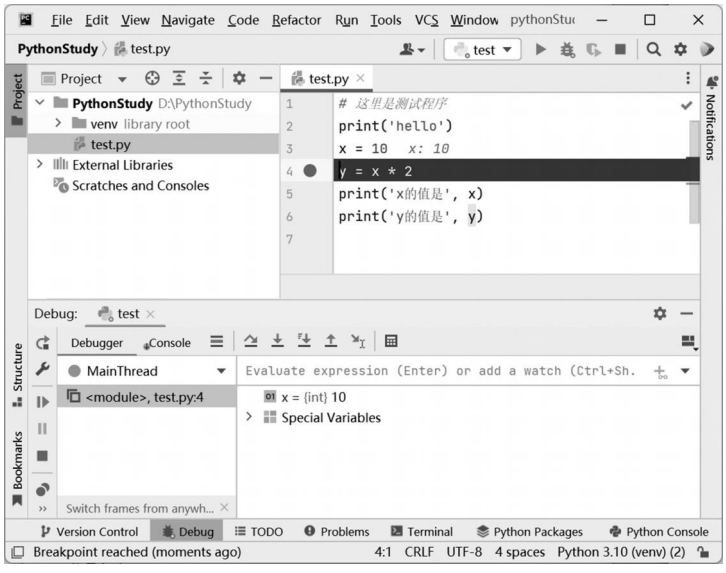


图 1-22　test. py 的调试界面

Debugger 窗口上的几个调试箭头提供不同的调试操作，也可以使用 Run 菜单下的选项，或者直接使用快捷键，PyCharm 的常用调试快捷键如表 1-1 所示。

表 1-1　PyCharm 的常用调试快捷键

快捷键	说　明
Ctrl + F8	在当前行设置或取消断点
F8	运行一条语句
F9	跳到下一个断点或程序结束处
F7	进入函数
Shift + F8	跳出函数

1.3.4　PyCharm 的常规配置

可以对 PyCharm 进行常规配置，单击菜单 “File” → “Settings”，进入配置界面。这里简要给出几个常用的配置信息。

“Appearance & Behavior” → “Appearance” 可以设置主题，“IntelliJ” 为白色背景，“Darcula” 为灰色背景，“High contrast” 是高对比度。“Editor” 是编辑器的设置，其 “Font” 子页面可以设置字体类型、字体大小和行间距等信息。

“Project：项目名” 下的 “Python Interpreter” 用于配置 Python 解释器。

在 PyCharm 中编写代码，代码下方经常会出现灰色波浪线。这并不是错误，而是不符合 PEP 8 编码规范。其配置参数在 “Editor” → “Inspections” 中，相应的 PEP 8 选项为 “PEP 8 coding style violation” 和 “PEP 8 naming convention violation”。建议程序员在学习初期就培养良好的编码习惯，尽量遵循 PEP 8 规范。右击 Python 项目或者 Python 程序并选择 “Reformat Code”，PyCharm 编辑器会对代码进行调整，使之尽可能符合 PEP 8 规范。

同步训练

（1）下载和安装 PyCharm 软件，并进行合理配置。

教学视频：test和p_print

（2）新建项目 “D:\Chapter1” 并新建 test.py，输入如图 1-20 所示内容，调试和运行程序。

（3）在项目 “D:\Chapter1” 中新建 p_print.py，显示如图 1-23 所示的效果。

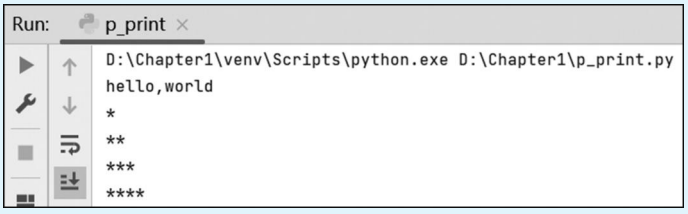


图 1-23　程序 p_print.py 的运行效果

（4）在项目 “D:\Chapter1” 中新建 p_product.py。创建三个变量 i、j 和 k，值分别是 3、4 和 5，显示这三个变量的值和 i * j * k 的值，然后修改 i 的值为 10，再次显示这三个变量的值和 i * j * k 的值。运行效果如图 1-24 所示。

