第1章

数据科学基础

数据科学是一门新兴科学，它以数据为中心，帮助我们理解数据，用数据进行创新，推动社会发展。今天数据科学的研究应用不仅限于科研人员、企业机构，针对它的教学已经拓展到大学甚至高中阶段，人们开始关注如何在工作、日常生活中应用数据科学。本章介绍数据科学的基本概念及涵盖的专业领域，重点介绍数据科学的应用实例、数据科学的工作流程，以及本书实现数据分析的工具。

1.1 数据科学概述

1.1.1 数据的力量

世界著名未来学家托夫勒曾说改变这个世界的力量有三种：暴力、知识、金钱，而如今我们的世界正在被第四种力量改变，那就是数据！

今天随着计算机技术的发展，数据正日益凸显其价值。工业、农业、服务业等各行业的行为以数据形式记录下来，人们的日常生活也被“数据化”，越来越多的政府、企业意识到数据正在成为组织最重要的资产，数据分析解读的能力成为组织的核心竞争力。数据分析帮助政府、企业、个人更好地洞察事实，改善计划和决策，反过来分析结果又影响了组织和个人的行为，甚至在一定程度上左右社会的未来。下面我们通过一些实例来认识今天数据对社会方方面面的影响。

随着互联网和信息系统的发展，政府机构汇集了医疗健康、城镇交通、义务教育、税收稽查、社会治理等各方面的数据。通过这些数据，政府能快速地获取关键、准确的信息，改进各项政策和工作，节约政府部门的治理时间、人力成本，也更新了治理思路和模式。

【例 1-1】 杭州公交借助共享单车轨迹改善公交线路。

杭州公交集团发现 286B 路公交线路，在某两站每天聚集着数百辆、最多时上千辆共享单车，杂乱地停在人行道、非机动车道甚至站台、行车道上。通过分析共享单车的出行轨迹，杭州公交集团发现了单车主要社区来源，对 286B 公交车的线路进行优化，调整了首末班时间、发车频率，将很多需要骑行到车站的乘客直接送到了家门口。新线路缓解了区域出行压力，也疏导了共享单车密集可能带来的道路隐患。

社会经济的发展和繁荣，依赖于全社会企业的总体经营状况。在企业日常运营中，每

天都产生大量的数据，对企业的运营和发展的决策起到重大作用。通过分析这些数据，企业能够正确地了解目前经营现状、及时发现存在的隐患并分析原因，进一步对未来的发展趋势进行预测，进而制定有效的计划、战略决策。

【例 1-2】 金融机构借助信用卡人群数据分析，改善信贷决策。

根据新浪整理的市场数据发现，信用卡的主流人群、活跃用户，70%是 18～35 岁的年轻人。虽然 18~24 岁的年轻人有较普遍的透支消费习惯，但透支消费能力差，收入较低且不稳定，他们的风险最高。25~35 岁的年轻人透支消费主要来源于房子、车子、孩子等刚性需求，存在长期大额信用贷款的巨大需求，且还贷能力强。数据显示，年轻男性的失信风险是女性的 1.3 倍。车主人群是无车人群信贷需求的 1.3 倍，但风险却低了 65%。所以目前金融信贷业务偏爱 25~35 岁人群、女性白领、车主等人群，为吸引这类人群制定了不同的信贷方案，拿出相应的权益和活动吸引他们信贷消费。

【例 1-3】 图像数据分析辅助放射科医生读片，提高医疗效率。

近年来，医疗诊断过程中 CT、X 片等应用日益广泛，据统计，我国医学影像数据的年增长率约为 30%，而放射科医师数量的年增长率为 4.1%。很多医疗机构与研究单位合作，基于医院历史的影像资料，利用机器学习等方法建立识别模型，自动读片进行疾病的检测，在皮肤癌、直肠癌、肺癌识别、糖尿病视网膜病变、前列腺癌、骨龄检测等方面达到甚至超过人工检测的准确率，这些疾病的检测模型需要几万至几十万正确标注后的影像资料进行训练才能达到目前的精度。相比较人工读片，机器读片比较容易继承经验知识，客观、快速地进行定性和定量分析，为医生诊断提供高效的辅助工具。

