项目 1　认知财经大数据

信息技术（IT）与经济社会的深度融合带动了数据的飞速膨胀，数据已晋升为国家的核心战略资源之一。大数据正逐渐对全世界的生产、流通、分配、消费行为，以及经济运行体系、社会生活模式和国家治理效能产生深远的影响。

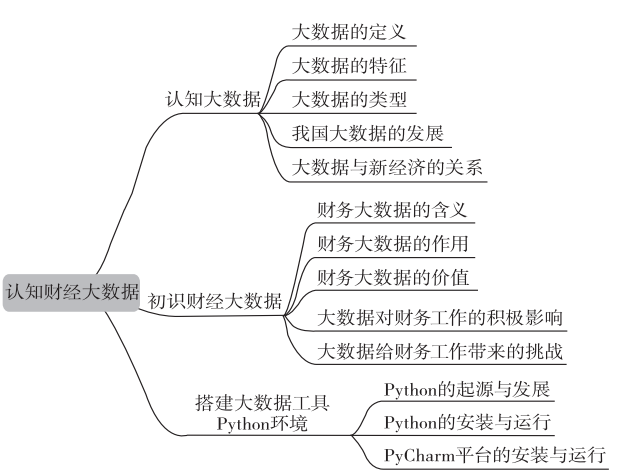
在当今信息爆炸的时代，财经大数据已经成为企业决策、市场分析、投资评估的重要依据。财经大数据是指与财经领域相关的海量数据，包括但不限于股票、债券、商品、外汇、宏观经济指标等。这些数据不仅数量庞大，而且种类繁多，涵盖了金融市场的各个方面。通过财经大数据，我们可以深入挖掘市场动态、预测未来趋势，为决策提供有力的支持。

此外，财经大数据的应用范围非常广。除了传统的金融行业，还包括电商、物流、制造业等领域。例如，电商平台可以通过用户购买行为和偏好分析，进行精准营销和个性化推荐；物流企业可以利用大数据优化运输路线和配送方案，提高运输效率。

项目提要

本项目主要介绍大数据的含义、基本特征、发展历程与趋势；财务大数据的含义、作用、价值和对财务工作的挑战；大数据工具 Python 编程语言运行环境搭建等。

项目思维导图



建议学时

4 学时。

任务 1－1　认知大数据

情境导入

张明很喜欢听音乐，他发现软件会自动推荐一些歌曲，而这些歌曲的风格大多与他平时常听的类似，非常贴近他喜欢的类型；购物软件也是如此，张明喜欢的电子产品、运动品牌及书籍的相关链接，也都出现在页面最显眼的地方。

那么，这些软件的精准推荐和大数据是不是有联系呢？

任务目标

知识目标：

1. 了解大数据相关基础知识：定义、特征、类型、思维。

2. 了解大数据的应用场景。

技能目标：

掌握大数据的基本用途。

素养目标：

通过介绍我国大数据的发展，了解我国先进的大数据技术，增强民族自豪感和对祖国的热爱之情。

建议学时

2 学时。

相关知识

一、大数据的定义

大数据是指无法在一定时间内用常规软件工具对其内容进行抓取、管理和处理的数据集合。当今社会高速发展，信息技术愈加发达。随着“云时代”的来临，大数据越来越受到人们的关注。大数据已成为各国政府和企业的重要战略资源，就像

计算机和互联网一样，其即将成为新一轮的技术革命的标志之一。大数据不仅仅是信息时代的产物，更是信息产业持续高速增长的新引擎。各行各业的决策正在由传统的“业务驱动”转变为“数据驱动”。

视频1-1

二、大数据的特征

虽然处理超过单个计算机的计算能力或存储数据的问题已经比较普遍，但近年来这种类型计算的普遍性、规模和价值已经大大扩展。大数据的确切定义很难界定，因为项目、供应商、从业者和商业专业人士使用它的方式完全不同。考虑到这一点，一般来说，大数据具备如下特征（简称4V特性）。

（一）数据量大

大数据的特征首先就体现为“大”，存储单位从过去的GB到TB，乃至现在的PB、EB级别。1 PB等于1 024 TB，1 TB等于1 024 GB，那么1 PB等于1 024 × 1 024个GB的数据。随着信息技术的高速发展，数据开始爆发性增长。社交网络（微博、推特、脸书）、移动网络、各种智能工具/服务工具等，都成为数据的来源。

淘宝网近4亿的会员每天产生的商品交易数据约20 TB，脸书约10亿的用户每天产生的日志数据超过300 TB。如此大规模的数据，迫切需要智能的算法、强大的数据处理平台和新的数据处理技术来统计、分析、预测和实时处理。

（二）数据类型繁多

如果只有单一的数据，那么这些数据几乎没有价值，如只有单一的个人数据或者单一的用户提交数据，这些数据还不能称为大数据。广泛的数据来源，决定了大数据形式的多样性。比如当前的上网用户中，每个人的年龄、学历、爱好、性格等特征都不一样。如果扩展到全世界，大数据的多样性会更强，每个地区以及每个时间段都会存在各种各样的数据。任何形式的数据都可以产生作用，目前应用最广泛的就是推荐系统，如淘宝、网易云音乐、今日头条等，这些平台都会通过对用户的日志数据进行分析，从而进一步推荐用户喜欢的东西。日志数据是结构化明显的数据，还有一些数据结构化不明显，如图片、音频、视频等，这些数据因果关系弱，就需要人工对其进行标注。

（三）处理速度快

大数据通过算法对数据逻辑处理的速度非常快，可从各种类型的数据中快速获得高价值的信息，这一点也和传统的数据挖掘技术有着本质的不同。

大数据的产生非常迅速，主要通过互联网传输。生活中每个人都离不开互联网，

也就是说每个人每天都在向大数据提供大量的资料，并且这些数据是需要及时处理的，因为花费大量资本去存储作用较小的历史数据是非常不划算的。对于一个平台而言，可能只有短期的数据存储，如过去几天或一个月，而远程数据则需要及时清理，以免造成巨大的损失。

基于这种情况，大数据对处理速度有非常严格的要求，服务器中大量的资源都用于处理和计算数据，很多平台都需要做到实时分析。数据无时无刻不在产生，谁的速度更快，谁就有优势。

（四）价值密度低

相比于传统的数据，大数据最大的价值在于通过从大量不相关的各种类型的数据中，挖掘出对未来趋势与模式预测分析有价值的数据，并通过机器学习方法、人工智能方法或数据挖掘方法深度分析，发现新规律和新知识。

如果有 1 PB 以上的全国所有 20～35 岁年轻人的上网数据，通过分析这些数据，就可以知道这些人的行为习惯、爱好等，进而指导产品的发展方向，那么这些数据就有了商业价值。如果有全国几百万病人的数据，根据这些数据进行疾病的预测与分析，这些数据同样具有价值。大数据在诸如农业、金融、医疗、物流、教育等各个领域中得以广泛应用，最终达到改善社会治理、提高生产效率、推进科学研究的效果。

三、大数据的类型

（一）结构化数据

结构化数据是遵循某种严格架构的数据，因此所有数据都具有相同的字段或属性。共享架构允许使用 SQL（结构化查询语言）等查询语言轻松搜索此类数据。通俗来讲就是其能够用数据或统一的结构加以表示，如数字、符号。学生信息表如表 1－1 所示。

表 1－1　学生信息表

学号	姓名	性别	年龄	考试成绩
95001	张三	男	21	88
95002	李四	男	22	95
95003	王梅	女	22	73
95004	林莉	女	21	96

（二）半结构化数据

所谓半结构化数据，就是介于结构化数据和非结构化数据之间的数据，如 XML

（可扩展标记语言）、HTML（超文本标记语言）文档等。它一般是自描述的，数据的结构和内容混在一起，没有明显的区分。

（三）非结构化数据

非结构化数据指的是其字段长度可变，每个字段的记录可以由可重复或不可重复的子字段组成的数据。通俗来讲就是其无法用数字或统一的结构表示，如文本、图像、声音、网页等。

四、我国大数据的发展

视频1-2

大数据对全球生产、流通、分配、消费活动及经济运行机制、社会生活方式和国家治理能力等方面产生越来越深远的影响。早在2015年10月26日至29日，中国共产党第十八届中央委员会第五次全体会议上，“十三五”规划建议就提出实施国家大数据战略，旨在全面推进我国大数据发展和应用，加快建设数据强国，推动数据资源开放共享，释放技术红利、制度红利和创新红利，促进经济转型升级。至此，大数据战略上升为国家战略。

习近平总书记在中共中央政治局第二次集体学习时强调，大数据发展日新月异，我们应该审时度势、精心谋划、超前布局、力争主动，深入了解大数据发展现状和趋势及其对经济社会发展的影响，分析我国大数据发展取得的成绩和存在的问题，推动实施国家大数据战略，加快完善数字基础设施，推进数据资源整合和开放共享，保障数据安全，加快建设数字中国，更好服务我国经济社会发展和人民生活改善。作为人口大国和制造大国，我国数据产生能力巨大，大数据资源极为丰富。随着数字中国建设的推进，各行业的数据资源采集、应用能力不断提升，将会导致更快、更多的数据积累。我国的数据总量在过去的几年里一直保持迅猛的增长势头。据统计，截至2023年底，我国数据总量已经以年均超过50%的速度持续增长。这一增速不仅彰显了我国数字经济的蓬勃发展，也反映了社会各界对大数据、云计算等新一代信息技术的广泛应用与高度依赖。

与此同时，全球超大规模数据中心的数量也在持续增长。相关报告显示，截至2023年年底，全球主要云和互联网服务公司运营的超大规模数据中心总数已经增至1 200多个，相比2020年底的597个有了显著增长，几乎是2015年的4倍。这一增长趋势反映了全球云计算和互联网行业的蓬勃发展与不断扩张，也带动了相关产业链的发展和创新。

在中国，随着数字经济的不断深入发展，超大规模数据中心的建设和运营也日

益成为行业关注的焦点。众多云和互联网服务公司纷纷加大投入，扩大数据中心规模，以满足不断增长的数据存储和处理需求。同时，政府也出台了一系列政策措施，推动数字经济和云计算产业的健康发展，为超大规模数据中心的建设和运营提供了有力支持。

展望未来，随着数字技术的不断创新和应用场景的不断拓展，我国数据总量和全球超大规模数据中心的数量还将保持快速增长的态势。这将为我国数字经济的持续繁荣和全球云计算产业的进一步发展注入新的动力。

五、大数据与新经济的关系

（一）新经济概述

新经济是指新的经济形态。社会主导产业形态的差异决定了社会经济形态的差异。在不同的历史时期，新经济有不同的内涵。目前，新经济是指创新性知识在知识中占主导、创新型产业成为产业领导者的智能经济形式。

大数据、云计算、无人机、3D 打印、虚拟现实、人工智能……日新月异、层出不穷的新技术、新业态、新产品，引领着未来经济发展的方向。现在越来越多的行业发生了翻天覆地的变化，大数据对于经济社会与人们日常生活的影响深入各个层面，社会对于数据处理能力的需求急剧增长。“新经济”也由此诞生。

