普通高等教育“十三五”人工智能与大数据规划教材

大数据的Python基础

Python Foundation for Big Data

董付国　编著

提供
教学课件和
全部例题源码

基础与应用并重，理论结合实践
案例丰富，代码纯正，解读Pythonic真谛

全部代码
适用于*Python*
3.5/3.6/3.7/3.8

机械工业出版社
CHINA MACHINE PRESS

普通高等教育“十三五”人工智能与大数据规划教材

大数据的 Python 基础

董付国　编著

机械工业出版社

全书共 10 章。第 1 章介绍 Python 开发环境的搭建、简单使用和 Python 代码规范；第 2 章讲解 Python 内置对象与运算符的使用；第 3 章讲解列表、元组、列表推导式与生成器表达式以及切片和序列解包的用法；第 4 章讲解字典应用；第 5 章讲解集合应用；第 6 章讲解字符串与正则表达式的应用；第 7 章讲解选择结构与循环结构语法和应用；第 8 章讲解函数设计与应用；第 9 章讲解文本文件操作、二进制文件操作、Office 文档操作以及文件夹操作；第 10 章讲解如何使用 Python 扩展库 numpy 和 pandas 处理数据。

本书全部代码适用于 Python 3.5/3.6/3.7 以及更新的版本。本书可以作为数据科学与大数据技术专业和计算机、电子信息等其他相关专业的 Python 程序设计课程教材，也可以作为相关的工程技术人员学习 Python 程序设计时的快速入门参考书。

本书配有电子课件和全部例题源码，欢迎选用本书作教材的教师登录 www.cmpedu.com 注册下载，或发邮件至 jinacmp@163.com 索取。

图书在版编目（CIP）数据

大数据的 Python 基础 / 董付国编著. —北京：机械工业出版社，2019.5

普通高等教育“十三五”人工智能与大数据规划教材

ISBN 978-7-111-62455-4

Ⅰ. ①大… Ⅱ. ①董… Ⅲ. ①软件工具—程序设计—高等学校—教材 Ⅳ. ①TP311.561

中国版本图书馆 CIP 数据核字（2019）第 068051 号

机械工业出版社（北京市百万庄大街 22 号 邮政编码 100037）

策划编辑：吉 玲 责任编辑：吉 玲 王 康

责任校对：张晓蓉 封面设计：张 静

责任印制：孙 炜

天津嘉恒印务有限公司印刷

2019 年 5 月第 1 版第 1 次印刷

184mm×260mm • 13 印张 • 315 千字

标准书号：ISBN 978-7-111-62455-4

定价：34.80 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

电话服务

服务咨询热线：010-88379833

读者购书热线：010-68326294

网络服务

机 工 官 网：www.cmpbook.com

机 工 官 博：weibo.com/cmp1952

教育服务网：www.cmpedu.com

金 书 网：www.golden-book.com

前言

本书是机械工业出版社策划的普通高等教育“十三五”人工智能与大数据规划教材中的一本，根据丛书的总体设计，本书定名为《大数据的 Python 基础》。虽然在整理内容时确实适当增加了数据分析与处理需要用到的 Python 基础知识和相关案例，但实际上大部分内容是使用 Python 做任何领域的开发都要用到的基础，毕竟基础知识是通用的。全书共 72 个例题，文件操作的案例稍多一些，不管是日常工作还是数据采集与分析，都用得到。还有很多例子是隐藏在基础知识和代码片段中的，需要仔细挖掘和体会。

笔者个人建议，书上的内容至少要阅读三遍。第一遍通览全书，快速了解总体内容和知识框架，并大概了解一些术语。第二遍仔细阅读并认真练习每一段代码和每一个例题，动手输入并运行这些代码，确保结果正确。第三遍重点体会书上的代码为什么要这样写，想想还有没有更好的写法，思考一下每个代码片段主要解决了什么问题，不同代码片段和例题的集成又可以解决哪些问题。

本书全部代码适用于 Python 3.5/3.6/3.7 以及更新的版本，可以作为 Python 爱好者的快速入门参考书，也可以作为数据科学与大数据技术专业和计算机、电子信息等其他相关专业的 Python 程序设计课程教材。作为教材建议采用 32 学时或 48 学时。

欢迎关注笔者的微信公众号“Python 小屋”，免费阅读超过 800 篇原创 Python 技术文章或者免费观看超过 300 节 Python 微课，虽然微课中的案例与本书并不完全一样，但是很多知识点是相通的，可以触类旁通，也有助于对本书中案例的学习和理解。选用本书作教材的教师可以在笔者的微信公众号文章中选择更多习题和实验项目用于教学。如果广大读者朋友和用书教师发现书中不足之处，欢迎通过微信公众号留言或者用电子邮箱 dongfuguo2005@126.com 与笔者联系，在此表示衷心感谢！