利用数据并不是政府、机构、企业的专利，每个人都能在自己的身边享受数据带来的红利。

【例 1-4】 做优秀的面包店长。

花小仙经营了一家面包房，经过几年的经营，希望自己的店能进一步成长。开业以来，花小仙细心记录了店内主要产品的相关数据，包括各种面包的销量、质量、原料数量、价格等。建立简单的回归和时序模型分析这些数据后，花小仙预测了未来半年的收益、现金流，以及加大生产所需的机器和人力成本，最终决定通过添置机器、不增加人力的方式来提高产量，整个成本控制在未来现金流内，不会导致面包店资金链出现风险。

【例 1-5】 物理实验数据分析。

小夏是大学生，大学物理实验课每次需要处理很多实验数据，撰写实验分析报告。小夏尝试数据科学方法来应对重复的数据处理过程。每次实验预习时，按照物理模式做出表格，编写分析小程序实现数据预处理、异常数据检测、数据相关性分析、曲线拟合和误差分析。实验过程中小夏只需记录数据，立刻就能得到分析结果，同时还能发现自己实验过程中的不合理数据，校正实验方法和步骤。小夏发现，他的小程序适应性很强，每次实验只需要根据实验原理，调整实验数据记录表格、物理原理公式计算函数就能满足大多数实验的分析要求。数据科学的工作方法提高了小夏物理实验的效率，当然也包括物理实验的成绩。

数据不仅是一种工具，而且是一种战略、世界观和文化，它将带来一场社会变革，每个人都应当以开放的心态、协同的精神来迎接这场变革。正如从矿物质里发现了钢铁、汽

油改变了人类的生活一样，数据也像一个矿，如何从中提炼出来提高生命质量的产品，现在才刚刚开始。“与数据的逻辑吻合，你自然会找到金子”。下面我们就开启金子的发现之旅。

1.1.2　数据科学的知识结构

数据是世界本真的原始记录，表示为零散的符号，如人的年龄、室外的温度、公园的路线图、腊梅花的图片、一段声音。数据本身并没有意义，经过组织和处理后，数据被抽象为信息，用来表示某件事物和某种场景，如冬天的公园；将数据和信息经过理解转化为一组规则来辅助决策，得到的就是知识，如基于公园的信息，给出在冬天公园的最佳观赏路线图。

数据科学（Data Science）研究的就是从数据形成知识的过程，通过假定设想、分析建模等处理分析方法，从数据中发现可使用的知识、改进关键决策过程。数据科学的最终产物是数据产品，是由数据产生的可交付物或由数据驱动的产物，表现为一种发现、预测、服务、推荐、决策、工具或系统。

数据科学虽然是新兴学科，但并不是一夜之间出现的，数据科学的研究者和从业人员继承了各个领域前辈们数十年甚至数百年的工作成果，包括统计学、计算机科学、数学、工程学及其他学科。数据科学已成为各行业发展的背后动力，迅速渗透到社会各个行业并通过高等教育传播开来。数据密集型、计算驱动的工作成为未来的热点。

今天数据科学的知识范畴主要包括专业领域、数学、计算机，可用韦恩图来表示，如图 1-1 所示。数据分析知识结构的韦恩图有众多的版本，这里给出的雪莉·帕尔默的说法。

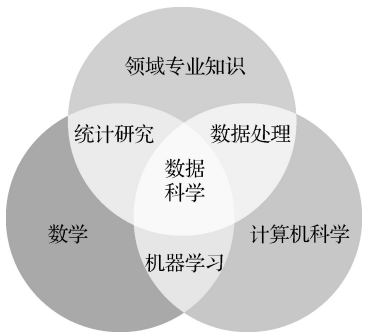


图 1-1　数据科学的韦恩图

1. 领域专长

从事数据工作的人员需要了解数据来源的业务领域，充分应用领域知识提出正确的问题。每个人都想知道如何提高销量，这确实是问题，但领域专家能问出更具体的问题，以引导实现可量化、可实现的提高。例如，使用数据集 ABC 是否可提高 XY 部门的产量？是否可以通过零售数据、天气模式数据及停车场密度数据来提高资产回报率？可以使用产品的哪些特性来增强其竞争力？这些细节问题将帮助数据分析找到行动的方向。

