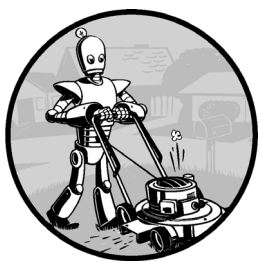
习题答案



本附录包含每章末习题的答案。我强烈建议你花时间解答这些习题。编程不只是记住语法和函数名列表。像学习外语一样，练习越多，收获就越大。有许多网站也包含编程习题。你可以在 No starch 出版社官网对应页面中找到这些网站的列表。

对于练习项目，没有唯一正确的程序。只要你的程序执行了项目要求的内容，就可以认为它是正确的。但是，如果你想看已完成的项目实例，可以在“Download the files used in the book”（下载书中使用的文件）链接中找到。

第 1 章

1．操作符是+、-、*和/。值是'hello'、-88.8 和 5。

2．变量是 spam，字符串是'spam'。字符串总是以引号开始和结束。

3．本章介绍的 3 种数据类型是整型、浮点型和字符串。

4．表达式是值和操作符的结合。所有表达式都求值为（即归约为）一个值。

5．表达式求值为一个值。语句不是这样。

6．bacon 变量被设置为 20。表达式 bacon + 1 并没有对 bacon 重新赋值（重新赋值需要一个赋值语句：bacon =bacon + 1）。

7．两个表达式都求值为字符串'spamspamspam'。

8．变量名不能以数字开始。

9．int()、float()和 str()函数将返回传入值的整型、浮点型和字符串形式。

10．该表达式导致错误是因为 99 是一个整数，只有字符串能用+操作符与其他字符串连接。正确的方式是'I have eaten ' + str(99) + ' burritos.'。

第 2 章

1．True 和 False，使用大写的 *T* 和 *F*，其他字母是小写。

2．and、or 和 not。

3．True and True 是 True。

True and False 是 False。

False and True 是 False。

False and False 是 False。

True or True 是 True。

True or False 是 True。

False or True 是 True。

False or False 是 False。

not True 是 False。

not False 是 True。

4．False

False

True

False

False

True

5．==、!=、<、>、<=和>=。

6．==是等于操作符，它比较两个值，求值为一个布尔值；而=是赋值操作符，将值保存在变量中。

7．条件是一个表达式，它用于控制流语句中，求值为一个布尔值。

8．3 个语句块是 if 语句中的全部内容，以及 print('bacon')和 print('ham')这两行：

```
print('eggs')
if spam > 5:
    print('bacon')
else:
    print('ham')
print('spam')
```

9．代码：

```
if spam == 1:
    print('Hello')
elif spam == 2:
    print('Howdy')
else:
    print('Greetings!')
```

10．按 Ctrl-C 快捷键来停止陷在无限循环中的程序。

11．break 语句将执行移出循环，接着执行循环之后的语句。continue 语句将执行移到循环的开始处。

12．它们都是做同样的事。range(10)调用产生的范围是从 0 直到（但不包括）10，range(0,10)明确告诉循环从 0 开始，range(0,10,1) 明确告诉循环每次迭代让变量增加 1。