同步训练

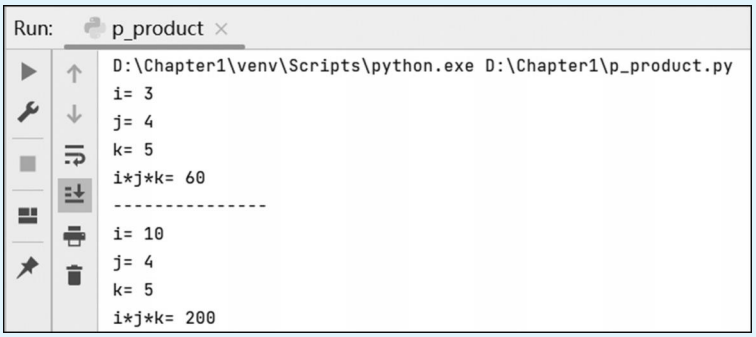


图 1-24　程序 p_product. py 的运行效果

1.4　Python 语法初识

教学视频：基本语法

1.4.1　基本语法

1. 区分大小写

Python 语言严格区分大小写，如图 1-25 所示。

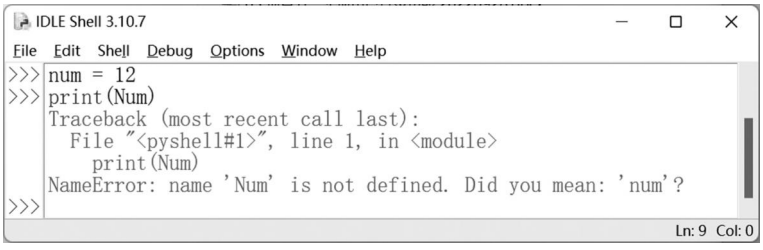


图 1-25　使用未定义的变量会报错 NameError

2. 逗号和分号

函数中若输入多个参数，参数间应用逗号分隔。

句末可以有分号也可以没有，但不建议有。通常每行一个语句，也可以使用分号“;”在同一行安排多个语句（不建议这么做）。参考代码如下。

```
1  >>> a = 1
2  >>> b = 2
3  >>> c = 3; d = 4; e = 5
4  >>> f, g, h = 6, 7, 8
5  >>> print(a, b, c, d, e, f, g)
6  1 2 3 4 5 6 7
7  >>> print('hello', a, b, c)
8  hello 1 2 3
```

[提示]“a = 值 1”“b = 值 2”“c = 值 3”可以联合写成“a, b, c = 值 1, 值 2, 值 3”。

3. 缩进

Python 以缩进来控制代码结构。缩进符包括空格和制表符（Tab 键）。行首不能随意出现空格或制表符。一级缩进通常是四个空格或者一个制表符，二级缩进则是八个空格或者两个制表符，以此类推。

PEP 8 规范建议使用四个空格表示一级缩进。不要混合使用制表符和空格，这在跨越不同平台的时候可能无法正常工作。行首多余的空格或制表符会导致报错，如图 1-26 所示。

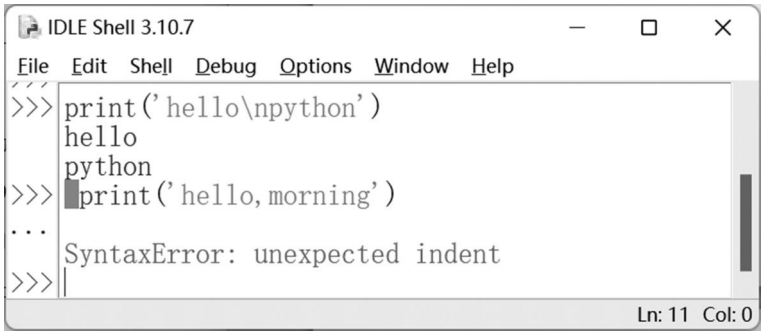


图 1-26　行首多余的空格或制表符导致的报错

[提示] 字符串中的'\n'是换行符，会产生换行效果。

[思考] PyCharm 的 Python Console 中在语句前输入多余的空格会报什么错误？为什么两个软件的报错信息有差别？

4. 语句块

代码行首的缩进层次决定语句的分组。同一层次的语句必须有相同的缩进。每一组这样紧连的语句称为语句块。图 1-27 中的第 8—10 行组成一个语句块，第 12 行则单句成块。

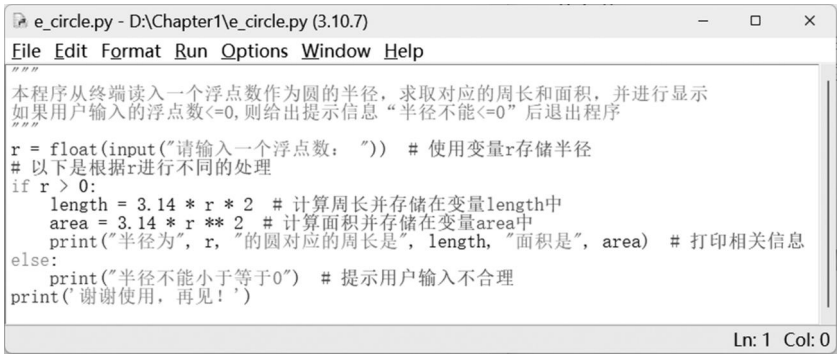


图 1-27　程序 e_circle.py

5. 帮助文档和注释

帮助文档用于告诉代码使用者或其他开发人员如何调用当前代码。帮助文档是写入代码中的，应用程序可以通过__doc__属性来访问帮助文档。

注释就是对代码的解释和说明，其目的是让人们能够更加轻松地了解代码。注释以#开头，不写入代码中。注释不是必须的，但是恰当的注释能帮人们更好地理解程序。

1.4.2　PEP 8 规范

Python 官网列出了 PEP 8 规范，其网址为 https://www.python.org/dev/peps/pep-0008/。该规范描述了 Python 编程风格的方方面面，目的在于让不同程序员编写的 Python 代码保持最大程度的相似风格，以便于阅读和交流。这里列举其中一些规范。

教学视频：PEP 8规范

(1) 每级缩进使用 4 个空格。Python 3 不允许混合使用制表符和空格进行缩进。

(2) 限制代码行的最大长度为 79 个字符。

(3) 不鼓励同一行里有多个语句。

(4) 用两个空行包围顶级函数和类定义。类的方法之间使用一个空行。

(5) 不要使用 l（小写的 L）、O（大写的 o）和 I（大写的 i）作为单字变量名。在某些字体中，这

些字很难与数字的 0 和 1 区分。

（6）类用 self 作为实例方法的第一个参数，用 cls 作为类方法的第一个参数。

1.4.3 Python 中的错误

程序在编写或者运行过程中可能出现语法错误、运行时错误和逻辑错误。

如果程序有语法错误，运行时会弹出 SyntaxError 窗口，提示“invalid syntax”，如图 1-28 所示。浅红色方块定位在错误处（定位不一定完全准确）。