2. 数学

在数据科学中，数学家是团队中解决问题的人，他们能够建立概率统计模型、进行信号处理、模式识别、预测性分析。数据科学具有魔力，能在大数据集上使用精妙的数学方法，产生不可预期的洞察力。科学家研发出人工智能、模式匹配和机器学习等方法来建立

这些预测模型。

3. 计算机科学

数据科学是由计算机系统来实现的，数据科学项目需要建立正确的系统架构，包括存储、计算和网络环境，针对具体需求设计相应的技术路线，选用合适的开发平台和工具，最终实现分析目标。

1.1.3 数据科学的工作流程

数据科学是系统科学，包括研究数据理论、数据处理及数据管理等。通常我们用术语“数据分析”表示数据科学的核心工作，即面向具体应用需求，进行原始数据收集、信息准备、模式分析并形成知识、创造价值的活动。

数据分析的关键步骤包括提出分析目标，从自然界中获得一个数据集，对该数据集进行探索发现整体特性，使用统计、机器学习或数据挖掘技术进行数据实验，发现数据规律，将数据可视化、构建数据产品，可以用图 1-2 所示的流程表示。

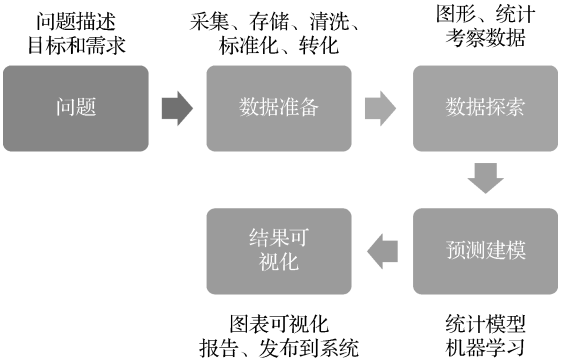


图 1-2　数据分析的关键步骤

1. 问题描述

数据科学不是因为有了数据，就针对数据进行分析，而是有需要解决的问题，才对应地搜集数据、分析数据。基于专业背景，界定问题，明确数据分析的目标和需求是数据分析项目成败的关键所在。从数据理论的角度，可将分析问题的种类分为推理性问题、描述性问题、探索性问题、预测性问题、因果问题，相关性问题等。

2. 数据准备

数据准备包括数据获取、清洗、标准化，最终转化为可供分析的数据。面向问题需求，我们可以从多种渠道采集到相关数据，如互联网爬取、业务系统生成、检测设备记录等，然后按照业务逻辑将这些形式各异的数据组织为格式化的数据，去掉其中的冗余数据、无效数据，填补缺失数据。

3. 数据探索

数据探索主要采用统计或图形化的形式来考察数据，观察数据的统计特性，数据成员之间的关联、模式等。可视化的方法能够提供数据概览，从而找到有意义的模式。数据探索过程中也会发现数据并不干净，含有重复值、缺失值或异常值，这就需要返回重新进行清洗。

4. 预测建模

根据分析目标，通过机器学习或统计方法，从数据中建立问题描述模型。选择何种方法主要取决于是分类预测问题，还是描述性问题，或是关联性分析问题。建立模型应尝试多种算法，每种算法都有相对适用的数据集，需要根据数据探索阶段获得的数据集特性来选择。因此，这个阶段另一个重要任务就是对生成的模型进行评估，尝试多种算法及各种参数设置，从而获得特定问题的相对最优解答。

5. 结果可视化

结果可视化整理分析结果，展示并将分析结果保存在应用系统中。展示的形式有多种，如报表、二维图、仪盘或信息图等。这些结果被粘贴到各种报告中，或者发布到 Web 应用系统、移动应用的页面上，形成数据产品。