董付国

Python 小屋

学时分配建议

本书内容适用于 32 学时或 48 学时的 Python 程序设计课程，采用 32 学时教学请参考下表第 2 列，采用 48 学时教学请参考第 3 列。另外，建议搭配 16 学时的上机练习（有教师指导），以及至少 40 学时的课外阅读或练习时间（无教师指导），这样才能保证学习效果，为后续课程打下良好的基础。

建议把较多精力放在第 2、3、4、6、9 章的内容讲解上，具体学时分配可根据教学进度和学习效果进行适当调整。

章次	学时（32）	学时（48）	上机（16）	课外（40）
第 1 章　Python 开发环境搭建与使用	2	2		2
第 2 章　Python 常用内置对象与运算符	4	6	2	4
第 3 章　列表与元组	4	6	2	6
第 4 章　字典	2	2	2	2
第 5 章　集合	2	4	2	2
第 6 章　字符串与正则表达式	4	6	2	6
第 7 章　程序控制结构	2	4	2	4
第 8 章　函数设计与应用	4	6	2	4
第 9 章　文件与文件夹操作	4	6	2	8
第 10 章　numpy 与 pandas 基础	4	6		2

目　录

前　言

学时分配建议

第 1 章　Python 开发环境搭建与使用 …… 1

本章学习目标 …… 1

1.1　Python 语言概述 …… 1

1.2　Python 开发环境搭建 …… 2

1.2.1　IDLE …… 2

1.2.2　Anaconda3 …… 3

1.3　安装扩展库 …… 5

1.4　标准库与扩展库对象的导入与使用 …… 7

1.5　Python 代码布局规范 …… 8

本章知识要点 …… 9

习题 …… 9

第 2 章　Python 常用内置对象与运算符 …… 10

本章学习目标 …… 10

2.1　Python 常用内置对象 …… 10

2.1.1　常量与变量 …… 11

2.1.2　数字 …… 13

2.1.3　字符串 …… 14

2.1.4　列表、元组、字典、集合 …… 15

2.2　Python 运算符与表达式 …… 15

2.2.1　算术运算符 …… 16

2.2.2　关系运算符 …… 18

2.2.3　成员测试运算符 …… 19

2.2.4　集合运算符 …… 19

2.2.5　逻辑运算符 …… 19

2.3　Python 常用内置函数 …… 20

2.3.1　类型转换函数 …… 22

2.3.2　max()、min()、sum() …… 25

2.3.3　input()、print() …… 26

2.3.4　sorted()、reversed() …… 27

2.3.5　map() …… 29

2.3.6　reduce() …… 30

2.3.7　filter() …… 31

2.3.8 range() …… 32
2.3.9 zip() …… 32
本章知识要点 …… 33
习题 …… 34
第 3 章 列表与元组 …… 35
本章学习目标 …… 35
3.1 列表 …… 35
3.1.1 列表创建与删除 …… 35
3.1.2 列表元素访问 …… 36
3.1.3 列表常用方法 …… 37
3.1.4 列表对象支持的运算符 …… 39
3.1.5 内置函数对列表的操作 …… 40
3.2 列表推导式语法与应用 …… 41
3.3 元组与生成器表达式 …… 45
3.3.1 元组创建与元素访问 …… 45
3.3.2 元组与列表的区别 …… 45
3.3.3 生成器表达式 …… 46
3.4 切片语法与应用 …… 47
3.5 序列解包 …… 49
本章知识要点 …… 50
习题 …… 50
第 4 章 字典 …… 52
本章学习目标 …… 52
4.1 基本概念 …… 52
4.2 字典创建与删除 …… 52
4.3 字典元素访问 …… 53
4.4 字典元素添加、修改与删除 …… 54
4.5 字典应用案例 …… 55
本章知识要点 …… 57
习题 …… 58
第 5 章 集合 …… 59
本章学习目标 …… 59
5.1 基本概念 …… 59
5.2 集合创建与删除 …… 59
5.3 集合常用操作与运算 …… 60
5.3.1 集合元素增加与删除 …… 60
5.3.2 集合运算 …… 61
5.3.3 内置函数对集合的操作 …… 61
5.4 集合应用案例 …… 62