（二）新经济的标志

新经济的基本特征与整个人类社会环境发生的深刻变化相对应，呈现给人们的是一个全新的经济时代。新经济时代的主要标志有以下几个。

1. 信息化和网络的快速发展

自 20 世纪以来，计算机、互联网和光纤的出现将整个世界带入信息时代。人们可以在世界任何地方了解世界任何时刻发生的事件，实现“不出家门的交流”和参与，这种交流的手段和方法也越来越简洁、透明。

2. 传统交通运输业大进步

现代化的交通运输业深度融合了数字化与智能化技术，不仅重塑了传统运输模式，更构建起高效、韧性、可持续的现代流通网络，有力地支撑了新经济的高效运转与全球互联。

3. 经济呈现全球一体化趋势

企业和自然人可以在全球范围内寻求自己的市场，集中表现为：市场全球化，即需求市场向世界上任何企业和自然人开放；资源配置全球化，即人们在选择配置

资源时，可以利用自己的实力和嗅觉，在全球范围内选择各种自己认可的资源，而不再局限于自己的国家和地区，从而提高自己的配置效率。竞争规则国际化最明显的是，大多数国家和地区已经加入世界贸易组织（WTO），并承认和适用其竞争规则。

（三）新经济的影响

新经济时代的出现不仅给各国的经济发展带来新的机遇，也给经济不发达国家的企业带来新的挑战。事实上，历次经济技术革命都在资源配置的手段、方式和效率上出现了巨大的变化，对人们的生活方式产生了深远的影响。

新经济的内涵包含两个方面：一方面，需要创新的技术，这是经济发展的核心动力；另一方面，创新的技术一定要与实体经济相结合，进而产生新的业态和新的生产方式。创新的技术带动行业和产业发展，新经济必然是围绕创新并引领时代发展的。

当今，在全球科学技术迅猛发展、国际政治经济格局剧烈变化、我国改革开放不断深化的新时代背景下，传统商科教育面临新的挑战。传统的商科课程设计是按照工具型人才培养标准的教育理念来进行，基于亚当·斯密（Adam Smith）的劳动分工理论，强调各个科目由单一、独特的内容组成，各学科都相对独立、封闭、自成体系，这样的设计倾向于割裂知识，割裂了整体的事物，使学生的理解力有了断层。

（四）新经济背景下的商科教育

2018 年，全国教育大会的召开，为新工科、新医科、新农科、新文科以及国际认证工作的推进提供了重要支持，对我国高等教育领域产生了深远的影响。作为高等教育中与社会发展、市场需求结合最紧密的领域，新商科也正在全国高校相关学院的努力下，呼之欲出。新商科，是在现有商科发展的基础上，回应科技、社会、经济所带来的挑战。

新商科是与传统商科对应的一个概念，是顺应经济社会发展的需要产生的商科教育模式。传统商科是培养“商业技术人才”的。学习财务管理专业的学生通常将自身定位为财务技术人员，而学习人力资源管理专业的学生则将自身局限于人力资源技术方面的人才。但是，随着时代的进步，仅仅关注财务知识或人力资源管理知识本身已经解决不了问题，还需要进一步了解行业发展现状甚至是国际、国内市场的竞争态势。

2020 年年末，教育部职业教育与成人教育司要求各省级教育行政部门根据现行专业目录，同时参考新版职业教育专业目录（征求意见稿），指导学校主动适应科

技革命和产业变革要求，密切关注各行业“十四五”期间发展趋势，充分对接新经济、新技术、新职业，分析产业新业态、新模式、新职业场景，以“信息技术 +”升级传统专业，及时发展数字经济催生的新兴专业、“一老一小”等民生领域紧缺急需专业，适应各地、各行业对技术技能人才培养的需要，适应学生全面可持续发展的需要，统筹考虑专业设置与人才培养方案制订，提高人才培养质量、实现更好就业。例如原财务管理专业将更名为大数据与财务管理。

2022 年，习近平总书记在党的二十大报告中强调，要实施科教兴国战略，强化现代化建设人才支撑。教育、科技、人才是全面建设社会主义现代化国家的基础性、战略性的支撑。这是党中央首次将教育、科技、人才三大战略一体规划、系统部署，充分体现了党中央对教育、科技、人才三者内在逻辑和发展规律的深刻把握，充分体现了党和国家对高等教育和人才培养的高度重视。

随着消费升级、互联网通信、大数据、云计算、人工智能、共享经济和商业的 3.0 时代的到来，我国的商贸服务业已经实现了从传统商业实体店到互联网电商再到互联网线上加线下的转型升级。新时代创新已然成为创业服务的创新发展的主旋律。新商科要根据实体经济供给侧的需求，走市场化、企业化的合作之路。

新一轮的科技革命和产业革命正在进行，互联网、云计算、大数据等新兴技术与模式正深刻改变人们的思维、生产、学习方式。共同探讨、支持新商科人才培养事业的发展，共建现代学习体系，培养大批创新人才，已经成为应对诸多复杂挑战、实现可持续发展的关键。

技能训练

举例说明大数据国家战略对相关领域产生的影响。

任务 1－2　初识财经大数据

情境导入

大数据的发展离不开 IT 产业和互联网技术的蓬勃发展。大数据正在以不可阻挡的磅礴气势，与现在最新的科技进步技术，如虚拟现实技术、增强现实技术、纳米技术、生物工程、移动平台应用等一起，揭开人类新世纪的序幕。同样，大数据技术已经渗透到我们每个人的日常生活之中，也渗透到了财经大数据分析中，谁都无法回避。了解大数据发展及其在财经领域的应用，为理解大数据技术形成基础。大

数据时代已经悄然来到我们身边，它提供了无法抵御的全媒体、云计算、虚拟仿真环境和不可或缺的网络服务。大数据让人类对一切事物的认识回归本源，影响了人们生活的各个领域，也影响了经济生活、政治博弈、社会管理、文化教育科研、医疗、保险、休闲等行业。那么，大数据技术在财经领域有哪些应用呢？

任务目标

知识目标：

1. 了解财务大数据的含义与作用。

2. 了解大数据给财务工作带来的机遇和挑战。

技能目标：

能够阐释财务大数据的典型应用场景。

素养目标：

培养学生具备基本的数据素养，为企业数字化运营提供数据阅读、操作、分析和讨论的基本素质支撑。

建议学时

1 学时。

相关知识

一、财务大数据的含义

财务大数据是“财务”和“大数据”的有效结合，是依托海量结构化和非结构化的数据，利用大数据技术对数据进行分析，提炼出能辅助企业进行战略决策的有用信息，使数据成为真正有价值的企业数字资产的统称。

二、财务大数据的作用

传统财务数据以财务报告数据为主，包括资产负债表、利润表、现金流量表、股东权益变动表以及报表附注等相关的财务数据。

大数据给企业带来了更大的风险与挑战，大数据不仅扩大了企业财务数据的范畴，而且也对企业财务数据的处理、分析及反馈提出了更高的要求。财务大数据除了涵盖传统的财务报告数据之外，还包含宏观数据、行业数据以及企业供应链等相关数据。

随着数字技术的迅猛进步，各行各业已经能够处理庞大且复杂的数据资源，并

逐步实现了高效的数据利用。对于企业而言，借助大数据及其关键技术形成财务大数据，可以赋能企业财务管理网络化、数据化和智能化。对于企业会计而言，利用大数据、人工智能等技术的支持，能够使预算、核算及决算工作更加快捷地完成，同时也有助于财务登记、审核以及财务档案等工作实现信息化，提升财务会计的工作效率。

三、财务大数据的价值

（一）助推税务服务精细化

数据分析在优化政策宣传和辅导上有广阔的应用场景。税务局通过大数据分析对关键时间点和相应企业进行匹配，在业务需求发生前主动、针对性地开展辅导和服务，帮助企业解决涉税难题，主动为纳税人提供政策辅导、申报指导、加速退税审批等服务，大数据助推税务服务精细化。

政策机器人实现了“政策红利账本、个性政策匹配、税费综合计算”三个维度的精准服务，能够自动抓取纳税人基础信息，通过系统智能计算，对纳税人和缴费人进行画像，通过“数据 + 规则 + 人工智能”的处理，建立一体化的“识别—分析—处理—结果”的“政策找人”流程，分类、分时、分层向法人、财务人员、办税人员推送，改变了需要纳税人自行逐项匹配政策适用的情况，通过立体式、精细化辅导确保减税降费利好政策准确直达企业。

不断提升办税效率，提供更好、更优、更快的纳税服务，是税务部门近年来努力的目标之一。除了利用数据分析对纳税人进行“画像”，税务部门还积极探索利用数据分析助力优化办税服务厅设置，进而提升纳税人办税体验。

税务部门利用现有大数据分析工具和平台，对后台和第三方数据内容进行采集、抽象、分析和包装整合，精准定位纳税人办税行为和办税需求，建立“数据—行为—需求—服务”链条，为前台开展税收服务和征管提供决策依据，实现后台数据资源到前台业务能力的转化。以申报业务为例，“智数小屋”通过申报征收指标模型分析形成高频进厅纳税人名单，安排纳税服务专员重点关注和辅导此类纳税人。数字化应用 RPA（机器人流程自动化）机器人和“智税精灵”通过大数据智能画像，有效提高了无纸化办理的效率。

（二）防控金融风险

2022—2023 年，大数据技术在防控金融风险方面的应用迎来了前所未有的关注和热潮。随着金融市场的不断发展和复杂化，传统的风险管理方法已经难以满足现代金融机构的需求。而大数据技术的出现，为金融机构提供了一种全新的视角和工

具，帮助它们更好地识别、评估和控制风险，从而提高业务效率和客户满意度。下面，将以汇丰银行、蚂蚁集团和京东金融为例，详细探讨大数据在防控金融风险方面的应用及其价值。

汇丰银行作为全球领先的金融机构之一，其在大数据防控金融风险方面的实践值得借鉴。面对日益严峻的金融欺诈风险，汇丰银行投入大量资源，利用大数据技术构建了一套全球业务网络的防欺诈管理系统。该系统通过收集和分析全球范围内的大量交易数据，运用先进的算法和模型，能够迅速识别和发现潜在的欺诈行为。一旦系统检测到可疑交易，就会立即触发警报，并采取紧急措施进行防范，从而有效地降低了欺诈风险。

这一防欺诈管理系统的建立，不仅提升了汇丰银行的风险防控能力，还极大地提升了客户满意度和业务效率。客户在进行交易时，能够感受到银行对于交易安全的高度重视和严密保障，这无疑增强了客户对汇丰银行的信任感和忠诚度。同时，由于系统能够实时监测和预警潜在风险，汇丰银行的业务效率也得到了显著提升。银行能够更快地处理交易，减少因风险事件而导致的业务延误和损失。此外，该系统还能够为银行提供有关客户行为和市场趋势的宝贵洞察，帮助银行制定更为精准的市场战略和产品创新。

与汇丰银行相似，蚂蚁集团作为中国领先的金融科技公司，也积极利用大数据技术进行风险防控。蚂蚁集团深知大数据技术对于金融风险管理的重要性，因此投入大量资源进行技术研发和创新。其利用大数据技术对用户行为、交易数据等进行分析和挖掘，构建了一套完善的风险管理体系。通过实时监测和预警，蚂蚁集团能够及时发现潜在风险，并采取相应措施进行防范和控制。这为企业提供了更加安全、高效的金融服务。