教学视频：Python中的错误

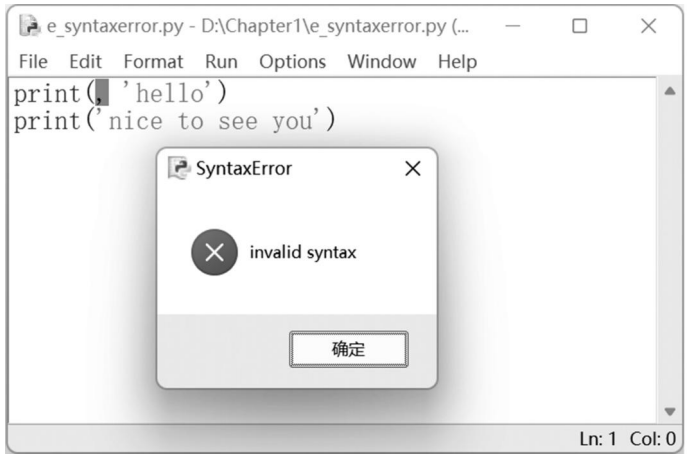


图 1-28 程序 e_syntaxerror. py 的语法报错

程序能运行，但是在运行的过程中出现错误并中断退出，表明程序存在运行时错误。Shell 窗口会给出具体的错误信息。要学会查看报错信息，从中找出错误原因并正确修改代码。如图 1-29 所示，程序 e_runerror. py 报错退出，从提示信息可以看出，错误出现在代码的“line 2”，具体内容是“print(x)”，错误提示是“NameError：name 'x' is not defined”。

［提示］变量要在创建之后才能使用。

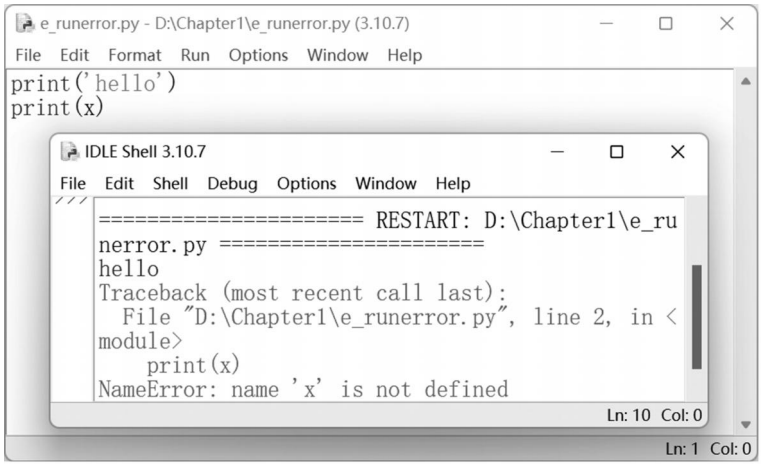


图 1-29 程序 e_runerror. py 运行时报错

程序可以运行，但是运行结果并不是预期的结果，表明程序存在逻辑错误，这通常是设计考虑不周或者编码疏忽导致的。如果不能快速定位错误并修正，最好的解决办法就是通过调试观察程序运行过程中数据的变化，以定位并修正错误。

项目实施

打开 PyCharm，新建项目 PythonStudy，新建并编写程序。

实施方案 1　core. py

```
1  print('富强 民主 文明 和谐')
2  print('自由 平等 公正 法治')
3  print('爱国 敬业 诚信 友善')
```

实施方案 2　core_v2. py

```
1  print('富强', '民主', '文明', '和谐')
2  print('自由', '平等', '公正', '法治')
3  print('爱国', '敬业', '诚信', '友善')
```

实施方案 3　core_v3. py

```
1  print('富强 民主 文明 和谐\n 自由 平等 公正 法治\n 爱国 敬业 诚信 友善')
```

实施方案 4　core_v4. py

```
1  print('富强 民主 文明 和谐', '自由 平等 公正 法治','爱国 敬业 诚信 友善', sep = '\n')
```

项目总结

（1）本项目的重点在于掌握 PyCharm 软件的使用方法。新建项目并新建程序，以及编写、运行和调试程序，是必须掌握的内容。

（2）项目实施给出了四个解答版本。对于一个程序功能，往往有多种解答方式。编程时不要局限于某一个思路，要学会多思考、广思考、深思考。

（3）“社会主义核心价值观”中的“敬业”是人们基于对一件事情、一种职业的热爱而产生的一种全身心投入的精神，是社会对人们工作态度的一种道德要求。作为学生的我们，也要努力做到爱岗敬业，树立主人翁责任感、事业心，追求崇高的职业理想，培养认真踏实、恪尽职守、精益求精的工作态度，力求干一行、爱一行、专一行，努力成为本行业的行家里手。Python 语言作为目前最流行的编程语言之一，其开源性适应国情的需要，因此成为我们学习编程语言的首选。初期接触 Python 语言较为简单，容易上手。但是，随着学习的深入，难度会日益增大。我们要有克服困难、勇于挑战、精益求精的学习态度，努力学好这门语言。

（4）本项目存在文字的色彩和样式过于单一的问题，后续的课程中我们将学习如何在画布上使用指定颜色和字体，并在合适位置进行文字的输出。

巩固练习

一、 单选题

1. 以下关于 Python 的描述，哪个是错误的？（　　）

A. Python 语言是一门支持面向对象的高级语言

B. Python 源代码的后缀名是. pyc

C. Python 语言提供丰富的扩展库

D. Python 3 向下不兼容，Python 2 程序不能直接在 Python 3 中运行

2. 以下关于 Python 的描述，哪个是错误的？（　　）

A. Python 语句的行末不允许出现分号，否则报语法错误

B. Python 语言是严格区分大小写的

C. 调用 Python 函数时，多个参数之间用逗号分隔

D. Python 变量必须在创建之后方可使用

3. Python 使用什么方式划分语句块？（　　）

A. 冒号　　B. 一对大括号　　C. 一对小括号　　D. 缩进对齐

4. Python 中用什么符号表示单行注释及行末注释？（　　）

A. $　　B. !　　C. #　　D. &

5. Python 中同一行有多个语句，语句间用什么符号分隔？（　　）

A. :　　B. ,　　C. ;　　D. .

6. 通常 Python 中的一条语句占一行，语句过长超过编辑器的水平长度时会导致不方便阅读，此时可使用续行符将一条语句写在多行之中。下列哪个符号是续行符？（　　）