本章知识要点 …… 64
习题 …… 65
第 6 章 字符串与正则表达式 …… 66
本章学习目标 …… 66
6.1 字符串编码格式 …… 66
6.2 转义字符与原始字符串 …… 67
6.3 字符串常用方法与操作 …… 67
6.3.1 format() …… 68
6.3.2 encode() …… 69
6.3.3 find()、rfind()、index()、rindex()、count() …… 70
6.3.4 split()、rsplit() …… 71
6.3.5 join() …… 72
6.3.6 lower()、upper()、capitalize()、title()、swapcase() …… 72
6.3.7 replace()、maketrans()、translate() …… 73
6.3.8 strip()、rstrip()、lstrip() …… 74
6.3.9 startswith()、endswith() …… 75
6.3.10 isalnum()、isalpha()、isdigit()、isspace()、isupper()、islower() …… 75
6.3.11 center()、ljust()、rjust() …… 75
6.3.12 字符串支持的运算符 …… 76
6.3.13 适用于字符串的内置函数 …… 76
6.3.14 字符串切片 …… 77
6.3.15 数据压缩与解压缩 …… 78
6.4 正则表达式语法与应用 …… 78
6.4.1 正则表达式基本语法 …… 78
6.4.2 使用正则表达式模块 re 处理字符串 …… 80
6.5 分词与中文拼音处理 …… 82
6.5.1 分词 …… 82
6.5.2 中文拼音处理 …… 83
6.6 应用案例 …… 84
本章知识要点 …… 93
习题 …… 93
第 7 章 程序控制结构 …… 95
本章学习目标 …… 95
7.1 基本语法 …… 95
7.1.1 条件表达式 …… 95
7.1.2 选择结构基本语法 …… 95
7.1.3 循环结构基本语法 …… 98
7.1.4 异常处理结构基本语法 …… 99
7.2 应用案例 …… 100

本章知识要点 …… 103
习题 …… 103
第 8 章 函数设计与应用 …… 104
本章学习目标 …… 104
8.1 函数定义与使用 …… 104
8.1.1 基本语法 …… 104
8.1.2 递归函数 …… 105
8.1.3 函数嵌套定义 …… 106
8.2 函数参数 …… 107
8.2.1 位置参数 …… 107
8.2.2 默认值参数 …… 108
8.2.3 关键参数 …… 108
8.2.4 可变长度参数 …… 109
8.3 变量作用域 …… 110
8.4 lambda 表达式 …… 111
8.5 生成器函数 …… 112
8.6 应用案例 …… 113
本章知识要点 …… 122
习题 …… 123
第 9 章 文件与文件夹操作 …… 124
本章学习目标 …… 124
9.1 文件的概念及分类 …… 124
9.2 文件操作基本知识 …… 125
9.2.1 内置函数 open() …… 125
9.2.2 文件对象常用方法 …… 125
9.2.3 上下文管理语句 with …… 126
9.3 文本文件内容操作案例 …… 126
9.4 JSON 文件操作 …… 127
9.5 CSV 文件操作 …… 129
9.6 二进制文件操作 …… 129
9.6.1 使用 pickle 模块读写二进制文件 …… 130
9.6.2 使用 struct 模块读写二进制文件 …… 131
9.7 标准库对文件与文件夹的操作 …… 132
9.7.1 os 模块 …… 132
9.7.2 os.path 模块 …… 133
9.7.3 shutil 模块 …… 134
9.8 Excel 与 Word 文件操作案例 …… 136
本章知识要点 …… 148
习题 …… 148

第 10 章　numpy 与 pandas 基础 …… 149
本章学习目标 …… 149
10.1　numpy 数组运算与矩阵运算基础 …… 149
10.1.1　数组生成与常用操作 …… 149
10.1.2　矩阵生成与常用操作 …… 160
10.1.3　计算特征值与特征向量 …… 164
10.1.4　计算逆矩阵 …… 165
10.1.5　矩阵 QR 分解 …… 165
10.1.6　计算行列式 …… 165
10.1.7　矩阵奇异值分解 …… 166
10.1.8　求解线性方程组 …… 167
10.1.9　计算矩阵和向量的范数 …… 167
10.2　pandas 数据处理基础 …… 167
10.2.1　一维数组 Series 与常用索引数组生成与操作 …… 168
10.2.2　创建二维数组 DataFrame …… 171
10.2.3　DataFrame 常用操作 …… 173
10.2.4　缺失值处理 …… 178
10.2.5　重复值处理 …… 179
10.2.6　异常值处理 …… 180
10.2.7　分组计算 …… 181
10.2.8　透视转换与交叉表 …… 182
10.2.9　数据差分 …… 184
10.2.10　计算相关系数 …… 185
本章知识要点 …… 186
习题 …… 187
习题答案 …… 188
附录 …… 189
附录 A　Python 编程常见问题与解答 …… 189
附录 B　Python 关键字清单 …… 192
附录 C　常用 Python 内置模块与标准库清单 …… 193
附录 D　常用 Python 扩展库清单 …… 194
参考文献 …… 195