蚂蚁集团的风险管理体系不仅关注传统的信贷风险，还广泛涉及市场风险、操作风险等多个领域。通过大数据技术对用户数据的深度挖掘和分析，蚂蚁集团能够更准确地评估用户的信用状况和还款能力，从而为用户提供更加个性化的金融产品和服务。同时，该体系还能够预测潜在的风险点，提前进行风险控制和防范。这大大降低了蚂蚁集团的风险敞口，也为其赢得了广泛的市场认可和良好的口碑。

除了汇丰银行和蚂蚁集团外，京东金融作为中国领先的电商金融平台，也在大数据防控金融风险方面取得了显著成果。京东金融通过收集和分析用户购物、支付等数据，能够全面评估用户的信用状况和还款能力。这使京东金融能够为用户提供更加精准的金融服务，如个性化的贷款额度、优惠的利率等。同时，京东金融还利用大数据技术对潜在风险进行预测和防范，确保金融业务的稳健发展。

这些企业的实例表明，大数据在防控金融风险方面具有巨大的潜力和价值。通过利用大数据技术，企业能够更好地识别、评估和控制风险，提高业务效率和客户满意度。大数据技术的广泛应用不仅有助于提升金融机构的风险管理水平，还有助于推动整个金融行业的创新和发展。

未来，随着大数据技术的不断发展和完善，相信会有更多企业加入大数据防控金融风险的行列中来。随着技术的不断进步和应用场景的不断拓展，大数据在防控金融风险方面的应用将会更加深入和广泛。同时，我们也应该看到，大数据技术的应用也面临着一些挑战和问题，如数据隐私保护、算法公正性等。因此，在推动大数据应用的同时，我们也需要关注这些挑战和问题，确保大数据技术的健康发展和应用效果的最大化。

（三）助力宏观经济分析

传统宏观经济分析依赖于抽样调查和汇总数据，存在时效性差、颗粒度粗、维度单一等局限。大数据涵盖领域广、体量庞大、类型多样，为宏观经济分析提供了前所未有的海量数据支持，给宏观经济分析带来了革命性的变革。

1. 区域经济状态分析

通过分析区域内所有企业多方面的数据，建立区域经济状态评价指标体系，实时动态地监测区域经济发展状况，对不良状况及时预警，从而确保经济健康平稳地发展。

2. 区域经济发展趋势分析

在实时监测区域经济状态的基础之上，建立各个评估指标的短、中、长期预测模型，对经济状况的发展趋势进行预测，从而做到提前发现机会或问题，提前布局或采取措施。

3. 辅助决策

通过构建相应的经济指标体系和大数据分析模型，对区域经济按区域、行业、个体等进行多维度分析，用时间轴的方式呈现不同角度的经济趋势图，为相关事务负责人制定重大决策提供翔实、可靠的数据支撑。

4. 构建社会征信体系

通过对市场监督管理、法院、海关等多个政府机构提供的关于法人或法人代表的数据进行分析挖掘，建立一套完善的法人和自然人等社会主体信用评估体系，提供有效的征信评估服务，为金融服务决策提供有力支持，为政府执政提供科学的

依据。

5. 促进创新创业

打造适合区域自身的大数据创新创业公共服务平台，将有效地整合和集成一系列包括人、财、物的相关互联、相关作用、相互影响的资源。同时将整合的互联网数据和允许开放的政务数据向大众公开，充分利用社会资源，发挥大众智慧，鼓励创新创业，让大众来分析数据，挖掘数据价值，促进区域经济发展。

大数据正在重塑宏观经济分析的方式，基于已汇集及拟汇集的数据，利用大数据技术分析区域经济的现状及发展趋势，为政府、企业和个人提供更加全面、及时、精准的决策支持。

四、大数据对财务工作的积极影响

（一）提高传统财务工作效率，重复性工作将被系统取代

很多企业都有制作财务日报、周报、月报的重复性报表需求，通过计算机系统可实时展现自动更新的财务分析报表，并做到定时、定点推送。在大数据背景下，计算机系统会把海量信息集合，按照程序给企业提供全面的分析报告，大量的重复工作将由程序自动处理，数据的收集、处理、分析速度不断加快，工作效率大大提升。

（二）促进传统财务流程重组，发展财务共享中心

大数据背景下，企业可以共享强大的数据，发现、剖析、预警数据中所隐藏的问题，及时应对业务中的风险，使用大数据技术对数据进行筛选、切割、排序、汇总等，自主灵活地达成期望的数据处理结果。企业也可以对数据进行分析，直观地发现增长点。

（三）预算管理更精准，财务会计走向决策高度

大数据能够使财务人员更精准地制定预算管理，通过大数据事前预测、事中把控、事后分析，全程参与业务，挖掘财会数据价值，为领导层提供决策依据。

（四）提升财务数据质量，降低财务风险

在大数据时代，财务人员与审计人员可以获得市场上多方面来源的信息，相互验证，公允价值等数据变得更加准确、透明和公开，实现财务报表的编制在遵循谨慎性原则的同时更加公允。

在传统模式下，企业无法有效收集和处理经营情况、宏观环境、外部市场和竞争对手动向等财务工作需要的内外部信息，存在信息不对称带来的经营风险和财务风险；在大数据时代，大数据技术可以为财务工作提供及时、准确、全面的信息支

持，有助于企业应对内外部可能发生的变化，确保将财务风险降到最低。

综上所述，在智能财务时代，通过大数据、人工智能等相关技术的应用，财务人员可以将更加细致的原始数据交由机器自动逐笔处理，而不用人工合并录入系统；财务人员可以利用智能机器从经济业务中提取到更加多维的数据，而不只包括简单辅助核算项的数据；财务人员可以利用大数据技术整合更加多元的公司内部和外部数据，而不仅仅是常规的部分内部数据。

五、大数据给财务工作带来的挑战

（一）对财务人员能力提出更高要求

（1）数据采集能力。大数据时代，各类业务信息都能及时地在网上展示并更新，财务人员必须通过互联网、智能系统或云端等提取信息，对专业知识与信息技术提出更高要求。

（2）数据分析能力。技术进步和大数据的普遍使用增加了财务分析的难度，对财务人员处理半结构化、非结构化信息的能力提出了更高要求。

（3）数据呈现能力。财务转型要求财务人员利用财务信息优势更多地参与决策，需要财务人员选择合适的可视化呈现技术，有效传递财务信息。

（二）增加信息保密难度

（1）数据集成。非结构化数据处理常用的蚁群算法存在固有缺陷，导致大数据技术在数据集成的拓扑领域面临保密性的挑战。

（2）数据控制。数据控制依赖各类交易密码，而密码本质也是数据的组合，只是阻挡盗窃的暂时性行为，未超出技术本身存在的惰性。

（三）给财务安全带来考验

（1）外部黑客攻击。现阶段财务软件侧重功能开发，较少顾及安全问题，面临黑客攻击后出现系统瘫痪、数据丢失等财务风险。

（2）内部能力不足。财务人员获取与处理信息的能力参差不齐，任何一个环节的失误都会对财务安全造成威胁；财务转型过程中，很可能存在内部控制的漏洞，企业内控存在失效的可能性。

技能训练

1. 在互联网中对财务大数据应用现状进行调研。

2. 会计信息系统中的科目余额表是非结构化的数据吗？

任务 1－3 搭建大数据工具 Python 环境

情境导入

宏达集团遇到了企业发展不适应时代变化、信息成本高、数据利用率低、向管理会计转型缺少信息支撑手段等问题，准备构建企业财务与经营大数据分析平台，破解所面临的问题。

构建财务大数据分析环境，在计算机上安装与运行 Python、PyCharm 大数据平台。

任务目标

知识目标：

1. 了解大数据工具 Python 的起源与发展。

2. 了解大数据工具的安装流程。

技能目标：

掌握大数据工具 Python、PyCharm、Anaconda 的安装与运行。

素养目标：

开阔财会青年视野、更新知识储备，培育财会青年树立直面财务大数据、用好财务大数据的目标和信心。

建议学时

1 学时。

相关知识

一、Python 的起源与发展

视频1-3

Python 是一种面向对象的解释型计算机程序设计语言，由荷兰人吉多·范·罗苏姆（Guido van Rossum）于 1989 年发明。1989 年圣诞节期间，在阿姆斯特丹，罗苏姆为了打发圣诞节的无趣，决心开发一个新的脚本解释程序，作为 ABC 语言的一种继承。

ABC 是由罗苏姆参加设计的一种教学语言。就罗苏姆本人看来，ABC 这种语言

非常优美和强大，是专门为非专业程序员设计的。但是 ABC 语言并没有成功，究其原因，罗苏姆认为是其非开放造成的。罗苏姆决心在 Python 中避免这一错误。同时，他还想实现在 ABC 中闪现过但未曾实现的东西。

视频1-4

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）。从一出生，Python 已经具有了类（class）、函数（function）、异常处理（exception）及包括表（list）和词典（dictionary）在内的核心数据类型，以及以模块（module）为基础的拓展系统。最初的 Python 完全由罗苏姆本人开发。Python 得到罗苏姆同事的欢迎。他们迅速地反馈使用意见，并参与到 Python 的改进。Python 和 C + + 、Java 一样是一门高级编程语言，也被认为是一门解释型语言，将高级语言的一条语句翻译为机器语言，然后运行，并且解释器一旦发现错误，程序会抛出异常或立即终止。罗苏姆和一些同事构成 Python 的核心团队。

Python 是初学者学习编程的最好语言，是一种不受局限、跨平台的开源编程语言，功能强大、易写易读，能在 Windows、Mac 和 Linux 等平台上运行。Python 在不断发展和完善，也变得越来越普及。

二、Python 的安装与运行

视频1-5

Python 的开发环境比较简单，用户可以直接登录官网下载 Python 的程序包进行安装。Python 自带的 IDE（集成开发环境）功能十分有限，不适合开发 Python 工程项目，目前比较流行的开发环境是 PyCharm 和 Anaconda，这两种开发环境下载和安装都比较方便，不需要进行复杂的环境配置。由于本书教学内容是基于中联集团教育科技有限公司的云教学平台，在配置 Python 开发环境时主要以 Anaconda 中自带的 Jupyter Notebook 为主，因此本书主要介绍 Jupyter Notebook 环境搭建。

具体操作步骤如下。

（1）登录 https：//www. anaconda. com/products/distribution 网站，单击 Download 按钮下载 Anaconda 安装包，如图 1 - 1 所示。也可以从清华大学开源软件镜像站（https：//mirrors. tuna. tsinghua. edu. cn/anaco - nda/archive/）下载 Anaconda 安装包。