一个成功的数据应用案例的核心因素不仅是分析技术方法，还在于对分析数据对象业务领域的理解，这几乎决定了案例的成败。数据科学的工作流程的每个环节都需要发挥领域知识的作用，指导分析过程走向正确的方向。

1.1.4　数据科学与大数据

近年来，大数据（Big Data）被广泛提及，人们用它来描述和定义“信息爆炸”时代产生的海量数据，通常用“4V”来反映大数据的特征。

- Volume（规模性），数据的存储与计算需要耗费海量规模的资源，如卫星收集的数据达到 32PB、新浪微博日活跃人数达到 1.65 亿人。
- Velocity（高速性），增长速度快，需要及时处理。支付宝“双 11”夜，0 点支付峰值达到 25.6 万笔/秒，上海地铁日均刷卡记录达到 2 千万次。
- Variety（多样性），数据的来源和形式多样，包括半结构化的关系数据、位置、非结构化的文本、图片、音/视频数据。信息来源大致可分为网络数据、企事业单位数据、政府数据、媒体数据等。
- Value（高价值性），大数据价值总量大，但知识密度低，需要通过数据分析有效地发现其价值。

大数据属于数据科学的范畴，大数据分析是大数据创造价值的重要途径。大数据分析遵循数据科学的工作流程，继承了数据分析的技术和方法，只是当数据量达到某种规模时，需要引入分布式、并行计算、云平台等其他技术实现大规模数据的存储、计算和传输，如

图 1-3 所示。

图 1-3　大数据分析技术

1）从底层来看，大数据需要高性能的计算架构和存储系统，如用于分布式计算的 MapReduce 计算框架、Spark 计算框架，用于大规模数据协同工作的分布式文件存储 HDFS 等。

2）大数据分析的基础是对大数据进行有效管理，为大数据高效分析提供基本的数据操作，传统的关系型数据库难以满足要求。新型数据库，如适应处理高访问负载的键值数据库、分布式大数据管理的列式存储数据库、适用于非结构化的文档数据库及社交网络和知识管理的图形数据库等，这些被统称为 NoSql 数据库。

3）传统的统计方法、机器学习方法和可视化技术在应用于大数据分析时，需要根据数据量大、数据维度高、数据缺乏结构等特性，发展出相应的数据整合、清洗、降维处理等技术，同时发展新的分析方法和技术。深度学习（深度神经网络）就是在大数据推动下演化出的有效方法，现在已广泛应用于各类数据分析领域，包括图像识别、语音处理、推荐系统等。

大数据的兴起及各领域对大数据的关注，推动了数据科学的发展，但数据科学并不局限于大数据，并不是只有大数据才具有分析价值，近百年来人们通过数值分析、统计分析等各种方法洞察世界、探索未知、促进社会进步。而今天大数据的挖掘分析，为我们提供了更强大的技术手段。

本书依据数据科学的工作流程，关注从数据中发掘知识的思维逻辑、技术方法，通过实例介绍数据探索与可视化的技术、基于机器学习的数据建模预测方法，以及数据科学在图像、序列数据、语音及自然语言等领域的应用。处理大数据额外需要的计算架构、数据存储与管理等方面的技术，本书不涉及。在大数据建模分析技术中，本书将介绍目前最重要的深度学习方法，以及在图像识别等前沿领域的应用。

思考与练习

1．结合自己的专业方向，使用互联网收集 1～2 个数据科学的应用案例。

2．收集自己的月收支和消费数据清单，分析哪些非必要开支影响了经济状况。

1.2　Python 数据分析工具

越来越多的人开始使用 Python 语言开展数据分析工作，与统计分析专业工具 R 语言和矩阵计算专业工具 Matlab 相比，Python 包含了数据分析过程需要的所有方法和工具，具有速度优势，能够支持大数据处理。Python 通过多个开源的第三方工具包来实现数据分析，能够紧跟新技术发展，已成为数据科学的首选工具。