第 1 章　Python 开发环境搭建与使用

本章学习目标

- 了解 Python 语言的特点
- 了解 Python 语言的应用领域
- 熟练安装和配置 Python 开发环境
- 了解 IDLE、Jupyter Notebook、Spyder 的简单使用
- 熟练安装常用的 Python 扩展库
- 了解 Python 代码编写规范
- 熟练掌握导入和使用 Python 标准库与扩展库对象的方法

1.1　Python 语言概述

Python 是一门跨平台、开源、免费的解释型高级动态编程语言，是一种通用编程语言。Python 目前已经渗透到系统安全、数值计算、统计分析、科学计算可视化、逆向工程与软件分析、图形图像处理、人工智能、机器学习、网站开发、数据爬取与大数据处理、密码学、系统运维、音乐编程、影视特效制作、计算机辅助教育、医药辅助设计、天文信息处理、化学、生物信息处理、神经科学与心理学、自然语言处理、电子电路设计、电子取证、游戏设计与策划、移动终端开发、树莓派开发等几乎所有专业和领域，在大数据和人工智能领域更是不可或缺的语言和技术。

除了可以解释执行，Python 支持将源代码伪编译为字节码来提高加载速度，还支持使用 py2exe、pyinstaller、cx_Freeze、py2app 或其他工具将 Python 程序及其所有依赖库打包成为各种平台上的可执行文件。

Python 支持命令式编程和函数式编程两种模式（推荐使用后者），完全支持面向对象程序设计，语法简洁清晰，功能强大且易学易用，最重要的是拥有大量的几乎支持所有领域应用开发的成熟扩展库。如果刚刚接触 Python 语言，甚至初次接触编程，可能本章提到的一些概念会比较陌生。不过没关系，一些术语可以先跳过去，随着后面章节的展开，会很快就熟悉这些概念了。

目前，Python 官方网站同时发行和维护着 Python 2.x 和 Python 3.x 两个不同系列的版本，这两个系列的版本之间很多用法和扩展库是不兼容的。Python 3.x 的设计更加合理、高效和人性化，代码开发和运行效率更高。另外，Python 官方早在几年前已经宣布，最迟到 2020 年将会全面放弃 Python 2.x 的维护和更新。所以，如果正在使用 2.x 系列，那么最好尽快转换成 3.x 并且选择较高的版本。如果刚刚开始接触 Python，那么一定要毫不犹豫地选择最新的 3.x

正式发行版本。

1.2 Python 开发环境搭建

常用的 Python 开发环境除了 Python 官方安装包自带的 IDLE，还有 Anaconda3、PyCharm、Eclipse、zwPython 等。相对来说，Python 安装包自带的 IDLE 环境稍微简陋一些，虽然也提供了语法高亮（使用不同的颜色显示不同的语法元素）、交互式运行、程序编写与运行以及简单的程序调试功能，但没有项目管理与版本控制等功能，而这些在大型软件开发中是非常重要的。其他 Python 开发环境对 Python 解释器主程序进行了不同程度的封装和集成，使得代码编写和项目管理更加方便一些。本节简单介绍 IDLE 和 Anaconda3 的用法，书中代码主要通过 Win 10+Python 3.6/64 位+IDLE 演示，同样也可以在 Python 的其他开发环境中运行。

1.2.1 IDLE

IDLE 应该算是最原生态的 Python 开发环境之一，默认没有集成任何扩展库，也不具备强大的项目管理功能，如果用来开发大型系统的话，要求用户具有深厚的 Python 功底和项目管理能力。

在 Python 官方网站 https://www.python.org/下载最新的 Python 3.6.x 或 Python 3.7.x 安装包（根据自己计算机操作系统选择 Windows、Mac OS X 或其他平台以及 32 位或 64 位）并安装（建议安装路径为 C:\Python36 或 C:\Python37）后，在“开始”菜单中可以打开 IDLE，如图 1-1 所示。