13．for 循环代码：

```
for i in range(1, 11):
    print(i)
```

while 循环代码：

```
i = 1
while i <= 10:
    print(i)
    i = i + 1
```

14．该函数的调用方式是 spam.bacon()。

第 3 章

1．函数减少了重复的代码。这让程序更短，更容易阅读，更容易修改。

2．函数中的代码在函数被调用时执行，而不是在函数定义时。

3．用 def 语句定义（即创建）一个函数。

4．函数包含 def 语句和在 def 子句中的代码。函数调用让程序执行转到函数内，函数调用求值为该函数的返回值。

5．有一个全局作用域；在一个函数被调用时，会创建一个局部作用域。

6．函数返回时，局部作用域被销毁，其中所有的变量都被遗忘了。

7． 返回值是函数调用求值的结果。像所有值一样，返回值可以作为表达式的一部分。

8．如果函数没有 return 语句，它的返回值就是 None。

9．global 语句强制函数中的一个变量引用该全局变量。

10．None 的数据类型是 NoneType。

11．import 语句导入了 areallyourpetsnamederic 模块（顺便说一句，这不是一个真正的 Python 模块）。

12．该函数可以通过 spam.bacon() 调用。

13．将可能导致错误的代码行放在一个 try 子句中。

14．将可能导致错误的代码放在 try 子句中。将发生错误时要执行的代码放在 except 子句中。

第 4 章

1．空的列表值，它是一个列表，不包含任何表项。这类似于''是空的字符串值。

2．spam[2] = 'hello'（注意，列表中的第三个值索引是 2，因为第一个值索引是 0）。

3．'d'（注意'3' * 2 是字符串'33'，它被传入 int()函数，然后再除以 11。这最终求值为 3。在使用值的地方，都可以使用表达式）。

4．'d'（负数索引从末尾倒数）。

5．['a', 'b']。

6．1。

7．[3.14, 'cat', 11, 'cat', True, 99]。

8．[3.14, 11, 'cat', True]。

9．列表连接的操作符是+，复制的操作符是*（这和字符串一样）。

10. append()方法只会将值添加在列表末尾，而 insert()方法可以将值添加在列表的任何位置。

11. del 语句和 remove()列表方法是从列表中删除值的两种方法。

12. 列表和字符串都可以传入 len()函数，都有索引和切片，用于 for 循环、连接或复制，并可与 in 和 not in 操作符一起使用。

13. 列表是可以修改的，它们可以添加值、删除值和修改值。元组是不可修改的，它们根本不能改变。而且，元组使用的是括号(和)，而列表使用的是方括号 [和]。

14. (42,)（末尾的逗号是必须的）。

15. 分别使用 tuple()和 list()函数。

16. 它们包含对列表值的引用。

17. copy.copy()函数将浅复制列表，而 copy.deepcopy()函数将深复制列表。也就是说，只有 copy.deepcopy()函数会复制列表内的所有列表。

第 5 章

1. 两个花括号：{}。

2. {'foo': 42}。

3. 保存在字典中的项是无序的，而列表中的项是有序的。

4. 会得到 KeyError 错误。

5. 没有区别。in 操作符检查一个值是不是字典中的一个键。

6. 'cat' in spam 检查该字典中是不是有一个'cat'键，而'cat' in spam.values()检查是否有一个值'cat'对应于 spam 变量中的某个键。

7. spam.setdefault('color', 'black')。

8. pprint.pprint()。

第 6 章

1. 转义字符表示字符串中的一些字符，这些字符用别的方式很难在代码中输出。

2. \n 是换行符，\t 是制表符。

3. \\转义字符表示一个反斜杠。

4. Howl's 中的单引号没有问题，因为使用了双引号来标识字符串的开始和结束。

5. 多行字符串让你在字符串中使用换行符，而不必用\n 转义字符。

6. 这些表达式求值为以下值：

- 'e'
- 'Hello'
- 'Hello'
- 'lo, world!

7. 这些表达式求值为以下值：

- 'HELLO'
- True
- 'hello'

8．这些表达式求值为以下值：

- ['Remember,', 'remember,', 'the', 'fifth', 'of', 'November.']
- 'There-can-be-only-one.'

9．分别用 rjust()、ljust()和 center()字符串方法。

10．可用 lstrip()和 rstrip()方法分别从字符串的左边和右边移除空白字符。

第 7 章

1．re.compile()函数。

2．使用原始字符串是为了让反斜杠不必转义。

3．search()方法返回 Match 对象。

4．group()方法返回匹配文本的字符串。

5．分组 0 是整个匹配，分组 1 包含第一组括号，分组 2 包含第二组括号。

6．句号和括号可以用反斜杠转义：\.、\(和\)。

7．如果正则表达式没有分组，就返回字符串的列表。如果正则表达式有分组，就返回字符串的元组的列表。

8．| 字符表示匹配两个组中的“任何一个”。

9．? 字符可以表示“匹配前面分组零次或一次”，或用于表示非贪心匹配。

10．+匹配一次或多次。*匹配零次或多次。

11．{3}匹配前面分组的 3 次实例。{3, 5}匹配 3 至 5 次实例。

12．缩写字符分类\d、\w 和\s 分别匹配一个数字、单词和空白字符。

13．缩写字符分类\D、\W 和\S 分别匹配一个不是数字、单词或空白字符的字符。

14．.*执行贪心匹配，.*?执行非贪心匹配。

15．[0-9a-z]或[a-z0-9]。

16．将 re.I 或 re.IGNORECASE 作为第二个参数传入 re.compile()，让匹配不区分大小写。

17．字符.通常匹配任何字符，换行符除外。如果将 re.DOTALL 作为第二个参数传入 re.compile()，那么.也会匹配换行符。

18．'X drummers, X pipers, five rings, X hens'

19．re.VERBOSE 参数允许为传入 re.compile()的字符串添加空格和注释。

20．re.compile（r'^\d{1,3}(,{3})*$'）将创建这个正则表达式，但其他正则表达式字符串可以生成类似的正则表达式。

21．re.compile(r'[A-Z][a-z]*\sNakamoto')。

22．re.compile(r'(Alice|Bob|Carol)\s(eats|pets|throws)\s(apples|cats|baseballs)\.', re.IGNORECASE)。