A. /　　B. \　　C. …　　D. ...

7. 以下关于 Python 注释的描述，哪个是错误的？（　　）

A. 程序头部的注释通常可标明作者和版权信息，程序中也应提供必要的注释

B. 程序运行时注释将被解释器过滤掉，不会跟随代码执行

C. 注释在程序运行时是无用的，所以程序中尽量不要有注释，避免占用资源

D. Python 帮助文档以三个单引号或三个双引号开头并结尾

8. Python IDLE 调试器中，下列哪个选项表示全局空间？（　　）

A. Stack　　B. Source　　C. Locals　　D. Globals

9. Python IDLE 调试程序时，希望执行一行代码，如果当前行是一个函数调用，则进入函数内部，应该单击哪个按钮？（　　）

A. Go　　B. Step　　C. Over　　D. Out

10. 如果 Python 程序的行首出现多余的空格或者制表符，会报什么错误？（　　）

A. SyntaxError　　B. NameError　　C. IndexError　　D. ValueError

二、 编程题

1. 在项目“D:\Chapter1”中新建 h_starpyramid.py，实现如下效果。

```
      *
     * * *
    * * * * *
  * * * * * * *
* * * * * * * * *
```

2. 在项目“D:\Chapter1”中新建 h_product.py，实现如下效果。从终端输入三个值，计算并显示它们的乘积。提示：“i = int(input('请输入一个整数：'))”将给出“请输入一个整数：”的提示信息，接收用户输入的一行字符并将其转为整数后传递给 i。

```
请输入整数 i 的值：10
请输入整数 j 的值：5
请输入整数 k 的值：3
乘积是 150
```

3. 在项目 “D:\Chapter1” 中新建 h_poem.py，输出毛泽东的诗词《沁园春·雪》。

4. 在项目 “D:\Chapter1” 中新建 h_scoretolevel.py，输入如下代码，运行并分析程序功能。

```
score = int(input('请输入成绩：'))
if score < 0 or score  >  100:
        print('无效分数')
elif score < 60:
        print('不及格')
else:
        print('及格')
```

5. 在项目 “D:\Chapter1” 中新建 h_pyramid.py，输入如下代码，运行并分析程序功能。

```
n = int(input('请输入金字塔行数　'))
for i in range(1, n + 1):
        for j in range(n − i):
                print(' ', end = '')
        for k in range(2 * i  −  1):
                print('*', end = '')
        print()
```

项目 2 实现找零换算

项目导航

知识目标

（1）掌握保留字的概念和作用。

（2）掌握标识符的命名规则。

（3）理解变量的基本概念。

（4）掌握常用的数值数据类型和进制间的转换方法。

（5）掌握转义字符的使用方法和字符串的基本使用方法。

（6）理解数据转换的基本原理。

（7）掌握各类基本运算符的使用方法和它们的优先级。

（8）掌握常见的内置函数。

能力目标

（1）能正确合理地命名变量和其他元素。

（2）能进行整数进制间的转换。

（3）能合理使用数据间的自动类型转换和强制类型转换。

（4）能使用基本数据类型和运算符进行编程。

（5）能正确使用常见的内置函数进行编程。

素养目标

（1）培养严谨规范、精益求精的职业素养和工匠精神。

（2）加强知识转化能力，能从理论联系实际，从工作生活中发现切入点，用编程改善生活品质，建设智慧家园。

项目描述

收银员在进行购物结算的时候经常会涉及找零的问题，人流量较大时容易出现计算错误。本项目旨在提供找零方案，输入用户要找零钱的数额（0～99.9，单位：元）。输出找零方式（包括 10 元、5 元、1 元、5 角和 1 角），要求：钞票数量尽可能少。

```
请输入找零金额：28.6
找回金额：71.4
10 元钞票数量是 7
5 元钞票数量是 0
1 元钞票数量是 1
5 角钞票数量是 0
1 角钞票数量是 4
```

```
请输入找零金额：38.7
找回金额：61.3
10 元钞票数量是 6
5 元钞票数量是 0
1 元钞票数量是 1
5 角钞票数量是 0
1 角钞票数量是 3
```

项目分析

（1）本项目涉及多个信息的存储，应合理地命名变量并给出相应的注释信息。

（2）找零需要进行数据间的计算，包括浮点数和整数间的转换和数学运算。

（3）数据在运算过程中可能会出现极小的误差，要合理地选择截断或者四舍五入，才能保证最后数据的准确性。

2.1　Python 中的基本概念

2.1.1　保留字

教学视频：保留字

Python 语言包含一些具有特殊含义的单词，称为保留字或者关键字（keyword）。我们将在后续的学习中逐步了解并掌握这些保留字的使用方法。

```
>>> help()  # 进入 help 环境
Welcome to Python 3.10's help utility! ……
help> keywords  # 在 help 中查询所有保留字
Here is a list of the Python keywords.  Enter any keyword to get more help.

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

help>q  # 退出 help
```

2.1.2　标识符

教学视频：标识符

标识符（identifier）是用来标识某个实体的一个符号，在不同的应用环境下有不同的含义。在计算机编程语言中，标识符是用户编程时使用的名字，用于给变量、常量、函数、类、模块和包等命名，以建立名称与使用之间的关系。