使用 Python 实现数据分析过程，工作人员重点关注分析的技术和方法，无须耗费大量精力掌握复杂的软件编程技术，代码量少，适用于初学者，同样也适用于专家。

1.2.1　科学计算集成环境 Anaconda

Python 是一个开源的、跨平台的编程语言，官方网站提供了针对各个平台的安装包（http://www.python.org/downloads），包含基础的 Python 编程环境，以及基础的方法库。使用 Python 分析数据，需要安装相关的第三方工具包（通过 Python 的 pip 命令逐个安装）。本书推荐使用 Python 的科学计算发行版 Anaconda（开源），它是一个跨平台的版本，支持 Windows、Linux、MacOS 等平台，包括近 200 个工具包，常见的 NumPy、SciPy、pandas、Matplotlib、scikit-learn、NLTK 等库都已经包含其中，满足了数据分析的基本需求。

Anaconda 可以在官方网站中（https://www.anaconda.com/download）下载，也可以到国内的镜像网站中下载（如 https://mirrors.tuna.tsinghua.edu.cn/help/anaconda）。本书代码统一遵循 Python 3 语法，推荐安装 Anaconda3-5.0.1 及以上版本。

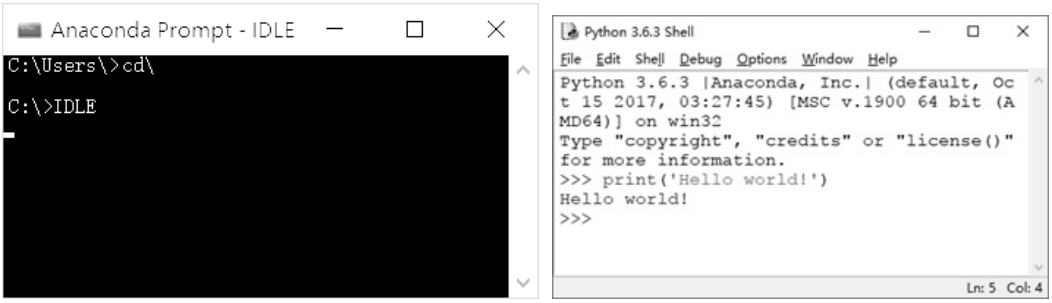
在 Windows 平台上安装完成后，在“程序”列表中将添加 Anaconda3 程序组，如图 1-4 所示，其中包含多个应用程序。Anaconda Navigator 提供第三方工具包的管理工具，Anaconda Prompt 是命令行工具，Jupyter Notebook 是交互式笔记本（详见 1.2.3 节），Spyder 是一个集成开发环境。

图 1-4　Anaconda3 程序组

1.2.2　Python 编译环境

Python 有很多功能丰富的集成开发环境，如 IDLE、Pycharm、Spyder 等，本书采用 IDLE，

它是一款轻量级的交互式解释环境，只要安装了 Python 解释器就会附带。打开 Anaconda Prompt，进入命令行界面，如图 1-5（a）所示。然后输入 IDLE 命令，即可打开 Python 的 Shell 界面，如图 1-5（b）所示。



（a）　　（b）

图 1-5　IDLE 交互式界面

IDLE 可以逐条运行代码，也可以创建、编辑 Python 源代码文件，运行完整的程序。在图 1-5 中，在命令提示符“>>>”后输入语句并回车，下一行蓝色的字体表示代码执行结果；单击“File”菜单的“Open”或“New File”即可进入源代码编辑界面，如图 1-6 所示。

图 1-6　源文件编辑与调试界面

程序编辑完成后，单击“Run”菜单的“Run Module”，即可运行解释并执行代码，代码执行的交互显示在 Shell 界面。

1.2.3　Jupyter Notebook

Jupyter Notebook 是一个基于 Web 的交互式笔记本，其主要特点是易于“讲故事”。它将程序存放在一个文件中，但可以分割成多个片段运行展示，可以实现：