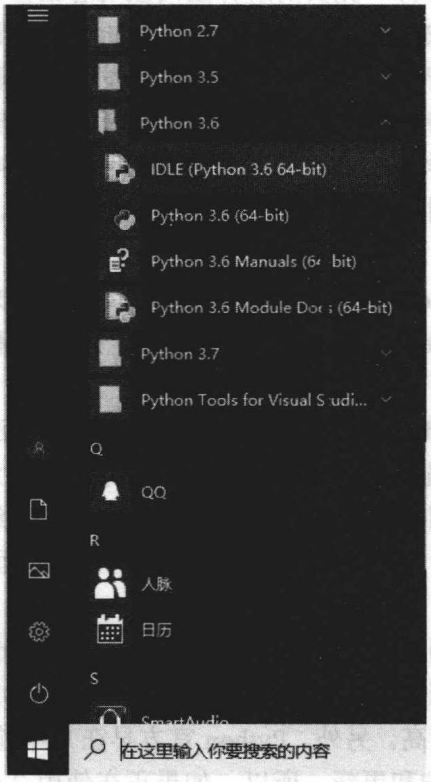

图 1-1 “开始”菜单

打开之后看到的界面就是默认的交互式开发环境，图 1-2 展示了 Python 3.7 的 IDLE 交互式开发界面，其他版本与此基本类似。

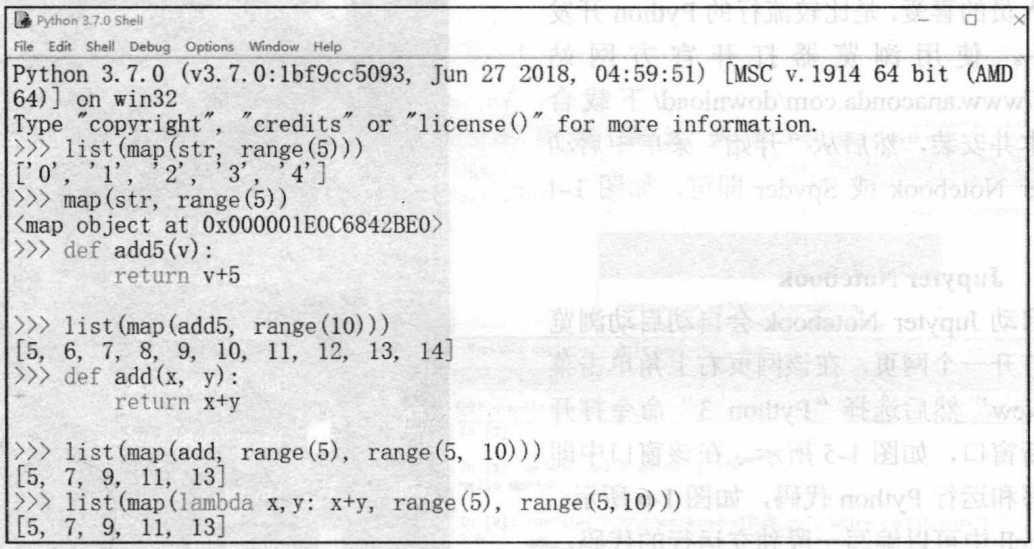

图 1-2　IDLE 交互式开发界面

在交互式开发环境中，“>>>”表示提示符，可以在提示符后面输入语句然后按 Enter 键执行。在交互式开发环境中，每次只能执行一条语句，必须等再次出现提示符“>>>”时才可以输入下一条语句。普通语句可以直接按 Enter 键运行并立刻输出结果，选择结构、循环结构、函数定义、类定义、with 块等属于一条复合语句，需要按两次 Enter 键才能执行。

如果要编写和执行大段代码，一般很难一次顺利完成，可能需要反复修改代码并查看运行结果，也可能需要保存为文件以备日后使用，或者保存为模块以便在其他程序中使用，或者使用多个程序文件组成更大的项目。可以在 IDLE 中单击菜单“File”==>“New File”命令创建一个程序文件，将其保存为扩展名为.py 或.pyw 的文件，注意文件名不要和标准库或已安装的扩展库文件名相同，否则会影响运行。保存后按<F5>键或单击菜单“Run”==>“Run Module”命令运行程序，然后结果会显示到交互式窗口中，如图 1-3 所示。在交互式开发环境中把一个表达式作为语句允许可以直接看到结果，而在程序中如果需要查看某个值，必须使用内置函数 print()将其输出，结果会显示到交互式开发环境中。

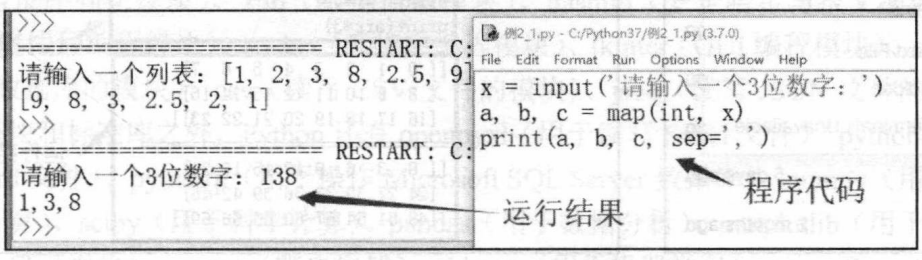

图 1-3　使用 IDLE 编写和运行 Python 程序

1.2.2　Anaconda3

Anaconda3 安装包集成了大量常用的 Python 扩展库，大幅度节约了扩展库安装和配置的

时间，主要提供了 Jupyter Notebook 和 Spyder 两个开发环境，得到了广大初学者和教学、科研人员的喜爱，是比较流行的 Python 开发环境。使用浏览器打开官方网站 https://www.anaconda.com/download/ 下载合适版本并安装，然后从“开始”菜单中启动 Jupyter Notebook 或 Spyder 即可，如图 1-4 所示。