1. 标识符的命名规则

对于标识符的命名规则，不同的编程语言有不同的要求。Python 语言的要求如下。

（1）以字母或下划线开头，并且只能由字母、数字或下划线组成。

（2）不能使用 Python 保留字。

（3）注意区分大小写。Num 和 num 表示不同的变量。

（4）不建议使用系统内置的模块、类型或函数，以及已导入的模块及其成员的名称作为变量名，这可能会改变其原来的类型和含义，如 abs、print 等。

（5）以下划线开头和结尾的标识符是有特殊意义的。我们将在面向对象的相关项目中学习它们的具体含义。

```
>>> year = 2018
>>> next_year = 2019
>>> 小狗 = 'pluto'
>>> cir-area = 314
SyntaxError: can't assign to operator
>>> a $name = 'betty'
SyntaxError:  invalid syntax
```

［思考］第 3 行的变量名为小狗，说明了什么？

```
>>> print
< built-in function print >
>>> print(2)
2
>>> print = 3
>>> print('abc')
Traceback (most recent call last):
    File "<pyshell#106>", line 1, in <module>
        print('abc')
TypeError:'int' object is not callable
>>>
================== RESTART:Shell ===============
>>> print('abc')
abc
```

［思考］为什么第 6 行报错 TypeError，第 13 行又能正常运行？

2. 标识符的命名约定

约定和规则不同，约定不是强制要求的，但是建议这样做。

命名变量或函数等时应尽量做到“见名知意”，如 speed、average。

复杂名称通常由多个单词组合而成，单词可以全小写，也可以除首字母大写外其余小写，还可以使用下划线连接，如 isinstance、fullName、api_version。命名应尽量符合 PEP 8 规范并保持项目代码的整体风格统一。

2.1.3 变量

1. 基本概念

变量来源于数学。例如，“r = 3.0”就是创建变量 r，其值为 3.0。引入到计算机语言中，变量就是在计算机中存储的某个数据信息，该数据信息通过变量名进行访问或者参与运算。变量的命名也要符合标识符的命名规则。

```
>>> gravity = 9.8   # 重力加速度
>>> fallingtime = 5   # 物体的下落时间
>>> distance = 1/2 * gravity * fallingtime * fallingtime   # 计算物体的下落距离
>>> distance
122.5
```

［技巧］编程时善用 Tab 键和“Alt +/”组合键进行变量名的自动补全。

上面代码中定义了一些数据，那这些数据的物理单位是什么呢？实际上，在电脑系统中存储的都是数据，单位的确认及换算是需要程序员自行考虑的。

例如，已知某高铁的速度为 300 千米每小时，它在铁轨上运行了 30 分钟，请问它行驶了多远？这里的千米每小时和分钟是不能直接运算的，需要程序员自行处理。

```
>>> speed = 300   # 根据题意，speed 的单位是千米每小时
>>> runningtime = 30 / 60   # 根据题意，runningtime 的单位是小时
>>> distance = speed * runningtime   # 那么，distance 的单位应该是千米
>>> distance
150.0
```

2. 变量的使用注意事项

变量的使用需要注意以下几点。

（1）Python 变量不需要事先声明数据类型，直接赋值即可创建。Python 解释器将计算等号右边的表达式从而得到相应的结果，并根据其数据类型为等号左边的变量分配存储空间。等号的左边必须是变量。

（2）函数 type(object)会返回 object 的数据类型。

（3）函数 isinstance(object,classinfo)用于测试 object 是否属于 classinfo 数据类型，是则返回 True，否则返回 False。

（4）变量的类型是可以改变的。

（5）使用 del 命令可以删除变量。del 的具体语法是 del 变量名。

```
>>> a = 20 * 5   # 创建 a，类型为 int，值为 100
>>> print(a, type(a), id(a))   # 显示 a 的值、类型、地址
100 <class 'int'> 1533230845264
>>> print(isinstance(a, int), isinstance(a, float))
True False
>>> a = 'Python'   # 这里 a 不再是整数，类型变为 str，值为'Python'
>>> print(a, type(a), id(a))
Python <class 'str'>  1533232155760
>>> del a
```

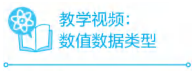
［思考］紧接 del a 后执行 print(a)，会产生什么运行结果？

［注意］与变量对应的概念是常量，通常指那些定义后不能被修改的量。C/C ++ 通过 const 保留字将标识符定义为常量，不允许修改其值。Python 中没有专门定义常量的方式，通常约定全大写名称表示常量，但仅仅是一种提示效果。

2.2　常见数据类型

Python 中一切皆为对象。数据类型都是类对象，简称类。根据类创建出来的对象，我们称之为实例

对象，简称实例。



2.2.1 数值型数据

1. 数值数据类型的分类

Python 数值数据类型包括 bool（布尔类）、int（整数类）、float（浮点数类）和 complex（复数类）。其中，bool 只包含 False 和 True 两个值，计算机内部用 0 来表示 False，用 1 来表示 True。

```
>>> a = False; b = True
>>> c, d, e = 100, 12.5, 23 + 4j
>>> print(type(a), type(c), type(d), type(e))
< class 'bool' > < class 'int' > < class 'float' > < class 'complex' >
>>> print(a + c, b + c)
100 101
```

关于复数 complex，需要注意以下几点。

（1）复数的虚部支持“j”和“J”，它的实部和虚部都是 float 型。

（2）x. real 和 x. imag 可分别获得复数 x 的实部和虚部。

（3）复数可以进行加减乘除，但是不可以比大小。

（4）内置函数 abs(x)返回 x 的绝对值，如果 x 是复数，则返回其数学意义上的模。

范例 2.1 e_complex. py

```
x = 6 + 8j
y = 3.0 + 4.0J
print(x.real, x.imag, abs(x))
print(x + y, x - y, x * y, x / y)
print(x > y)
```