（2）双击安装程序进行安装，打开 Anaconda 安装导向，在安装导向界面单击“Next”按钮，进入协议选择界面，单击 I Agree 按钮。

（3）选择安装类型，在选择安装类型界面中，“Install for：All Users（requires

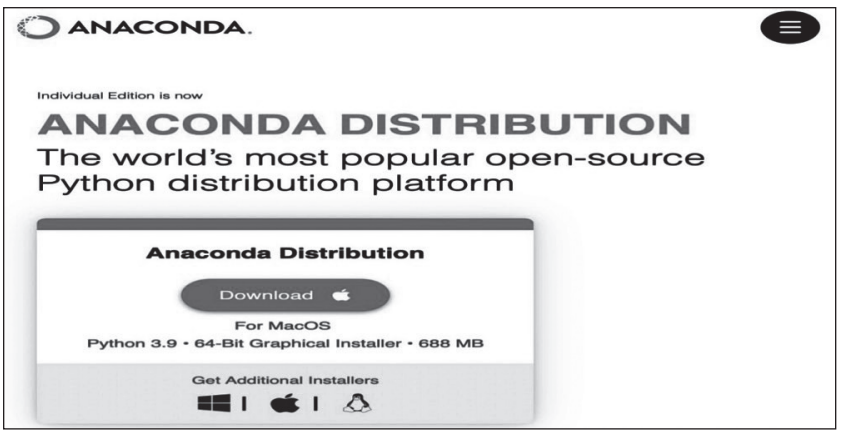


图 1－1　Anaconda 安装包

admin privileges)”表示安装的 Anaconda 软件可以为当前计算机上的所有用户使用，“Install for：Just me（recommended)”表示安装的 Anaconda 软件仅供当前计算机上的当前用户使用。本书选择“All Users（requires admin privileges)”，单击 Next 按钮。

（4）选择合适的安装路径，单击 Next 按钮，进入“Advanced Installation Options”界面，选择“Register Anaconda3 as the system Python3. 9”，然后单击 Install 按钮安装 Anaconda。

（5）在 Anaconda 安装成功界面单击 Next 按钮，在弹出的界面中继续单击 Next 按钮，弹出安装导向的最后一个界面，然后单击 Finish 按钮结束安装程序。

（6）安装完成后，在开始菜单中找到 Jupyter Notebook（Anaconda3）的运行文件，启动 Jupyter Notebook 环境，在启动 Jupyter Notebook 环境的过程中，会弹出一个黑色的启动窗口，按照启动窗口中的提示在浏览器中打开 Jupyter Notebook 的主界面。

（7）单击主界面右上角的下拉按钮 New，选择 Python，Jupyter 将打开一个可编辑 Python 程序代码的新 Notebook 页面。

可以通过菜单栏中的功能操作 Jupyter Notebook，从而实现对程序代码的编辑和运行等，工具栏中的工具按钮均来自菜单栏，工具栏是菜单栏中使用非常频繁的菜单项的列示区域。在实际运用中，可以使用工具栏中的工具按钮，也可以使用菜单栏中的菜单项进行操作。

以上就是完整的 Python 的安装与运行过程，希望通过其安装与配置过程，能够体会到每一步操作体现的严谨性，养成认真严谨对待数据的思维方式。

三、PyCharm 平台的安装与运行

PyCharm 是一个适合用于开发的多功能 IDE，可下载免费社区版。PyCharm 常用功能包括：代码分析与辅助功能，拥有补全代码、高亮语法和错误提示；项目和代码导航，专门的项目视图、文件结构视图和文件、类、方法等的快速跳转；支持网络框架 Django，web2py 和 Flask；集成 Python 调试器；集成单元测试，按行覆盖代码；集成版本控制系统，为 Git、Subversion 和 CVS 提供统一的用户界面，拥有修改以及合并功能。

具体操作步骤如下。

（1）下载 PyCharm，可以从官网（http：//www. jetbrains. com/pycharm/download/）下载，如图 1 -2 所示。

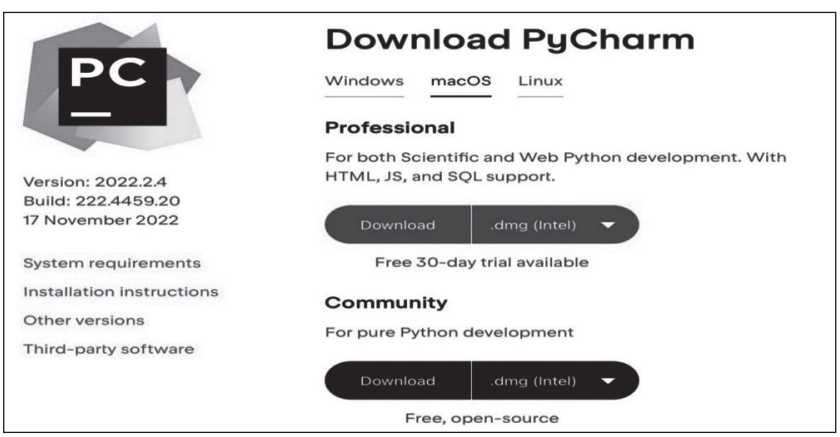


图 1 -2　PyCharm 官网下载

（2）双击 PyCharm 执行程序，进入欢迎安装 PyCharm 界面。

（3）选择安装路径，建议选择默认，可根据需要修改安装路径，单击 Next 按钮。

（4）安装完成，单击 Finish 按钮，完成安装。

技能训练

1. 尝试自主完成 Anaconda 平台的安装与运行。

2. 请以小组为单位完成主题为“大数据技术对我们生活的影响”的调查分析报告。

即测即练

课后拓展阅读

拓展阅读1-1

思政小课堂

视频1-6

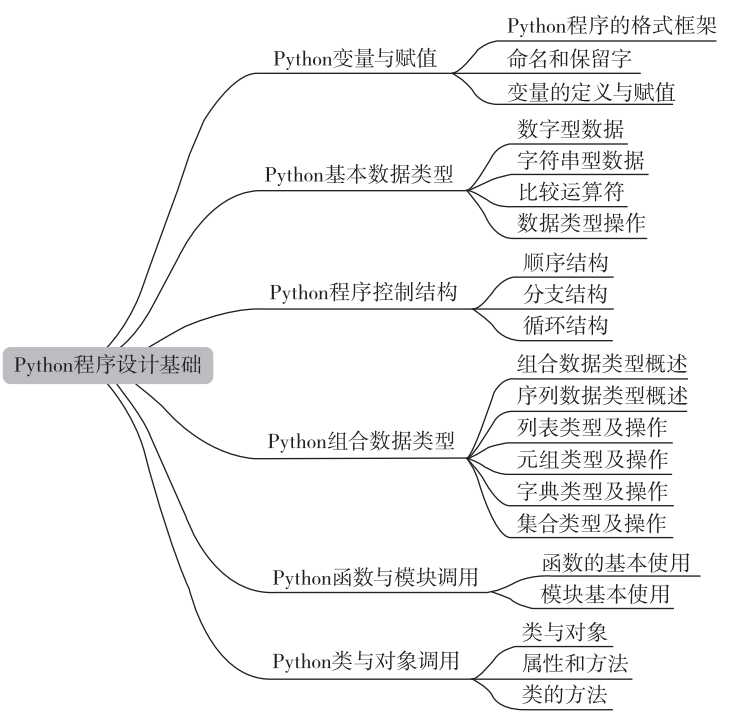
项目 2　Python 程序设计基础

Python 语言是一种解释型、面向对象、动态数据类型的高级程序设计语言，具有 30 多年的发展历史，成熟且稳定。Python 的语法简洁清晰，入门容易，包含一组丰富而强大的标准库和大量的第三方库，能够轻松完成很多常见的任务，被应用在很多领域，如科学计算、图形处理、云计算、Web 开发、网络爬虫、数据分析等。

项目提要

本项目主要介绍 Python 变量与赋值、Python 基本数据类型、Python 程序控制结构、Python 组合数据类型、Python 函数与模块调用、Python 类与对象调用等。

项目思维导图



建议学时

16 学时。

任务 2－1 Python 变量与赋值

情境导入

某公司要进行财务大数据分析，需要利用 Python 工具完成。进行数据分析之前，需要确定数据分析的工具，Python 的语法简单，容易学习和掌握。学习 Python 需要先了解 Python 语言的基本语法，如掌握 Python 变量的使用。

任务目标

知识目标：

1. 了解 Python 的基本语法元素规则。

2. 熟悉变量的命名规则。

技能目标：

掌握赋值语句的规范用法。

素养目标：

培养严谨认真、实事求是的科学精神。

建议学时

2 学时。

相关知识

一、Python 程序的格式框架

Python 基本语法元素包括缩进、注释、语句续行符号、语句分隔符号等。

（一）缩进

Python 程序是依靠语句块的缩进来体现代码与代码之间的逻辑关系的，缩进结束就表示一个代码块的结束。每行代码开头的空白（空格或制表符）表示缩进级别，而缩进级别又用于确定语句块的组成。缩进是 Python 语法的一部分，用于表示

代码间的包含和层级关系，如果缩进错误，将导致程序运行错误。缩进在程序内保持一致即可，每个层级一般用4个空格或按1次<Tab>键实现。

Python默认从程序的第一条语句开始，按顺序依次执行各条语句。在Java、C/C++等语言中，用大括号{}表示代码块。在Python中则使用缩进（空格）表示代码块，连续的多条具有相同缩进量的语句为一个代码块。通常，语句末尾的冒号表示代码块的开始。示例代码如下：

```
if a>0:
    b=1
else:
    b=-1
```

Python语言采用严格的“缩进”表明程序的格式框架。在代码编写中，缩进用Tab键实现，也可以用多个空格实现，但二者一般不混用。建议采用4个空格方式予以缩进。严格的缩进格式有利于约束程序结构，也有利于提高代码结构的可读性。

必须注意，同一个代码块中的语句，其缩进量应保持相同，否则会发生IndentationError（缩进错误）异常。示例代码如图2-1所示。

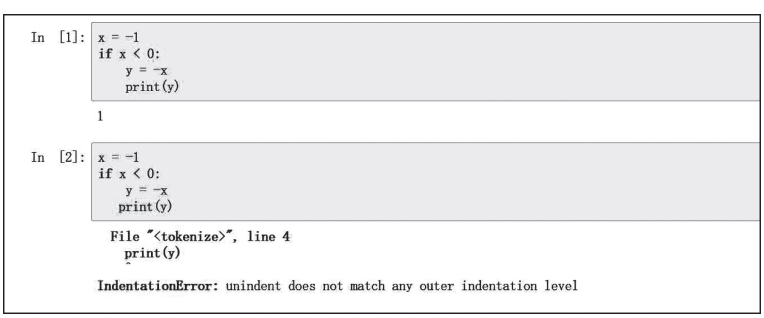


图2-1　示例代码

（二）注释

注释用于为程序添加说明性的文字，能让程序员读懂每一行代码的意义，也为后期代码维护提供便利。注释是代码中不被计算机执行的辅助性说明文字，因其会被编译器或解释器略去，所以一般用于在代码中标明编写者及版权信息、解释代码原理和用途或辅助程序调试等。

任何编程语言都少不了注释，Python也不例外，根据内容的量级差异，注释可分为单行注释和多行注释。以下是Python注释的具体用法。

1. 单行注释

Python 编程语言的单行注释常以“#”开头，单行注释可以单独占一行，也可以放在语句末尾。示例代码如下：

```
# 这是一个 Python 程序
```

2. 多行注释

Python 中多行注释有两种方式：一种是需要在每行注释内容的开头使用“#”；另一种是三引号注释，在特殊的程序位置上，该方式也称为文档字符串。文档字符串是一个解释程序的重要工具，有助于读者理解程序。简而言之，它可以实现“帮助文档”的功能，可以提供函数的基本信息、函数的功能简介以及形式参数的类型和使用方式等信息。当然，这些信息都是由编写者填写、创建的，函数不会自动提供。

文档字符串是一个多行字符串，通过在函数体的第一行使用一对三个单引号（'''）或者一对三个双引号（"""）来定义。首先用于介绍模块的大致功能，第二行为空行，从第三行开始是此模块的详细介绍。可以使用“_ doc_ ”（注意是双下划线）方法查看文档字符串。