- 查看算法每步运行的中间结果；
- 反复修改、运行代码片段；
- 存储中间结果，并修改；
- 展示代码成果（可以是文本、代码和图像等形式）。

在 Anaconda3 程序组中单击 Jupyter Notebook，启动操作系统默认的浏览器， 打开

Jupyter 应用程序，如图 1-7 所示。

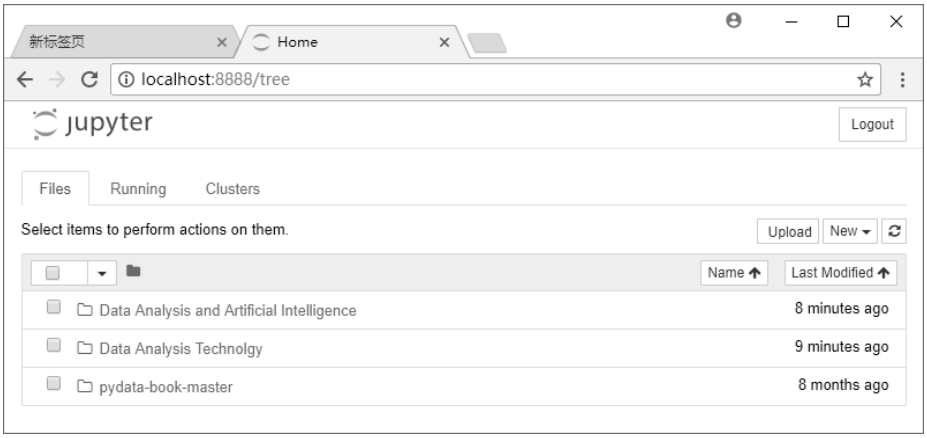


图 1-7　Jupyter Notebook Web 界面

单击“New”菜单的“Python 3”，打开一个新窗口，就可以创作自己的 Notebook 了，文件后缀名为“.ipynb”，如图 1-8 所示。窗口下部由可以编写代码的单元（cell）组成。单元“In[n]:”（n 为单元执行的序号）里面既可以存放一段文本，也可以存放一段代码。选中某个单元，单击工具栏的“ ”，即可运行该单元的代码。结果在此单元下方显示，用“Out[n]:”表示。

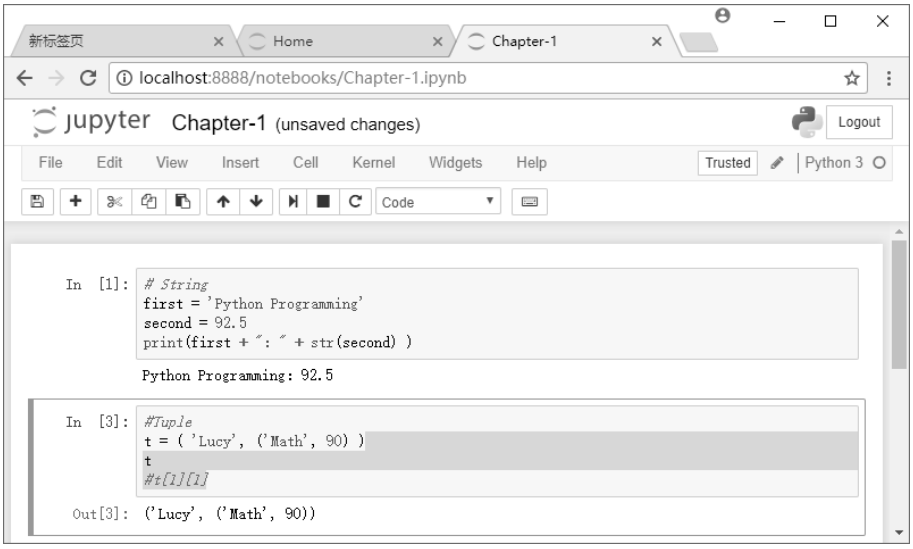


图 1-8　Jupyter Notebook 文本编辑界面

当某个单元运行后，其运行结果会被保留下来，后面的单元运行时，将继承前面的运行结果，可以访问、修改前面的变量值。

单击“File”菜单的“Rename”，可以为 Notebook 文件重新命名。

1.3 Python 语言基础

本节简要介绍 Python 3 的基本语法，主要包括后续章节所需使用的特性。

1.3.1 常用数据类型

Python 内置的常用数据类型有字符串、布尔量、元组、列表和字典。

数字（Number）包括整数、浮点数和复数类型，使用方法类似于数学计算。布尔值（Bool）有固定的表示，True 表示真，False 表示假。

```
>>> print(3+5 == 6)
False
```