第 8 章

1．不是。PyInputPlus 是一个第三方模块，不随 Python 标准库一起提供。

2．这可以让你的代码更短：你可以输入 pyip.inputStr()，而不是输入 pyinputplus.inputStr()。

3．inputInt()函数返回的是一个 int 值，而 inputFloat()函数返回的是 float 值。这就是返回 4 和 4.0 的区别。

4．调用 pyip.inputint(min=0, max=99)。

5．显示允许或拒绝的 regex 字符串列表。

6．该函数将引发 RetryLimitException。

7．函数会返回值'hello'。

第 9 章

1．相对路径是相对于当前工作目录。

2．绝对路径从根文件夹开始，如/或 C:\。

3．在 Windows 操作系统上，它求值为 WindowsPath('C:/Users/Al')。在其他操作系统上，它求值为不同类型的 Path 对象，但路径相同。

4．表达式'C:/Users'/'Al'会导致一个错误，因为你不能使用/操作符来连接两个字符串。

5．os.getcwd()函数返回当前工作目录。os.chdir()函数改变当前工作目录。

6．文件夹.是当前文件夹，文件夹..是父文件夹。

7．C:\bacon\eggs 是目录名，而 spam.txt 是基本名称。

8．字符串'r'对应读模式，'w'对应写模式，'a'对应添加模式。

9．已有的文件用写模式打开，原有内容会被删除并完全覆写。

10．read()方法将文件的全部内容作为一个字符串返回。readlines()返回一个字符串列表，其中每个字符串是文件内容中的一行。

11．shelf 值类似于字典值，它有键和值；keys()和 values()方法类似于同名的字典方法。

第 10 章

1．shutil.copy()函数将复制一个文件，而 shutil.copytree()将复制整个文件夹以及它的所有内容。

2．shutil.move()函数用于重命名文件以及文件移动。

3．send2trash 模块中的删除函数将一个文件或文件夹移到回收站，而 shutil 模块中的删除函数将永久地删除文件和文件夹。

4．Zipfile.ZipFile()函数等价于 open()函数，第一个参数是文件名，第二个参数是打

开 ZIP 文件的模式（读、写或添加）。

第 11 章

1．`assert spam >= 10,'The spam variable is less than 10.'`。

2．`assert eggs.lower() != bacon.lower(), 'The eggs and bacon variables are the same!'`或`assert eggs.upper() != bacon.upper(), 'The eggs and bacon variables are the same!'`。

3．`assert False, 'This assertion always triggers.'`。

4．为了能调用`logging.debug()`，必须在程序开始时加入以下两行：

```
import logging
logging.basicConfig(level=logging.DEBUG, format=' %(asctime)s -
%(levelname)s - %(message)s')
```

5．为了能利用`logging.debug()`将日志消息发送到文件`programLog.txt`中，必须在程序开始时加入以下两行：

```
import logging
>>> logging.basicConfig(filename='programLog.txt', level=logging.DEBUG,
format=' %(asctime)s - %(levelname)s - %(message)s')
```

6．DEBUG、INFO、WARNING、ERROR 和 CRITICAL。

7．`logging.disable (logging.CRITICAL)`。

8．可以禁用日志消息，且不必删除日志函数调用。可以选择禁用低级别日志消息。可以创建日志消息。日志消息提供了时间戳。

9．单击 Step In 按钮让调试器进入函数调用。单击 Step Over 按钮将快速执行函数调用，不会单步进入其中。单击 Step Out 按钮将快速执行余下的代码，直到走出当前所处的函数。

10．在单击 Continue 按钮后，调试器将在程序末尾或断点处停止。

11．断点设在一行代码上，在程序执行到达该行时，它将导致调试器暂停。

12．要在 Mu 中设置断点，就在行号上单击，使得它旁边出现一个红点。

第 12 章

1．`webbrowser`模块有一个`open()`方法，它启动 Web 浏览器，打开指定的 URL。`requests`模块可以从网上下载文件和页面。`BeautifulSoup`模块解析 HTML。`selenium`模块可以启动并控制浏览器。

2．`requests.get()`函数返回一个`Response`对象，它有一个`text`属性，包含下载内容的字符串。

3．如果下载有问题，`raise_for_status()`方法将抛出异常；如果下载成功，什么也不做。

4．`Response`对象的`status_code`属性包含了 HTTP 状态码。

5．以`'wb'`，即“写二进制”模式在你的计算机上打开新文件后，利用一个`for`循环迭代遍历

Response 对象的 iter_content()方法，将各段写入该文件。下面是例子：

```
saveFile = open('filename.html', 'wb')
for chunk in res.iter_content(100000):
    saveFile.write(chunk)
```