［思考］该程序的运行结果是什么？

2. 科学计数法

科学计数法是指用形如 $a * 10^b$ 的表达式来描述浮点数。Python 用 e 或 E 表示 10 的次方。

```
>>> x = 1.5e3; y = 3.5e-5; z = 0.000034
>>> print(x, y, z)
1500.0 3.5e-05 3.4e-05
```

3. 进制间转换

Python 支持二进制、八进制、十进制和十六进制。当数字以 0b/0B、0o/0O、0x/0X 开头时，分别表示二进制、八进制和十六进制。

可以使用不同进制来表示同一数据，也可以进行不同进制数之间的切换。

int(a)将括号里的数据强制转换成整数。若采用 int(a,b)形式，则 b 指定 a 的进制类型。

如果不输入进制，可以使用 eval( )函数将各进制字符串转为整数。

```
>>> a = 0b1010; b = 0o12; c = 10; d = 0xa   # a、b、c、d 分别是整数 10 的不同进制表示形式
>>> print(a, b, c, d)   # 可以看到 print()时都是显示十进制的 10
10 10 10 10
>>> bin(100); oct(100); hex(100)   # 分别返回 100 的二进制、八进制和十六进制的字符串形式
'0b1100100'
'0o144'
```

```
'0x64'
>>> print(0b1101001, 0o127, 0x6fb)
105 87 1787
>>> print(int('0b1100110', 2), int('0o146', 8), int('0x66', 16), int('123', 7))   # 由字符串获得对应的整数
102 102 102 66
>>> s = '0b1100'; print(int(s, 2), eval(s))
12 12
>>> int('0b102', 2)
Traceback (most recent call last):
    File "<pyshell#12>", line 1, in <module>
        int('0b102', 2)
ValueError: invalid literal for int() with base 2: '0b102'
```

［思考］（1）int('123',7)的值为什么是 66？

（2）int('0b102',2)为什么报错 ValueError？

2.2.2　字符串

1. 字符编码

ASCII 码是最早的字符集，出现于 20 世纪 60 年代。ASCII 码建立了 128 个基本字符（包括 32 个不能打印出来的控制符号）与数字之间的对应关系，如空格是 32、字母 A 是 65。这些符号只占用一个字节的后 7 位，最前 1 位统一规定为 0。

随着计算机的发展，各国语言均被引入系统中。此时，128 个字符远远不够用，多种字符集应运而出，如 GB2312、BIG5、GB18030 和 Unicode 等。

Unicode 字符集扩展自 ASCII，用两个字节来表示常用的各语言文字，它为每种语言的各个常用字符设定统一并且唯一的二进制编码，以满足跨语言、跨平台的文本转换和文本处理的要求。

2. 字符串

Python 支持 Unicode 编码，并定义一对单引号或双引号包围起来的一串字符为字符串，数据类型是 str。当使用一对单引号来表示字符串时，字符串内允许出现双引号，反之亦然。

```
>>> s1 = "Tom'book"; s2 = '他说"你好"'
>>> s1; s2
"Tom'book"
'他说"你好"'
>>> print(s1, s2)
Tom'book 他说"你好"
```

［思考］控制台直接显示字符串和使用 print()显示字符串有什么差别？

Python 没有为单个字符设计对应的类，单个字符被视为长度为 1 的字符串。

内置函数 len(s)将得到字符串实例 s 的长度，即 s 中字符的个数。

```
>>> s1 = 'Python'; s2 = 'Java'; s3 = 'h'
>>> print(len(s1), len(s2), len(s3))
6 4 1
```

3. 字符和编码的转换

chr(a)和 ord(a)实现单字符和对应编码值的相互转换。

```
>>> print(ord('a'), ord('A'))
97 65
>>> print(chr(98), ord('学'), chr(23450))
b 23398 定
>>> 'x' < 'T'
False
```

字符比大小时，实际上比较的是它们在 Unicode 字符集中的编号。

［提示］'0' < '1' < … < '9' < … < 'A' < 'B' < … < 'Z' < … < 'a' < 'b' < … < 'z'。

4. 转义字符

转义字符“反斜杠 + 字母”被用来描述部分特殊字符。常见的转义字符如表 2-1 所示。

表 2-1　常见的转义字符

转义字符	意义	ASCII 码值（十进制）
\0	空字符（NUL）	000
\t	水平制表符（Tab 键）	009
\n	换行符	010
\"	双引号	034
\'	单引号	039
\\	反斜线字符	092
\ddd	三位八进制数所代表的任意字符	二位八进制
\xhh	二位十六进制所代表的任意字符	二位十六进制

［注意］计算字符串长度时，转义字符组合只能算长度 1。

```
>>> s1, s2, s3, s4 = '他说："您好！"',  'Tom\'s book',  'ab\137cd',  'ab\x6dcd'
>>> print(s1, s2, s3, s4, sep = '\n')  # sep = '\n'使得每打印一个变量就换行
他说："您好！"
Tom's book
ab_cd
abmcd
>>> print(len(s1), len(s2), len(s3), len(s4))
8 10 5 5
```