下面重点介绍数据分析中常用的字符串、元组、列表和字典数据类型。

1. 字符串（string）

字符串是由一系列字符组成的数据类型，使用一对单引号'、双引号"或三引号'''表示。字符串变量的值不可以修改，任何类型的变量都可以使用内置函数 str() 转换为字符串。

```
>>> course=  'Python Programming'
>>> score = 92.5
>>> print(course + "：" + str(score) )
Python Programming：92.5
```

Python 内置了字符串常用函数，支持字符串查找、替换、比较等功能。

2. 元组（tuple）和列表（list）

元组和列表是有序的元素序列，具有相同的索引方式，每个元素可以是任意类型的数据。不同的是，元组的数据不可修改，列表数据可以修改。

元组使用一对( )将所有元素括起来，元素的数据类型可以不同，如('Wang', 32, 1.67)，元素本身也可以是元组。元组中的元素使用**变量名[索引]**来表示，索引范围[0, *n*−1]或[−*n*, −1]，如图 1-9 所示，其中 *n* 为元素个数（也称为元组长度）。

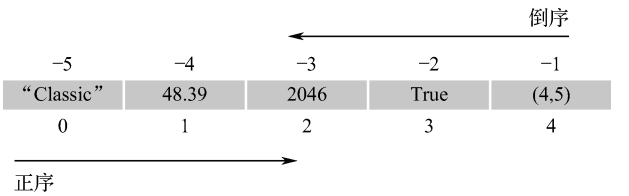


图 1-9 序列的索引

```
>>> t = ( 'Lucy', ('Math', 90) )#两个元素分别是字符串“Lucy”和元组('Math',90)
>>> t
('Lucy', ('Math', 90))
>>> t[1][1]
```

```
90
```

实际上字符串可以看作元组的特例，每个元素必须是字符的元组。

列表采用一对[ ]表示，是最灵活的序列表示形式，用来存储值需要变化的数据序列。

```
>>> ls = []
>>> ls.append(1)              #添加一个数值 1
>>> ls.append('wang')         #添加一个字符串“wang”
>>> ls
[1, 'wang']
>>> ls[0] = 2                 #修改第一个元素为数值 2
>>> ls
[2, 'wang']
```

3. 字典（dictionary）

字典是由一组“键-值对”元素组成的无序集合，字典元素的“键”具有唯一性。键和值通过冒号连接，不同键值对通过逗号隔开，如{ 'Wang':1.89, 'Li':1.76}。通过“键”，可以找到与之关联的“值”。

```
>>> d = dict(name="Lucy",age=8,hobby=("bike","game"))
>>> d
{'hobby': ('bike', 'game'), 'name': 'Lucy', 'age': 8}
>>> d["hobby"]
('bike', 'game')
```

字典数据可以通过“键”方便地添加、删除和修改。

```
>>> d["age"] = 9
>>> d                         #修改“键”对应的“值”
{'hobby': ('bike', 'game'), 'age': 9, 'name': 'Lucy'}
>>> d["gender"] = "F"
>>> d                         #添加新的“键-值对”
{'hobby': ('bike','game'), 'age':9, 'name':'Lucy', 'gender':'F'}
>>> del d['hobby']            #删除“键”及其对应的“值”
{'name': 'Lucy', 'gender': 'F', 'age': 9}
```

1.3.2　流程控制

1. 程序格式

Python 采用严格的“缩进”来表示代码的层次关系，且只能通过“缩进”表示，如图 1-10 所示。要求同一段程序内，每个层次“缩进”采用的空格数一致，否则判定为语法错误。

```
DARTS = 1000
hits = 0
for i in range(1,DARTS):
    x, y = random(), random()
    distance = sqrt(x**2 + y**2)
    if distance <= 1.0:
        hits += 1
pi = 4 *(hits/DARTS)
print("Pi = ", pi )
```