6．按 F12 键在 Chrome 中打开开发者工具。按 Ctrl-Shift-C 快捷键（在 Windows 和 Linux 操作系统上）或 Command-option-C 快捷键（在 macOS 上），在 Firefox 中打开开发者工具。

7．右击页面上的元素，并从菜单中选择 Inspect Element。

8．'#main'。

9．'.highlight'。

10．'div div'。

11．'button[value="favorite"]'。

12．spam.getText()。

13．linkElem.attrs。

14．selenium 模块是通过运行 from selenium import webdriver 导入的。

15．find_element_*方法将第一个匹配的元素返回，作为一个 WebElement 对象。find_elements_*方法返回所有匹配的元素，作为一个 WebElement 对象列表。

16．click()和 send_keys()方法分别模拟鼠标点击和键盘按键。

17．对表单中的任意元素调用 submit()方法将提交该表单。

18．forward()、back()和 refresh()等 WebDriver 对象方法模拟了单击这些浏览器按钮。

第 13 章

1．openpyxl.load_workbook()函数返回一个 Workbook 对象。

2．sheetnames 属性返回一个 Worksheet 对象。

3．执行 wb['Sheet1']。

4．用 wb.active。

5．sheet['C5'].value 或 sheet.cell(row=5, column=3).value。

6．sheet['C5']='Hello'或 sheet.cell(row=5,column=3).value='Hello'。

7．使用 cell.row 和 cell.column。

8．它们分别返回表中最高列和最高行的整数值整型。

9．openpyxl.cell.column_index_from_string('M') 。

10．openpyxl.cell.get_column_letter(14) 。

11．sheet['A1':'F1']。

12．wb.save('example.xlsx')。

13．公式的设置和值一样。将单元格的 value 属性设置为公式文本的字符串。记住公式以=号开始。

14．在调用 load_workbook() 时，传入 True 作为 data_only 关键字参数。

15．sheet.row_dimensions[5].height = 100。

16. sheet.column_dimensions['C'].hidden = True。

17. 冻结窗格就是总是会出现在屏幕上的行和列。它们作为表头是很有用的。

18. openpyxl.charts.Reference()、openpyxl.charts.Series()、openpyxl.charts.BarChart()、chartObj.append(seriesObj)和 add_chart()。

第 14 章

1. 要访问 Google Sheets，你需要一个证书文件、Google Sheets 的令牌文件和 Google Drive 的令牌文件。

2. EZSheets 有 ezsheets.Spreadsheet 和 ezsheets.Sheet 对象。

3. 调用 Spreadsheet 的 downloadAsExcel()方法。

4. 调用 ezsheets.upload()函数并传入 Excel 文件的文件名。

5. 访问 ss['Students']['B2']。

6. 调用 ezsheets.getColumnLetterOf(999)函数。

7. 访问 Sheet 对象的 rowCount 和 columnCount 属性。

8. 调用 Sheet 的 delete()方法。只有当传入 permanent=True 关键字参数时，这种删除才是永久的。

9. Spreadsheet 的 createSpreadsheet()函数和 createSheet()方法将分别创建 Spreadsheet 和 Sheet 对象。

10. EZSheets 将限制你的方法调用。

第 15 章

1. 由 open()返回的 File 对象。

2. 对 PdfFileReader()用读二进制（'rb'），对 PdfFileWriter()用写二进制（'wb'）。

3. 调用 getPage(4)将返回第 5 页的 Page 对象，因为 0 页就是第 1 页。

4. 在 PdfFileReader 对象中，numPages 变量保存了页数的整数。

5. 调用 decrypt('swordfish')。

6. rotateClockwise()和 rotateCounterClockwise()方法。旋转度数作为整数参数传入。

7. docx.Document('demo.docx')。

8. 文档包含多个段落。段落从一个新行开始，包含多个 Run 对象。Run 对象是段落内连续的字符分组。

9. 使用 doc.paragraphs。

10. Run 对象有这些属性（不是 Paragraph）。

11. True 总是让 Run 对象成为粗体，False 总是让它不是粗体，不论样式的粗体设置是什么。None 让 Run 对象使用该样式的粗体设置。

12. 调用 docx.Document()函数。

13. doc.add_paragraph('Hello there!')