注释和文档字符串都可以起到注释说明的作用，并且不影响 Python 程序的运行。但注释不能被程序调用，文档字符串可以被程序调用。

多行注释使用三个单引号（'''）或者三个双引号（"""）来标记，作为注释的开始和结束符号。示例代码如下：

```
'''
这是多行注释，使用单引号。
多行注释结束。
'''
"""
这是多行注释，使用双引号。
多行注释结束。
"""
```

（三）语句续行符号

通常，Python 中的一条语句占一行，没有语句结束符号。可以使用语句续行符号将一条语句写在多行之中。Python 语句续行符号为“\”。示例代码如下：

```
If x < 50   \
```

```
    and x > 10:
        y = x * 2
else:
    y = 0
```

注意：在符号“\”之后不能有任何其他符号，包括空格和注释。

（四）语句分隔符号

Python 使用分号“;”作为语句分隔符号，从而将多条语句写在一行。示例代码如下：

```
print("资产负债表");print(2 + 3)
```

使用语句分隔符号分隔的多条语句可以视为一条复合语句，Python 允许将单独的语句或复合语句写在冒号之后。

二、命名和保留字

Python 程序采用“变量”来保存和表示具体的数据值。为了更好地使用变量等其他程序元素，给它们赋予一个标识符（名字），这个赋予标识符的过程称为命名。命名用于保证程序元素的唯一性。

Python 语言允许采用大写字母、小写字母、数字、下划线和汉字等字符及其组合给变量命名。但名字的首字符不能是数字，中间不能出现空格。

注意：标识符对大小写敏感，Ab 和 ab 是两个不同的名字。

每种程序设计语言都有一套保留字，一般来说，命名时不能与 Python 的保留字相同。保留字也称为关键字，是由设计者或维护者预先创建并保留使用的标识符。保留字一般用于构成程序的整体框架、表达关键值和具有结构性的复杂语义等。程序员编写程序时不能定义与保留字相同的标识符。Python 的保留字对大小写也敏感。Python 3.12 版本共有 35 个保留字，具体保留字如表 2-1 所示。

表 2-1　Python 3.12 的保留字

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

三、变量的定义与赋值

(一) 变量的定义

视频2-1

程序中用来保存和表示数据的语法元素称为变量，它是一种常见的占位符号。变量采用标识符表示，由数字、汉字、下划线、大小写字母等字符组合而成，如 TempStr、Python3 学习方法等。

(二) 变量的命名规则

(1) 变量名由数字、字母、下划线等组成，但不能以数字开头，如 12python 是不合法的。

(2) 变量名不能使用 Python 关键字，如 if、while、true 等。

(3) 变量名对英文字母的大小写敏感，如 a = 2 和 A = 2 是不同的变量。

(4) 变量名中除了下划线“_”以外，不能有其他任何特殊字符。

(5) 使用可理解、行业通用的变量名，而非无意义的简写，便于理解变量含义。

(三) 赋值语句

赋值语句用于将数据赋值给变量，将数据放入变量的过程叫作赋值。在 Python 语言中，使用“ = ”作为赋值运算符。

具体格式如下：

name = value

name 表示变量名；value 表示变量值，也就是要存储的数据。

注意，变量是标识符的一种，它的名字不能随便起，要遵守 Python 标识符命名规范，还要避免和 Python 内置函数以及 Python 保留字重名。

变量赋值常见的情况有单变量赋值、多变量赋值及同步赋值。

1. 单变量赋值

单变量赋值，即一次为一个变量进行赋值。例如，下面的语句将整数 10 赋值给变量 n，即 $n = 10$，从此以后，n 就代表整数 10，使用 n 也就是使用 10。变量的值不是一成不变的，它可以随时被修改，只要重新赋值即可；另外，也不用关心数据的类型，可以将不同类型的数据赋值给同一个变量。注意，变量的值一旦被修改，之前的值就被覆盖了，不复存在了，再也找不回了。换句话说，变量只能容纳一个值。

除了赋值单个数据，也可以将表达式的运行结果赋值给变量，如图 2 - 2 所示。

图 2－2　表达式结果赋值变量

此处，成本、余数、str1 都是变量名，但建议使用 str1 这样有提示作用的英文加数字的命名法。

2. 多变量赋值

元组赋值语句可以同时给多个变量赋值，即同时计算等号右侧所有表达式值，并一次性给等号左侧对应变量赋值。Python 按先后顺序依次将数据赋值给变量，示例如图 2－3 所示。

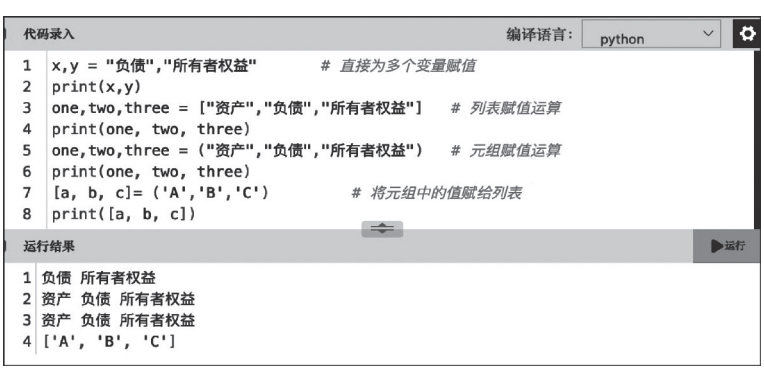


图 2－3　多变量赋值

3. 同步赋值

Python 还有一种同步赋值语句，可以同时给多个变量赋值，基本格式如下：

＜变量 1＞，…，＜变量 N＞＝＜表达式 1＞，…，＜表达式 N＞

例：将变量 x 和 y 的值交换时，采用单个赋值，需要 3 行语句：

```
t = x
x = y
y = t
```

即通过一个临时变量 t 缓存 x 的原始值，然后将 y 值赋给 x，再将 x 的原始值通过 t 赋给 y。

采用同步赋值语句，则仅需要一行代码：

```
x,y = y,x
```

示例如图 2 -4 所示。

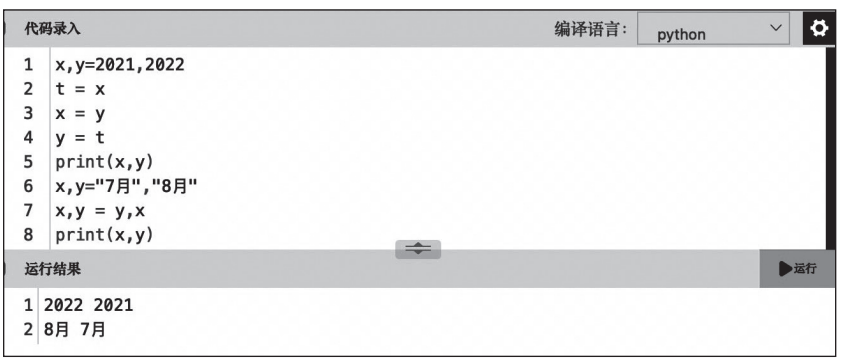


图 2 -4　同步赋值

同步赋值语句可以使赋值过程变得更简洁，通过减少变量使用，简化语句表达，提高程序的可读性。但是，应尽量避免将多个无关的单一赋值语句组合成同步赋值语句，否则会降低程序可读性。

技能训练

A 公司截至 2023 年 11 月累计营业收入为 890 000 元，12 月营业收入发生额为 88 000 元。请使用赋值语句计算 2023 年全年的营业收入，并将计算结果输出显示。

提示：设置变量 income 和 accumulated income，将营业收入发生额赋值给变量 income，将累计营业收入赋值给变量 accumulated income。

任务 2 -2　Python 基本数据类型

情境导入

某公司利用 Python 进行财务大数据分析，在数据处理过程中，数据类型决定 Python 如何存储和处理数据。Python 有 4 个基本数据类型，分别是整数型 int、浮点型 float、字符串型 str 和布尔型 bool。

任务目标

知识目标：

1. 熟悉 Python 的整数型 int、浮点型 float 等数据类型。

2. 能区分 Python 基本数据类型的不同。

技能目标：

掌握基本数据类型的正确操作方法。

素养目标：

增强信息素养，培养严谨细致的工作态度。

建议学时

3 学时。

相关知识

在利用 Python 对数据进行运算和操作时，需要明确数据的类型，不同的类型具有不同的操作，并且每一种数据类型都有自己独特的形式。Python 语言支持多种数据类型，常见的主要有数字型、字符串型等。

视频2-2

一、数字型数据

数字类型用于存储数值，是不可改变的数据类型。这意味着，若要改变变量的数据类型，就要为其赋值一个新的变量。Python 常用的数字型数据可细分为整数型、浮点型、布尔型、复数型。

（一）整数型

整数型与数学中的整数相对应，整数就是没有小数部分的数字，Python 中的整数包括正整数、0 和负整数。

1. 整数的表示形式

二进制形式：由 0 和 1 两个数字组成，书写时以 0b 或 0B 开头。例如，101 对应十进制数是 5。

八进制形式：八进制整数由 0 ~ 7 共八个数字组成，以 0o 或 0O 开头。注意，第一个符号是数字 0，第二个符号是大写或小写的字母 O。

十进制形式：我们平时常见的整数就是十进制形式，它由 0 ~ 9 共十个数字排

列组合而成。

十六进制形式：由 0 ~ 9 十个数字以及 A ~ F 或 a ~ f（字母 a ~ f 表示 10 ~ 15）六个字母组成，书写时以 0x 或 0X 开头。

进制形式是整数数值的不同显示方式，同一个整数的不同进制形式在数学意义上是没有区别的，程序可以直接对不同进制形式的整数进行运算或比较。无论采用何种进制形式表示数据，运算结果均以默认的十进制形式显示，示例如下：

1010，99，－217

0x9a，－0X89（0x，0X 开头表示十六进制数）

0b010，－0B101（0b，0B 开头表示二进制数）

0o123，－0O456（0o，0O 开头表示八进制数）

2. int 数据类型

整数是 Python 的 int 数据类型的值，Python 整数与数学中的整数概念一致，没有取值范围限制，可以用 Python 的内置函数 type() 查看其数据类型，如图 2－5 所示。

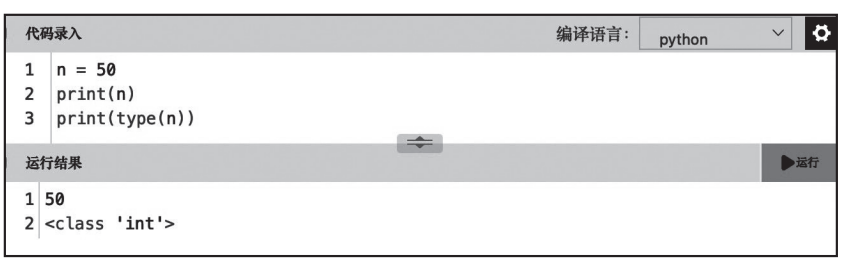


图 2－5　查看数据类型

Python 的 int 数据类型如表 2－2 所示。

表 2－2　Python 的 int 数据类型

类别	说明						
int 对象的值	一系列 0 到 9 的阿拉伯数字组合						
典型字面量	1287、96、0、1000000						
运算	加	减	乘	除	整除	取余	乘幂
运算符	+	－	*	/	//	%	* *