后续还会再深入学习字符串类，目前仅简单介绍字符串。

同步训练

教学视频：标识符、变量名和收益计算

（1）下面哪些名称可以作为 Python 标识符？

①passwod　②_num　③_　④cost $ PerDay

⑤id Card　⑥total- Num　⑦true　⑧None

⑨in　⑩print　⑪int　⑫not

（2）下面哪些名称作为变量名更合理？

①表示速度：a 还是 speed？

②表示自定义打印函数：print 还是 myprint？

③表示汽车的数量：x 还是 NumOfCars？

同步训练

(3)编写程序 p_profit.py,实现收益计算。

①创建收入变量 revenue 并赋值 98456;

②创建支出变量 costs 并赋值 45000;

③创建收益变量 profit 并赋值 revenue-costs;

④输出这三项数据(数字是通过变量输出的)。

```
收入是 98456
支出是 45000
收益是 53456
```

2.3　运算符

Python 运算类似于数学运算，它支持算术运算、赋值运算、关系运算、测试运算、逻辑运算和位运算等。

2.3.1　算术运算

Python 算术运算包含但不仅限于加减乘除，其算术符运算符如表 2-2 所示。

表 2-2　算术运算符

算术运算符	功能描述	案例
+	相加运算	1 + 2 得 3
-	相减运算	100 - 1 得 99
*	乘法运算	2 * 2 得 4
/	除法运算	4/2 得 2
//	整除运算，即返回商的整数部分	9//2 得 4
%	求余运算	10 % 3 得 1
**	幂运算。x ** y 返回 x 的 y 次幂	2 ** 3 得 8

新增的几个算术运算符，像加减乘除一样使用即可。

```
>>> a = 100; b = 3.5; c = 3 + 4j
>>> print(a + b * c, a % b, a // b, 28.0 * b + 2.0, sep = '    ')
(110.5 + 14j)    2.0    28.0    100.0
>>> a / 0
Traceback (most recent call last):
    File "<pyshell#19>", line 1, in <module>
        a / 0
ZeroDivisionError: division by zero
```

[提示] 除数为 0 时，/、%、//运算都会产生 ZeroDivisionError。

+ 和% 根据其应用的不同场景有不同的含义。这里只对 + 做解释,%将在字符串的格式化中再深入讲解。

```
>>> a = 100 + 5; s = "hello!" + "Python"   # 数字相加；字符串的合并
>>> print(a, s)
105 hello!Python
>>> a + s   # 数字和字符串不能直接相加
Traceback (most recent call last):
    File " <pyshell#5 > ", line 1, in <module >
        a + s
TypeError: unsupported operand type(s) for + : 'int' and 'str'
```

不难看出，+ 在 Python 中有如下含义。

（1）+ 的两个操作数均为数字时，执行常见的加法操作，将两个数字相加。

（2）+ 的两个操作数是字符串、列表或元组时，执行合并操作，也就是将前后内容连起来，形成一个新的字符串、列表或元组。

（3）当 + 的左右操作数无法被 + 时，将报错 TypeError。

教学视频：赋值运算

2.3.2 赋值运算

赋值运算符包括我们常用的 = 和扩展赋值运算符，如表 2-3 所示。

表 2-3 赋值运算符

赋值运算符	功能描述	案例
=	直接赋值	x = 3 表示将 3 赋值给变量 x
+=	加法扩展赋值	x += 3 等价于 x = x + 3
-=	减法扩展赋值	x -= 3 等价于 x = x - 3
*=	乘法扩展赋值	x *= 3 等价于 x = x * 3
/=	除法扩展赋值	x /= 3 等价于 x = x / 3
%=	求余扩展赋值	x %= 3 等价于 x = x% 3
**=	幂扩展赋值	x **= 3 等价于 x = x ** 3
//=	整除扩展赋值	x //= 3 等价于 x = x// 3

赋值运算符的左边必须为变量，不能是数字等常量，否则会报错 SyntaxError。运算时，先计算右边表达式的值，再将该结果赋值给左边的变量。

```
>>> x = 10
>>> x += 200; x
210
>>> x /= 3 + 3; x
35.0
>>> x **= 5 - 2; x
42875.0
```

同步训练

教学视频：口算验证和 p_ballmotion

（1）口算以下题目，并在计算机上验证。

①7 ** 2　②14% 4　③13.5% 2.5　④13.5// 2.5

⑤$\frac{1}{2^3+4}$　⑥$(1+2*2)^4$　⑦$100*(1+2.5\%)$　⑧$\frac{(8+12*\frac{2}{5})^3}{4*10}$

同步训练

(2)编写程序 p_ballmotion.py。假设一个球的初始速度 $v_0 = 15$ 米每秒,初始高度是 $h_0 = 5$ 米,向上笔直抛出这个球。3 秒后球所在的高度是多少?结果保留两位小数,显示高度值。t 秒后的高度公式是 $h_t = h_0 + v_0 * t - \frac{1}{2} * g * t^2$。重力加速度 g 取 9.8。

2.3.3　关系运算

教学视频:关系运算

关系运算用于比较两个数据的大小,关系成立则返回 True,否则返回 False。关系运算符包括 ==、!=、<、>、<=和>=,如表 2-4 所示。

表 2-4　关系运算符

关系运算符	功能描述	案例(已知 a = 100,b = 20)
==	判断等于运算,比较对象的值是否相等	a == b 返回 False
!=	不等运算,比较对象的值是否不相等	a != b 返回 True
<	小于运算	a < b 返回 False
>	大于运算	a > b 返回 True
<=	小于等于运算	a <= b 返回 False
>=	大于等于运算	a >= b 返回 True

使用关系运算符时需要注意以下 3 点。

(1)注意“=”和“==”的差别,前者是赋值等于,后者是判断等于。

(2)复数是不能比较大小的。

(3)字符串、列表和元组比较大小时,按照从左往右的顺序逐个比较,有比没有大。

```
>>> s1 = "abc"; s2 = "Gfe"
>>> print(s1 == s2, s1 != s2, s1 < s2, s1 > s2, s1 <= s2,s1 >= s2)
False True False True False True
>>> s1 = 'abc'; s2 = 'abcde'
>>> print(s1 == s2, s1 != s2, s1 < s2, s1 > s2,s1 <= s2, s1 >= s2)
False True True False True False
```