14．整数 0、1、2、3 和 4。

第 16 章

1．在 Excel 中，电子表格的值可以是字符串以外的数据类型，单元格可以有不同的字体、大小或颜色设置，单元格可以有不同的宽度和高度，相邻的单元格可以合并，可以嵌入图像和图表。

2．传入一个 File 对象，通过调用 open()获得。

3．对于 reader 对象，File 对象需要以读二进制模式（'rb'）打开；对于 writer 对象，需要以写二进制模式（'wb'）打开。

4．writerow()方法。

5．delimiter 参数改变了分隔行中单元格所用的字符串。lineterminator 参数改变了分隔行的字符串。

6．json.loads()。

7．json.dumps()。

第 17 章

1．许多日期和时间程序使用的一个参考时刻，该时刻是 1970 年 1 月 1 日 0 点，UTC。

2．time.time()。

3．time.sleep(5)。

4．返回与传入参数最近的整数。例如，round(2.4)返回 2。

5．datetime 对象表示一个特定的时刻。timedelta 对象表示一段时间。

6．运行 datetime.datetime(2019，1，7).weekday()，它返回 0，这意味着周一，因为 datetime 模块使用 0 表示周一，1 表示周二，以此类推，6 表示周日。

7．threadObj = threading.Thread(target=spam)threadObj.start()。

8．确保在一个线程中执行的代码不会和另一个线程中的代码读写相同的变量。

第 18 章

1．分别是 SMTP 和 IMAP。

2．smtplib.SMTP()、smtpObj.ehlo()、smptObj.starttls()和 smtpObj. login()。

3．imapclient.IMAPClient()和 imapObj.login()。

4．IMAP 关键字的字符串列表，例如'BEFORE <date>'、'FROM <string>'或'SEEN'。

5．将变量 imaplib._MAXLINE 赋值为一个大整数，例如 10000000。

6．pyzmail 模块读负责取下载的邮件。

7．credentials.json 和 token.json 文件告诉 ezgmail 模块在访问 Gmail 时要使用哪个 Google 账户。

8．一个消息代表一个邮件，而涉及多个邮件的来回对话是一个主线。

9．在你传递给 search()的字符串中包含'has:atttachment'文本。

10．你需要 Twilio 账户的 SID 号、认证标识号，以及你的 Twilio 电话号码。

第 19 章

1．RGBA 值是 4 个整数的元组，每个整数的范围是 0 至 255。4 个整数对应于颜色的红、绿、蓝和 alpha 值（透明度）。

2．函数调用 ImageColor.getcolor('CornflowerBlue', 'RGBA')将返回(100, 149, 237, 255)，即该颜色的 RGBA 值。

3．矩形元组是 4 个整数的元组：分别是左边的 *x* 坐标，顶边的 *y* 坐标，宽度和高度。

4．Image.open('zophie.png')。

5．imageObj.size。这是由两个整数组成的元组，表示宽度和高度。

6．imageObj.crop((0, 50, 50, 50))。请注意，传入 crop()的是一个矩形元组，不是 4 个独立的整数参数。

7．调用 Image 对象的 imageObj.save('new_filename.png')方法。

8．ImageDraw 模块包含在图像上绘画的代码。

9．ImageDraw 对象有一些绘制形状的方法，例如 point()、line()或 rectangle()。这些对象是将 Image 对象传入 ImageDraw.Draw()函数后返回的。

第 20 章

1．将鼠标指针移到屏幕的左上角，即坐标（0，0）处。

2．pyautogui.size()返回两个整数的元组，表示屏幕的宽和高。

3．pyautogui.position()返回两个整数的元组，表示鼠标指针的 *x* 坐标和 *y* 坐标。

4．moveTo()函数将鼠标指针移到屏幕的绝对坐标处，而 move()函数相对于鼠标指针的当前位置来移动鼠标指针。

5．pyautogui.dragTo()和 pyautogui.drag()。

6．pyautogui.typewrite('Hello, world!')。

7．要么向 pyautogui.typewrite()输入键盘按键字符串的列表（例如'left'），要么向 pyautogui.press()输入单个键盘按键字符串。

8．pyautogui.screenshot('screenshot.png')。

9．pyautogui.PAUSE = 2。

10．你应该使用 Selenium 来控制一个 Web 浏览器，而不是 PyAutoGUI。

11．PyAutoGUI 会盲目地点击和输入，不容易发现它的点击和输入窗口是否正确。意外的弹出窗口或错误会使脚本偏离预期的工作，需要你关闭它。

12．调用 pyautogui.getWindowsWithTitle('Notepad')函数。

13．运行 w = pyatuogui.getWindowsWithTitle('Firefox')，然后运行 w.activate()。