（二）浮点型

在编程语言中，小数通常以浮点数的形式存储。浮点数类型的名称为 float。

1. Python 中小数的表示形式

（1）十进制形式。这种就是我们平时看到的小数形式，如 34.6、346.0、0.346。书写小数时必须包含一个小数点，否则会被 Python 当作整数处理。

（2）指数形式。Python 小数的指数形式的写法为：

aEn 或 aen

其中，a 为尾数部分，是一个十进制数；n 为指数部分，是一个十进制整数；E 或 e 是固定的字符，用于分割尾数部分和指数部分。整个表达式等价于 $a \times 10^n$。例如：$2.1E5 = 2.1 \times 10^5$，其中，2.1 是尾数，5 是指数。小数的指数形式示例如图 2－6 所示。

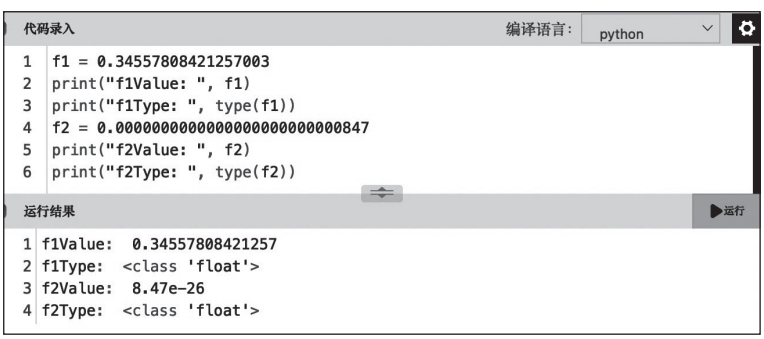


图 2－6　小数的指数形式

2. float 数据类型

Python 的 float 数据类型如表 2－3 所示。

表 2－3　Python 的 float 数据类型

类别	说明				
float 对象的值	一系列 0 到 9 的阿拉伯数字加上小数点				
典型字面量	3.14159、1.287e23、96.0、1.4142135				
运算	加	减	乘	除	乘幂
运算符	+	-	*	/	* *

（三）布尔型

bool 数据类型用于表示逻辑值：真或者假。bool 数据类型包含两个值，其对应的字面量为：True 和 False。布尔运算的操作数为 True 或 False，结果依旧为 True 或 False。布尔数据类型表面看起来简单，却是计算机科学的基础之一。

Python 的 bool 数据类型如表 2－4 所示。

表 2－4 Python 的 bool 数据类型

类别	说明
bool 对象的值	真或者假
典型字面量	True
运算	逻辑与 逻辑或 逻辑非
逻辑运算符	and，or，not

bool 对象的运算符称为逻辑运算符，共 3 个，分别是：and、or 和 not，各种布尔运算的真值表如表 2－5 所示。从表 2－5 中可以看出，and 和 or 是二元运算符，带有两个操作数，not 是一元运算符，只有一个操作数。

表 2－5 bool 运算的真值表

a	b	a and b	a or b	not a
False	False	False	False	True
False	True	False	True	True
True	False	False	True	False
True	True	True	True	False

以表 2－5 中第三行的运算为例，执行结果如图 2－7 所示。

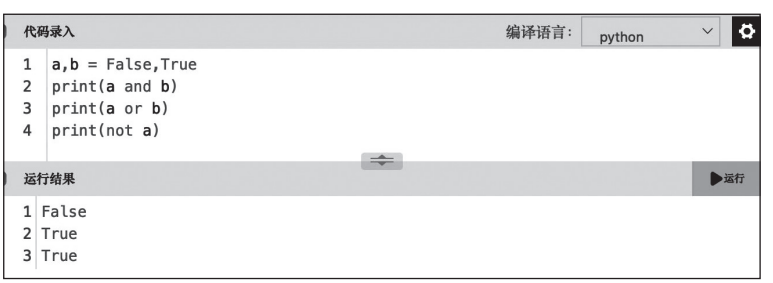


图 2－7 bool 运算示例

（四）复数型

复数类型与数学中的复数相对应，其值由实数部分和虚数部分组成，虚数部分的基本单位为 j。复数类型的一般形式为 $x + y$j，其中，x 是复数的实数部分，yj 是复数的虚数部分，这里的 x 和 y 都是实数。（当 $y = 1$ 时，1 是不能省略的，因为 j 在 Python 程序中是一个变量，此时如果将 1 省略，程序会出现报错。）

复数型数值的实数部分和虚数部分都是浮点数。对于一个复数，可以用“. real”和“. imag”得到它的实数部分和虚数部分。虚数部分不能单独存在，Python 会为其自动添加一个值为 0.0 的实数部分以与其一起构成复数。虚数部分必须

有 j 或 J。

二、字符串型数据

（一）Python 字符串类型

字符串是一种有序的字符集合，用于表示文本数据。若干个字符的集合就是一个字符串（String）。Python 中的字符串必须由双引号" "，或者单引号'，或者三个单引号或双引号包围。

Python 字符串中的双引号和单引号没有任何区别。字符串的内容可以包含字母、标点、特殊符号、中文、日文等全世界的所有文字。Python 的 str 数据类型如表 2－6 所示。

表 2－6　str 数据类型

类 别	说 明
str 对象的值	字符系列
典型字面量	'Hello，world'、'Python \ s'
运算	字符串拼接
运算符	+　　*

（二）Python 字符串的拼接

字符串可以使用加号（+）连接，使用加号连接的各个变量或者元素必须是字符串类型。字符串拼接的示例代码如图 2－8 所示。

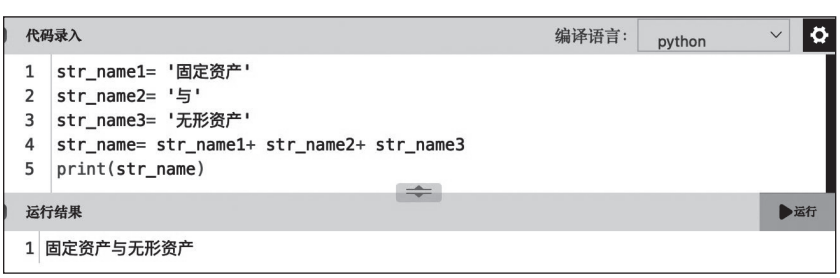


图 2－8　字符串拼接

多个相邻的字符串（以空白符分隔），所用的引号可以彼此不同，其含义等同于全部拼接为一体。因此，'固定资产'与'无形资产'等同于"固定资产与无形资产"，如图 2－9 所示。此特性可以减少反斜杠的使用，以方便将很长的字符串分成多个物理行，甚至每部分字符串还可分别加注释。

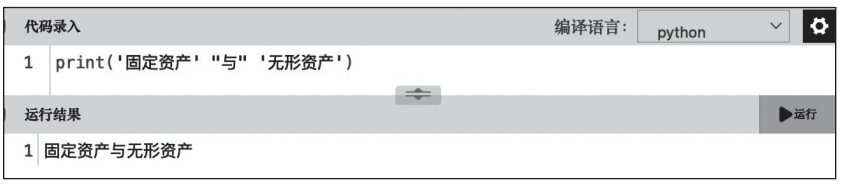


图 2－9　相邻字符串拼接

也可以使用运算符 * 重复拼接同一个字符串，尤其是产生固定数量的空格，下面将含有“空格”二字的字符串重复 8 次，示例如图 2－10 所示。同样，执行代码 print("" *8)，恰好可以得到 8 个位置的空格。

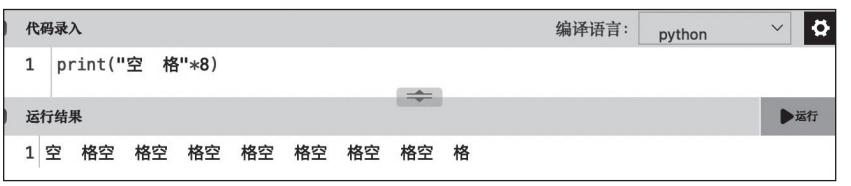


图 2－10　字符串重复

（三）Python 字符串格式化

1. 转换说明符的应用

print() 语句可以格式化输出，print() 函数使用% 开头的转换说明符对各种类型的数据进行格式化输出，各类转换说明符的具体含义如表 2－7 所示。

表 2－7　各类转换说明符的具体含义

转换说明符	解释
%d、%i	转换为带符号的十进制整数
%o	转换为带符号的八进制整数
%x、%X	转换为带符号的十六进制整数
%e	转化为科学记数法表示的浮点数（e 小写）
%E	转化为科学记数法表示的浮点数（E 大写）
%f、%F	转化为十进制浮点数
%g	智能选择使用 %f 或 %e 格式
%G	智能选择使用 %F 或 %E 格式
%c	格式化字符及其 ASCII 码
%r	使用 repr() 函数将表达式转换为字符串
%s	使用 str() 函数将表达式转换为字符串

在 print() 函数中，由引号包围的是格式化字符串，它相当于一个字符串模板，

可以放置一些转换说明符（占位符）。其中格式化字符串中包含一个%d说明符，它最终会被后面的cash变量的值所替代。中间的%是一个分隔符，它前面是格式化字符串，后面是要输出的表达，如图2-11所示。

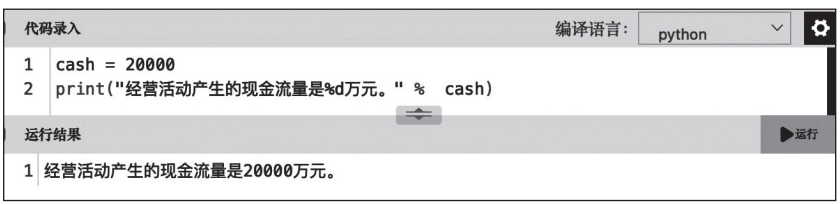


图2-11　格式化输出

格式化字符串中也可以包含多个转换说明符，这个时候也得提供多个表达式，用以替换对应的转换说明符；多个表达式必须使用小括号（）包围起来，如图2-12所示。

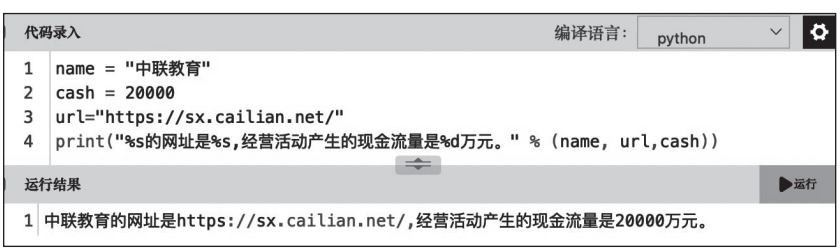


图2-12　多个转换说明符的应用

2. format()方法

格式化字符串，也可以使用format()方法，该方法灵活方便，不仅可以使用位置进行格式化，还支持使用关键参数进行格式化。

（1）使用位置进行格式化，示例如图2-13所示。

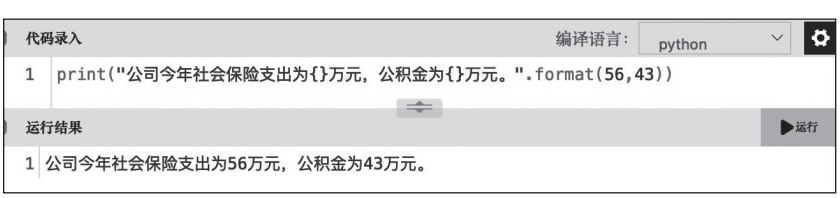


图2-13　使用位置进行格式化

（2）使用关键参数进行格式化，示例如图2-14所示。

图 2－14　使用关键参数进行格式化

三、比较运算符

Python 中的某些混合型运算符作用于一种数据类型，而结果却返回另外一种数据类型。最常用的混合型运算符是比较运算符（==、!=、<、<=、>和>=），可作用于整数和浮点数，返回布尔结果值。以 int 数据类型为例说明比较运算符的运算结果，如表 2－8 所示。