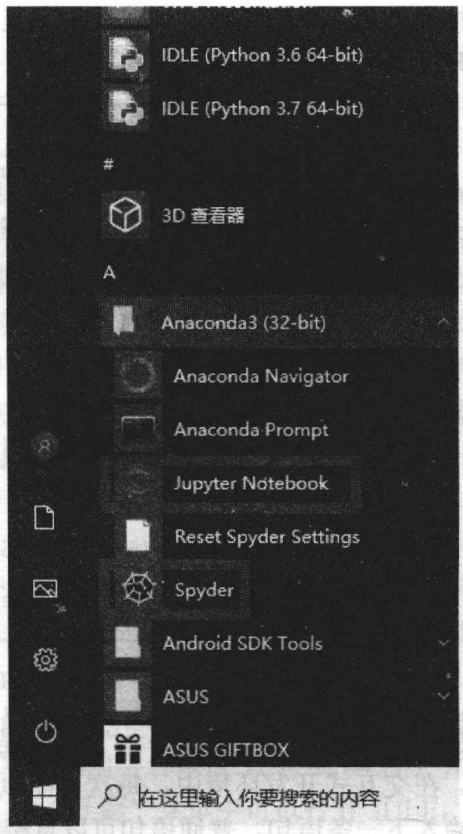

图 1-4 “开始”菜单

1. Jupyter Notebook

启动 Jupyter Notebook 会自动启动浏览器并打开一个网页，在该网页右上角单击菜单“New”然后选择“Python 3”命令打开一个新窗口，如图 1-5 所示。在该窗口中即可编写和运行 Python 代码，如图 1-6 所示。每个 cell 中可以编写一段独立运行的代码，但是前面的 cell 运行结果会影响后面的 cell，也就是前面 cell 中定义的变量在后面的 cell 中仍可以访问，这一点要注意。另外，还可以通过菜单“File”==>“Download as”命令把当前代码以及运行结果保存为不同形式的文件，方便日后学习和演示。

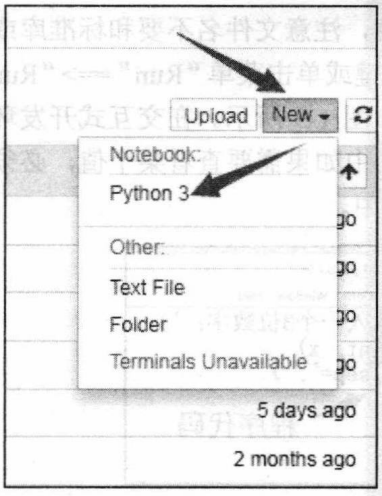

图 1-5 Jupyter Notebook 主页面右上角菜单

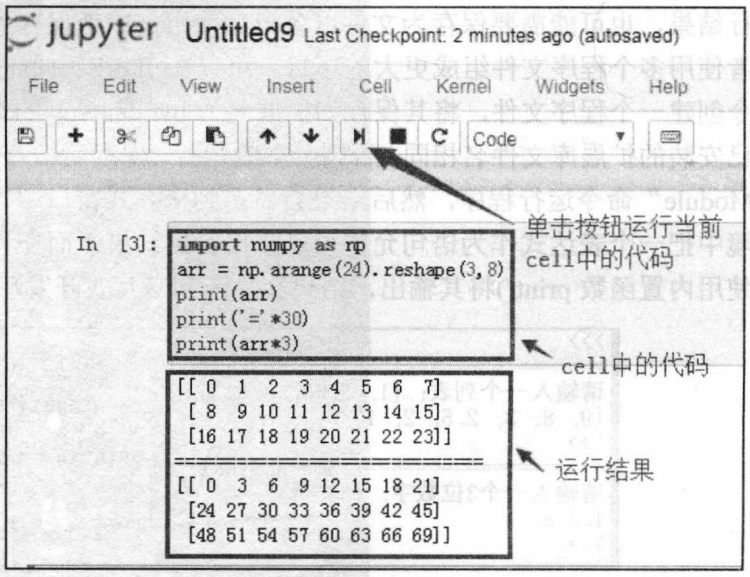

图 1-6 Jupyter Notebook 运行界面

2. Spyder

Anaconda3 自带的集成开发环境 Spyder 同时提供了交互式开发界面和程序编写与运行界

面，以及程序调试和项目管理功能，使用更加方便。在图 1-7 中，箭头 1 表示交互式运行，箭头 2 表示程序编写窗口，单击工具栏中绿色的“▶”按钮运行程序并在交互式窗口显示运行结果，如箭头 3 所示。