2.3.4　测试运算

教学视频:测试运算

测试运算包括成员测试(in、not in)和同一性测试(is、not is),测试运算符如表 2-5 所示。

表 2-5　测试运算符

测试运算符	功能描述	案例(已知 s1 = 'abc',s2 = 'abcde')
in	判断前者是否属于后者,是则返回 True,否则返回 False	s1 in s2 返回 True
not in	判断前者是否属于后者,是则返回 False,否则返回 True	s1 not in s2 返回 False
is	判断两者是否为同一对象,是则返回 True,否则返回 False	s1 is s2 返回 False
is not	判断两者是否为同一对象,是则返回 False,否则返回 True	s1 is not s2 返回 True

使用测试运算符时需要注意 is 和 == 的差别。== 要求两个操作数的值相同，而 is 的要求则更高，它要求两者是同一对象，也就是在同一地址。is 比 == 更严格，效率也更高。

```
>>> s1 = s2 = 'abcd#efg'
>>> s3 = 'abcd#efg'
>>> print(s1, s2, s3, sep = '  ')  # s1、s2 和 s3 同值
abcd#efg  abcd#efg  abcd#efg
>>> print(id(s1), id(s2), id(s3), sep = '  ')  # s1 和 s2 同址，但和 s3 不同址
2132803704688  2132803704688  2132803704816
>>> print(s1 == s2, s1 is s2, s1 == s3, s1 is s3, sep = '  ')
True  True  True  False
```

2.3.5 逻辑运算

学习逻辑运算前需要先了解零值和非零值这两个概念。Python 的数据类型一般有其对应的零值，在逻辑运算中等同于 False；除了零值，其他都是非零值，在逻辑运算中等同于 True。

int、float、complex 的零值是 0；字符串的零值是"（即空字符串）；list（列表）的零值是［］（即空列表）；tuple（元组）的零值是（）（即空元组）。None（空值）也等同于 False。

逻辑运算符包括 and、or 和 not，如表 2-6 所示。

表 2-6 逻辑运算符

逻辑运算符	功能描述	案例
A and B	与运算。首先计算表达式 A，若其结果为 False /零值，则直接返回该值，否则继续计算表达式 B 并返回表达式 B 的值	True and True 返回 True True and False 返回 False False and True 返回 False False and False 返回 False
A or B	或运算。首先计算表达式 A，若其结果为 True /非零值，则直接返回该值，否则继续计算表达式 B 并返回表达式 B 的值	True or True 返回 True True or False 返回 True False or True 返回 True False or False 返回 False
not A	非运算。A 为 False/零值时返回 True，A 为 True/非零值时返回 False	not True 返回 False not False 返回 True

使用逻辑运算符时需要注意以下两点。

（1）Python 逻辑运算具有短路概念，如果运算符左边的操作数能最终决定运算结果，则右边的表达式不再计算，直接返回左边的操作数。

（2）虽然零值等同于 False，非零值等同于 True，但是逻辑运算返回的是数据本身。

```
>>> '风轻云淡' and '碧水绕城'
'碧水绕城'
>>> '' and 100 and (10 + 2j)
''
>>> '金蝉脱壳' and '' and '背水一战'
''
>>> 100 or 12.34 or 'abc'
100
```

［技巧］not not(x)将非零值归于 True，零值归于 False。

同步训练

教学视频：口算验证与Python表达式

(1)口算如下题目,并在计算机上进行验证。

①'' and 'Python'

②(100 and(2 ** 3)) or ('' and True)

③(not (2 + 5j)or 100) and ('' or 'Python')

④(not 0 or 100) and ('' and 'Python')

(2) 写出 Python 表达式描述以下问题。

①已有整型 x，判断 x 除以 5 的余数是否为 0。

②已有整型 x、y、z，判断 x 和 y 能否同时被 z 整除。

③已有分数 score，判断该分数是否在［60，80）范围内。

④已有年龄 age 和职业 job，判断年龄是否在［25，30］内且 job 是不是'teacher'。

⑤已有三条边长 a、b、c，判断它们是否能构成三角形。

(3) 编写 p_abc.py，输入字符串 a、b、c，判断并输出 a 是否在 b + c 中。

教学视频：p_abc

```
请输入字符串 a: 天使
请输入字符串 b: 蜀道之难难于上青天
请输入字符串 c: 使人听此凋朱颜
天使 in 蜀道之难难于上青天使人听此凋朱颜 is True
```

2.3.6　位运算

教学视频：位运算

Python 位运算把数字看作二进制进行计算。位运算符包括 &、|、^、~、<< 和 >>，如表 2-7 所示。

表 2-7　位运算符

位运算符	功能描述	案例（已知 a = 12，b = 10）
x & y	按位与运算。x 和 y 的两个对应位都为 1 则该位的结果为 1，否则为 0	a & b 返回 8
x | y	按位或运算。x 和 y 的两个对应位有一个为 1 时结果位就为 1，否则为 0	a | b 返回 14
x ^ y	按位异或运算。x 和 y 的两个对应位相异时结果位为 1	a ^ b 返回 6
~ x	按位取反运算。对 x 的每个位取反，即把 1 变 0，把 0 变 1	~a 返回 -13
x << n	左移动运算。x 的各位全部左移 n 位，高位丢弃，低位补 0	a <<2 返回 48
x >> n	右移动运算。x 的各位全部右移 n 位，低位丢弃	a >>2 返回 3