表 2－8　比较运算符示例

运算符	含义	True	False
==	等于	2 == 2	2 == 3
!=	不等于	3 != 2	2 != 2
<	小于	2 < 22	2 < 2
<=	小于或等于	2 <= 5	4 <= 2
>	大于	22 > 2	2 > 22
>=	大于或等于	3 >= 2	2 >= 3

四、数据类型操作

Python 可以使用 type() 函数查看数据类型，不同数据类型之间可以进行转换。Python 提供了多种可实现数据类型转换的函数，常见类型转换函数如表 2－9 所示。

表 2－9　常见类型转换函数

函数	作用	举例
int(x)	将 x 转换为整数	int("123")结果为整数 123
float(x)	将 x 转换为浮点数	float("1.2")结果为 1.2
str(x)	将对象 x 转换为字符串	str(12)结果为'12'
chr(x)	将整数 x 转换为字符	chr(65)结果为 A
ord(x)	将字符 x 转换为它对应的整数值	ord(A)结果为 65
eval(str)	用来计算字符串中的有效的 Python 表达式，并返回一个对象	eval("10+20+30")结果为 60
hex(x)	将一个整数转化为一个十六进制字符串	hex(4286)结果为'0x10be'

续表

函数	作用	举例
oct(x)	将一个整数转换为一个八进制字符串	oct(4286)结果为'0o10276'
repr(x)	将对象 x 转化为表达式字符串	repr(3 * 8)结果为'24'

需要注意的是，在使用类型转换函数时，提供给它的数据必须是有意义的。例如，int()函数无法将一个非数字字符串转换成整数，示例如图 2 – 15 所示。

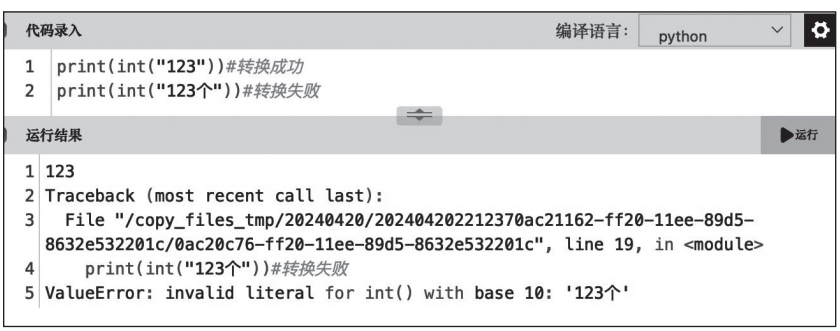


图 2 – 15　类型转换函数应用

技能训练

结合中联信合资产有限责任公司 2023 年现金流量表（部分），根据 Python 代码编辑器“1023692. ipynb”中预置的部分代码，将相关代码补全，并依次输出以下项目及对应的金额：

1. 经营活动现金流量。
2. 销售商品、提供劳务收到的现金。
3. 收到的税费返还。
4. 收到其他与经营活动有关的现金。
5. 经营活动现金流入小计。

其资料如表 2 – 10 所示。

表 2 – 10　中联信合资产有限责任公司 2023 年现金流量表（部分）　　万元

项目	本年金额
一、经营活动产生的现金流量金额	
销售商品、提供劳务收到的现金	37 825 542. 86
收到的税费返还	0
收到其他与经营活动有关的现金	1 561 208. 89
经营活动现金流入小计	

任务2－3 Python程序控制结构

情境导入

某公司利用Python进行财务大数据分析，在数据处理过程中，需要程序控制的三种结构：顺序结构、分支结构和循环结构。顺序结构是程序中的语句按照先后顺序执行，分支结构是根据条件执行不同的代码，循环结构则重复执行相同的代码。Python用if语句实现分支结构，用for和while语句实现循环结构。

任务目标

知识目标：

1. 熟悉程序控制中的顺序结构、分支结构和循环结构。
2. 能够区分顺序结构、分支结构和循环结构的不同。

技能目标：

1. 掌握if语句、for语句和while语句的正确使用方法。
2. 熟练应用单分支、二分支、多分支结构。

素养目标：

增强数据逻辑的判断能力，培养代码规范意识。

建议学时

2学时。

相关知识

一、顺序结构

视频2-3

程序由顺序结构、分支结构和循环结构三种基本结构组成。

顺序结构是程序按照线性顺序依次执行的一种运行方式，如图2－16所示。

二、分支结构

分支结构是程序根据条件判断结果而选择不同向前执行路径的一种运行方式，其中，向前执行表示向代码前进方向直行。

根据程序中分支路径上的完备性，保证程序判断条件充分的情况下，分支结构主要可以分为单分支结构、二分支结构、多分支结构三种。

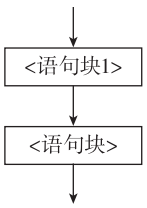


图 2 – 16　顺序结构

（一）单分支结构：if

使用 if 对条件进行判断，控制流程图如图 2 – 17 所示。

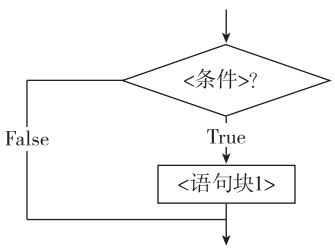


图 2 – 17　单分支结构控制流程图

语法使用方式如下：

if < 条件 > :

　　< 语句块 >

其中，if、< 条件 > 、: 、< 语句块 > 和 < 语句块 > 前的缩进部分都是语法的一部分。

< 语句块 > 是 if 条件满足后执行的一个或多个语句序列，缩进表达 < 语句块 > 与 if 的包含关系。< 条件 > 是一个产生 True 或 False 结果的语句，当结果为 True 时，执行 < 语句块 > ，否则跳过 < 语句块 > ，示例代码如图 2 – 18 所示。

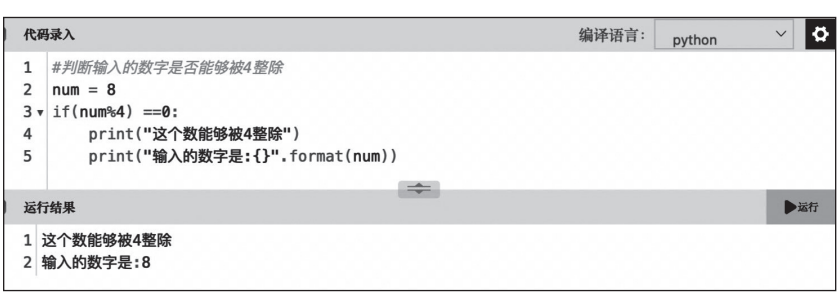


图 2 – 18　if 条件判断应用

（二）二分支结构：if – else

使用 if – else 对条件进行判断，控制流程图如图 2 – 19 所示。

语法格式如下：

if < 条件 > :

< 语句块 1 >

else:

< 语句块 2 >

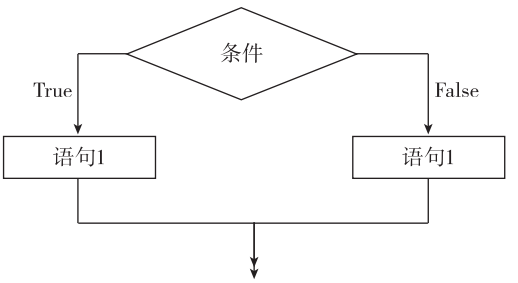


图 2 – 19 二分支结构控制流程图

其中，if、< 条件 > 、:、< 语句块1 > 、else、:、< 语句块 2 > 和 < 语句块 1 > 、< 语句块 2 > 前的缩进部分都是语法的一部分。

< 语句块 1 > 在 < 条件 > 满足，即为 True 时执行；< 语句块 2 > 在 < 条件 > 不满足，即为 False 时执行。简单说，二分支结构根据条件的 True 和 False 结果产生两条路径，示例如图 2 – 20、图 2 – 21 所示。

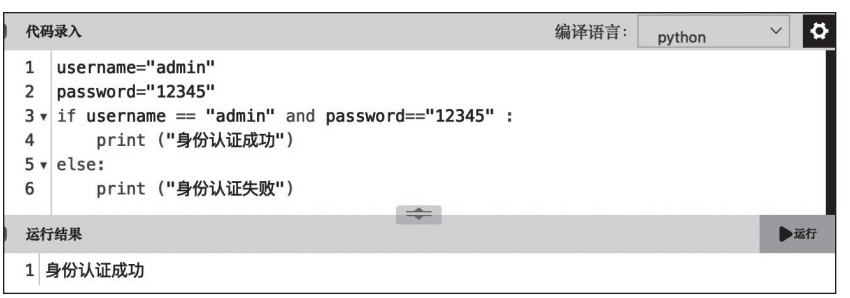


图 2 – 20 二分支结果为 True 示例

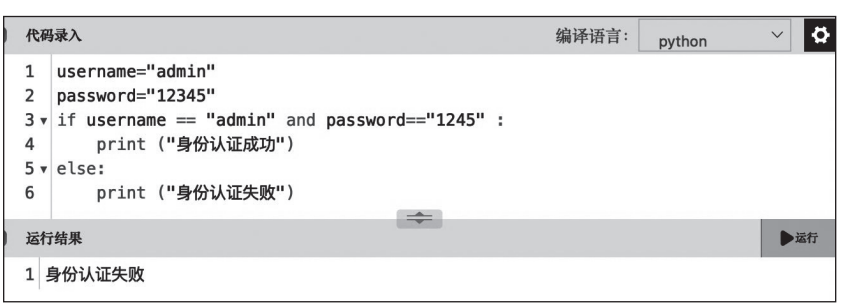


图 2 – 21 二分支结果为 False 示例

（三）多分支结构：if – elif – else

使用 if – elif – else 对多个相关条件进行判断，并根据不同条件的结果按照顺序选择执行路径，控制流程图如图 2 – 22 所示。

语句格式如下：

if < 条件 1 > :

<语句块1>
elif <条件2>：
<语句块2>
else：
<语句块3>

其中，if、elif、else、：和 <语句块> 前的缩进部分都是语法的一部分。

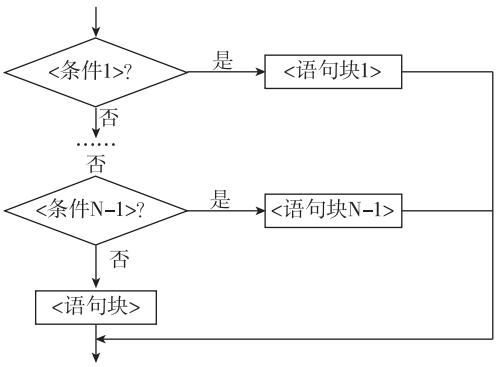


图 2-22　多分支结构控制流程图

三、循环结构

Python 语言的循环结构包括两种：遍历循环结构和条件循环结构。遍历循环使用关键字 for 依次提取遍历结构元素进行处理；条件循环使用关键字 while 根据判断条件执行程序。