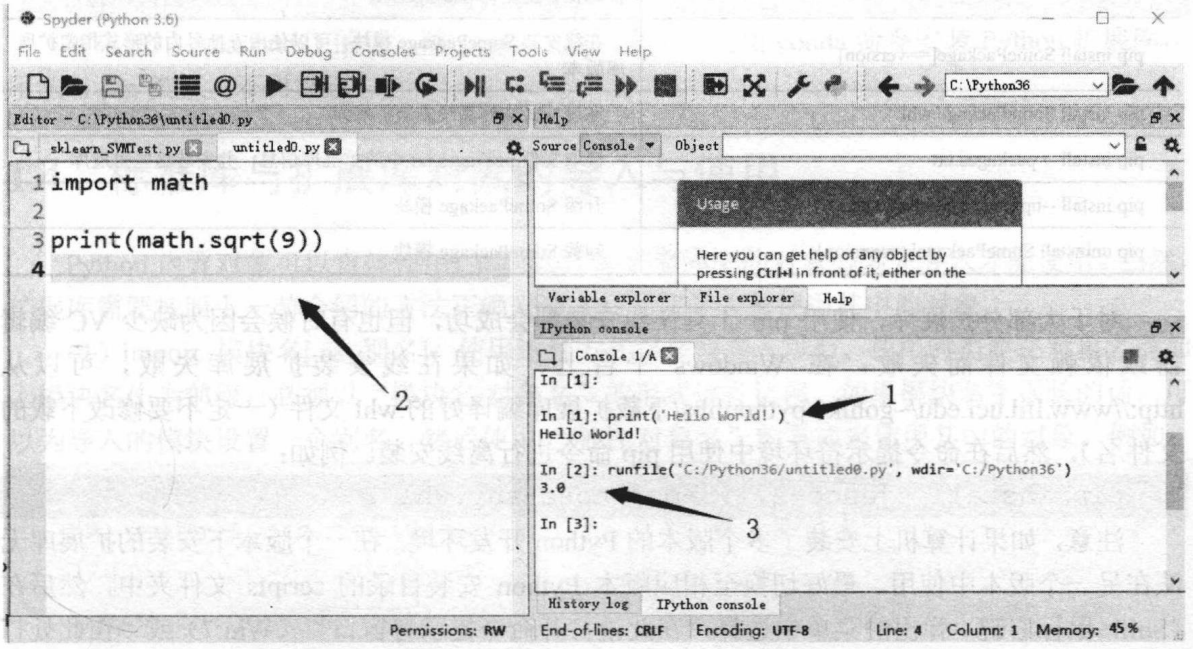

图 1-7　Spyder 运行界面

1.3　安装扩展库

在 Python 中，库或模块是指一个包含若干函数定义、类定义或常量的 Python 源程序文件。除了 math（数学模块）、random（与随机数以及随机化有关的模块）、datetime（日期时间模块）、collections（包含更多扩展版本序列的模块）、functools（与函数以及函数式编程有关的模块）、urllib（与网页内容读取以及网页地址解析有关的模块）、itertools（与序列迭代有关的模块）、re（正则表达式模块）、os.path（与文件、文件夹有关的模块）、pickle（二进制文件序列化与反序列化模块）、zlib（数据压缩模块）、hashlib（安全哈希与报文摘要模块）、threading（多线程编程模块）、socket（套接字编程模块）、tkinter（GUI 编程模块）、sqlite3（操作 SQLite 数据库的模块）、csv（读写 CSV 文件的模块）、json（读写 JSON 文件的模块）等大量内置模块和标准库之外，Python 还有 openpyxl（用于读写 Excel 文件）、python-docx（用于读写 Word 文件）、pymssql（用于操作 Microsoft SQL Server 数据库）、numpy（用于数组计算与矩阵计算）、scipy（用于科学计算）、pandas（用于数据分析）、matplotlib（用于数据可视化或科学计算可视化）、scrapy（爬虫框架）、sklearn（用于机器学习）、tensorflow（用于深度学习）等几乎渗透到所有领域的扩展库或第三方库。

在标准的 Python 安装包中，只包含了内置模块和标准库，没有包含任何扩展库，开发人员根据实际需要再安装和使用合适的扩展库。Python 自带的 pip 工具是管理扩展库的主要方式，支持 Python 扩展库的安装、升级和卸载等操作。常用 pip 命令的使用方法如表 1-1 所示。