图 1-10 “缩进”表示代码层次关系

2. 注释

Python 的注释语句有两种形式：单行注释以“#”开头，多行注释用一组"""括起来。

```
#This is the comment

"""
This is a multiline comment
In Python
"""
```

3. 输入和输出语句

Python 使用 input 语句将键盘输入以单个字符串保存变量，print 语句实现屏幕显示。

```
s = input("姓名和年龄: ")
name,age = s.split(",")                          #用“,”切分字符串
print(name,age)                                  #屏幕输出
print("name:{}, age:{}".format(name,age))        #格式化输出
```

4. 分支结构

Python 支持单分支结构、双分支结构和多分支结构，基本格式如下。

```
if t>100:
    s = 20 + 0.4*(t-100)
elif t>50:
    s = 10 + 0.2*(t-50)
else:
    s = 10
```

代码依次计算 if、elif 后面的表达式，执行第一个结果为真的表达式对应的分支语句。如果没有任何一个表达式结果为真，则执行 else 对应的语句。

5. 循环结构

Python 提供两种循环语句，for 和 while。

1）for 循环在循环代码重复运行过程中，循环变量根据给定的序列依次赋值。

```
s = 0
for i in [1,3,5,7,9]:
    s += i
```

代码 for 语句循环 5 次，变量 i 依次被赋值 1、3、5、7、9，并被累加到变量 s 上。循环结束后，s 的值为 25。

通常可以使用 **range(start, end, step)**函数生成指定的数字序列，函数按照步长 step 在范围[start, end-1]内生成等差序列，start 默认为 0，step 默认为 1。

```
for i in range(0,10,2):
    print(i)
```

代码依次输出整数 0、2、4、6、8。

2）while 语句判断表达式的结果，如果为 True 继续循环，否则中止。

```
sum = 0
x = input("Input a number (<Enter> '' to quit): ")
while x != "":
    sum = sum + eval(x)
    x = input("Input a number (<Enter> '' to quit): ")
```

程序判断用户输入的内容是否为空字符串，为空则循环中止，否则计算累加和并等待再次输入。

1.3.3　函数和方法库

1. Python 内置函数

Python 提供大量的内置函数，无须说明，可直接使用。如 input 函数、range 函数等，但大部分的第三方库（library）或包（package）并没有被加载到解释器中，因此在使用时需要先导入，Python 提供 3 种导入形式。

1）直接导入整个方法库或包，调用时需要加上包名。

```
>>> import math                 #导入 math 包
>>> math.sqrt(5)
2.23606797749979
```

2）导入方法库中某个函数，调用时直接使用函数名。

```
>>> from math import sqrt       #从 math 包导入 sqrt 函数
```

```
>>> sqrt(5)
2.23606797749979
```

3）导入方法库中某个类或函数并重命名，调用时使用临时替代名。

```
>>> from math import sqrt as sq      #从 math 包导入 sqrt 函数，重命名为 sq
>>> sq(5)
2.23606797749979
```

2. Python 自定义函数

Python 使用关键字 def 定义函数，函数定义时，变量类型无须说明，同时可以在参数列表的最后定义多个带有默认值的参数。函数调用时，具有默认值的形参，可以不传实参。

```
>>> def say(message, times = 1):
    print (message * times)
>>> say( 'Hello' )
Hello
>>> say( 'World', 5 )
WorldWorldWorldWorldWorld
```

思考与练习

1．查阅资料，编写 Python 代码实现列表和字典元素的遍历输出。

2．使用 Jupyter Notebook，将练习 1 的程序保存在.ipynb 文件中。

综合练习题

1．在个人计算机上下载 Anaconda 科学计算工具包，并正确安装。

2．编写 Python 程序实现功能：从键盘输入若干学生的姓名，保存在字符串列表中。输入某个学生的名字，检索是否已保存列表中。

3．编写 Python 程序实现功能：使用字典记录学生的姓名及对应身高值，输入任意学生的姓名，查找并显示所有高于此身高值的学生信息。