表 1-1 常用 pip 命令的使用方法

pip 命令示例	说 明
pip freeze [>packages.txt]	列出已安装模块及其版本号，可以使用重定向符“>”把扩展库信息保存到文件 packages.txt 中
pip install SomePackage[==version]	在线安装 SomePackage 模块，可以使用方括号内的形式指定扩展库版本
pip install SomePackage.whl	通过 whl 文件离线安装扩展库
pip install -r packages.txt	读取文件 packages.txt 中的扩展库信息，并安装这些扩展库
pip install --upgrade SomePackage	升级 SomePackage 模块
pip uninstall SomePackage[==version]	卸载 SomePackage 模块

对于大部分扩展库，使用 pip 工具在线安装都会成功，但也有时候会因为缺少 VC 编辑器或依赖文件而失败。在 Windows 平台上，如果在线安装扩展库失败，可以从 http://www.lfd.uci.edu/~gohlke/pythonlibs/下载扩展库编译好的.whl 文件（一定不要修改下载的文件名），然后在命令提示符环境中使用 pip 命令进行离线安装。例如：

```
pip install Django-2.1.3-py3-none-any.whl
```

注意，如果计算机上安装了多个版本的 Python 开发环境，在一个版本下安装的扩展库无法在另一个版本中使用。最好切换至相应版本 Python 安装目录的 scripts 文件夹中，然后在〈Shift+鼠标右键〉弹出的菜单中选择“在此处打开命令提示符窗口”（Win 7）或“在此处打开 PowerShell 窗口”（Win 10）命令，进入命令提示符环境执行 pip 命令，如果要离线安装扩展库的话，也要把.whl 文件下载到相应版本的 scripts 文件夹中。

在线安装扩展库时，会有大量输出，如果不想看到这些输出，可以在执行 pip 命令时加上-q 选项表示安静模式。例如：

```
pip install openpyxl -q
```

另外，由于安装扩展库时需要从国外网站下载，速度较慢，可以使用-i 选项设置临时使用国内的镜像网站，同时使用参数--trusted-host 指定可信任主机。例如：

```
pip install -i http://pypi.douban.com/simple/django --trusted-host=pypi.douban.com
```

国内比较稳定的扩展库镜像网站如表 1-2 所示。

表 1-2 国内扩展库镜像网站

镜像地址	所属单位
https://pypi.tuna.tsinghua.edu.cn/simple/	清华大学
http://mirrors.aliyun.com/pypi/simple/	阿里云
http://pypi.douban.com/simple/	豆瓣网
https://pypi.mirrors.ustc.edu.cn/simple/	中国科技大学
http://pypi.hustunique.com/	华中理工大学
http://pypi.sdutlinux.org/	山东理工大学

如果需要了解 pip 命令更多高级用法，可以使用下面的命令查看 pip 的更多子命令：

```
pip -h
```

或者使用下面的形式查看特定子命令的更多用法：

```
pip freeze -h
pip install -h
```

如果使用 Anaconda3 的话，除了 pip 之外，也可以使用 conda 命令安装 Python 扩展库，用法与 pip 类似，不再赘述。

1.4　标准库与扩展库对象的导入与使用

Python 内置对象可以直接使用，但标准库和扩展库中的对象必须先导入才能使用。当然，扩展库需要按照上一节介绍的方法正确安装之后才能导入和使用其中的对象。

（1）import 模块名[as 别名]：使用这种方式将模块导入以后，使用时需要在对象之前加上模块名作为前缀，必须以“模块名.对象名”的形式进行访问。如果模块名字很长的话，可以为导入的模块设置一个别名，然后使用“别名.对象名”的方式来使用其中的对象。例如：

```
>>> import math                          #导入标准库 math
>>> math.factorial(6)                    #计算 6 的阶乘
720
>>> math.gcd(48, 39)                     #返回两个整数的最大公约数
3
>>> import numpy as np                   #导入扩展库 numpy，设置别名为 np
>>> np.sin([0, np.pi/4, np.pi/2, np.pi])
                                         #计算多个角度的正弦值
array([0.00000000e+00,   7.07106781e-01,   1.00000000e+00,
       1.22464680e-16])
>>> import os.path as path               #导入标准库 os.path，设置别名为 path
>>> path.isfile(r'C:\Windows\notepad.exe')
                                         #检查指定的路径是否为文件
                                         #字符串前面加字母 r 表示原始字符串
                                         #不对其中的任何字符进行转义
True
```

（2）from 模块名 import 对象名[as 别名]：使用这种方式仅导入明确指定的对象，使用时不需要使用模块名作为前缀，可以减少程序员输入的代码量。这种方式可以适当提高代码运行速度，打包时可以减小文件体积。例如：

```
>>> from random import choice, randint
>>> choice('abcdefg')                    #从字符串中随机选择一个字符
'f'
>>> randint(1, 100)                      #在 1 到 100 之间生成一个随机数
55
>>> from os.path import getsize
```