（一）遍历循环：for

通过 for 循环依次提取遍历结构元素进行处理，控制流程如图 2-23 所示。

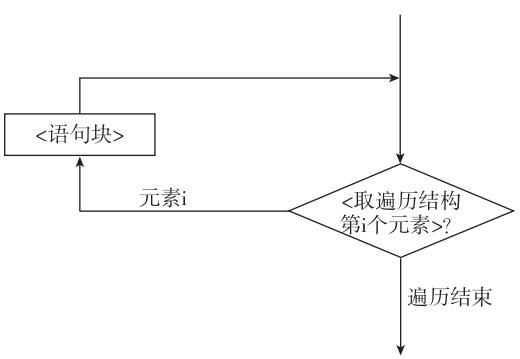


图 2-23　for 循环控制流程

基本语法格式如下：

for <循环变量> in <遍历结构>：
<语句块>

遍历循环还有一种扩展模式，使用方法如下：

for <循环变量> in <遍历结构>：
<语句块1>
else：
<语句块2>

遍历循环可以理解为，从遍历结构中逐一提取元素，放到循环变量中，对于每

个所提取的元素执行一次语句块。for 语句的循环执行次数是根据遍历结果中的元素个数确定的。

同时 for 还可以实现遍历功能，遍历结构可以是字符串、文件、range（）函数、组合数据类型等，示例如图 2－24 所示。

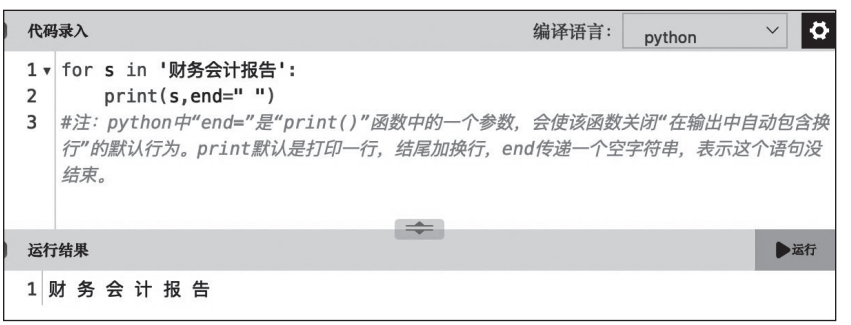


图 2－24　for 遍历循环示例

（二）条件循环：while

通过 while 循环根据判断条件执行程序，条件循环流程如图 2－25 所示。

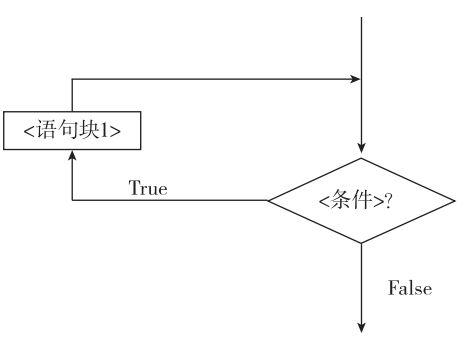


图 2－25　while 条件循环流程

基本使用方法如下：

while < 条件 >：

< 语句块 >

当 < 条件 > 判断为 True 时，循环体重复执行语句块中语句；当 < 条件 > 为 False 时，循环终止，执行与 while 同级别缩进的后续语句，示例如图 2－26 所示。

（三）循环保留字：break 和 continue

在 Python 语言中，可以通过 break 和 continue 保留字来实现辅助循环控制。break 用来跳出最内层 for 或 while 循环，脱离该循环后程序继续执行循环后续代码。

代码录入　　编译语言：python

```
#输出100以内的素数
x=1  #初始化变量
while x<100:
    n=2
    while n<x-1:
        if x%n==0:break #若余数为0，说明x不是素数，结束当前循环
        n+=1  #等价于n=n+1，意味着n被重新赋值
    else:
        print(x,end=' ')#正常结束循环，说明x没有被[2,x-1]范围内的数整除，是素数，输出
    x+=1 #等价于x=x+1，意味着x被重新赋值
else:
    print('over')
```

运行结果　　运行

```
1 2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
over
```

图 2－26　while 条件循环示例

continue 用来结束当前当次循环，即跳出循环体中下面尚未执行的语句，但不跳出当前循环。

两个语句的区别是：continue 语句只结束本次循环，不终止整个循环的执行，而 break 具备结束整个当前循环的能力。示例代码如图 2－27 所示。

代码录入　　编译语言：python

```
'''找出100～999范围内的前10个回文数字（"回文数"是一种数字。如：98789，这个数字正读是98789，倒读也是98789，正读倒读一样，所以这个数字就是回文数。）'''
a=[]
n=0
x=100
while x<999:
    s=str(x)  #将x转化为字符串类型
    if s[0]!=s[-1]:   #判断s中的第一个元素是否不等于最后一个元素
        x=x+1
        continue  #若x不是回文数字，回到循环开头，x取下一个值开始循环
    a.append(x)   #x是回文数字，将其加入列表
    n+=1  #累计获得的回文数字个数
    x=x+1
    if n==10:break  #找出10个回文数字时，跳出while循环
print(a)  #break跳出时，跳转到该处执行
```

运行结果　　运行

```
[101, 111, 121, 131, 141, 151, 161, 171, 181, 191]
```

图 2－27　continue 和 break 辅助循环控制示例

技能训练

运用 for 语句和 while 语句，对数字 1 ~ 9 利用 Python 输出数字金字塔，如图 2 - 28 所示。

提示：

用 for 语句输出每行前的空格，以便对齐。

用 while 语句输出每行前半部分数据。

用 while 语句输出每行的剩余数据。

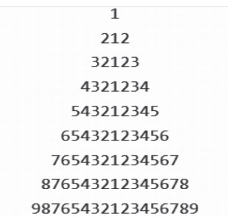


图 2 - 28　输出数字金字塔

任务 2 - 4　Python 组合数据类型

情境导入

现实中的问题所涉及的信息通常不是单一或孤立的，而是相互联系的一系列数据。尽管数字类型可以表示数值，字符串类型可以表示文本，但都只能表示单个值，如果要表示包含多个值的数据需要用 *N* 个数据，且不能体现彼此间的关联，显然会导致效率低下。这就好比搭建一间积木房子，使用的是单块积木还是成型的窗户、墙壁和门。当需要对大量数据进行批量处理时，Python 将 *N* 个元素按某种方式组合起来，使得数据操作更简捷、高效。组合类型根据数据中元素的组织方式和数据特点不同可以分为列表、元组、字典和集合。

任务目标

知识目标：

1. 熟悉组合数据类型的特点。

2. 理解集合类型、序列类型和映射类型的含义。

技能目标：

1. 掌握列表、元组和字典的创建和操作函数应用。

2. 掌握集合类型操作符和常见操作函数的正确使用方法。

素养目标：

独木难成林，树牢集体意识和规范意识，增强团结协作能力。

建议学时

4 学时。

相关知识

一、组合数据类型概述

视频2-4

计算机不仅对单个变量表示的数据进行处理，通常的情况下，其需要对一组数据进行批量处理。例如：给定一组单词“Python、it、data、function、list”，计算并输出每个单词的长度；给定一个班级的信息，统计男女比例；给定某单位的财务报表，对财务报表中大量数据进行分析等。这种能够表示多个数据的类型称为组合数据类型。

组合数据类型能够将多个同类型或者不同类型的数据组织起来，使数据操作更有序、更容易，Python 语言中最常用的组合数据类型有三类，分别是集合类型、序列类型和映射类型。集合类型（集合 set），是一个元素集合，元素之间无序，相同元素在集合中唯一存在。序列类型（字符串 str、列表 list、元组 tuple），是一个元素向量，元素之间存在先后关系，通过序号访问，元素之间不排他。序列类型的典型代表是字符串类型和列表类型。映射类型（字典 dict），是“键 - 值”数据项的组合，每个元素是一个键值对，表示为（key ：value），映射类型的典型代表是字典类型。

不同组合数据类型的特点、表现形式和要求都有所不同，Python 的组合数据类型表如表 2 - 11 所示。

表 2 - 11　Python 的组合数据类型表

内容	列表（list）	元组（tuple）	字典（dict）	集合（set）
定界符	方括号 [ ]	圆括号 ()	大括号 {}	大括号 {}
元素分隔符	逗号，	逗号，	逗号，	逗号，
是否可变	是	否	是	是
是否有序	是	是	否	否
元素形式要求	无特定要求	无特定要求	键：值	必须可哈希
元素值的要求	无特定要求	无特定要求	“键”必须可哈希	必须可哈希
元素是否可重复	可重复	可重复	“键”不允许重复；“值”可以重复	否
元素查找速度	非常慢	很慢	非常快	非常快
增删元素速度	尾部快，其他位置慢	不允许	快	快

二、序列数据类型概述

序列类型是一维元素向量，元素之间存在先后关系，通过索引序号访问元素，序列类型包括字符串、列表和元组。

序列的基本思想和表示方法均来源于数学概念，例如：n 个数的序列 S，可以表示为：$S = s_0$，s_1，s_2，…，s_{n-1}，当访问序列中某个特定元素，只需要通过访问下标即可，如访问 s_2 元素只需要访问序号 2，需要注意，下标编号从 0 开始。另外，由于元素之间存在顺序性，数值相同的两个数可以出现在不同位置。

序列类型各具体类型使用相同的索引体系，即正向递增序号和反向递减序号，如图 2 – 29 所示。

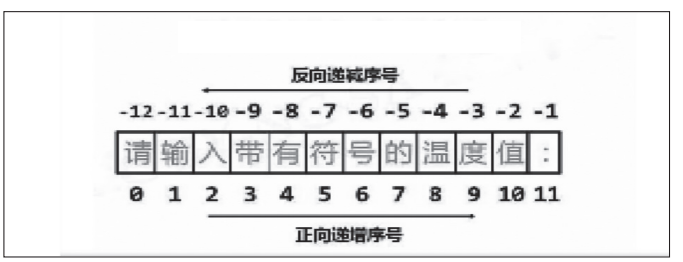


图 2 – 29　正向递增序号和反向递减序号

序列类型总共有 12 个通用的操作符和函数，如表 2 – 12 所示。

表 2 – 12　序列类型的操作符和函数

操作符	描述
x in s	如果 x 是 s 的元素，返回 True；否则返回 False
x not in s	如果 x 不是 s 的元素，返回 True；否则返回 False
s + t	连接 s 和 t
s * n 或 n * s	将序列 s 复制 n 次
s[i]	索引，返回序列的第 i 个元素
s[i:j]	切片，返回包含序列 s 第 i 个到第 j 个元素的子序列（不包含第 j 个元素）
s[i:j:k]	步骤切片，返回包含序列 s 第 i 个到第 j 个元素以 k 为步骤的子序列
len(s)	序列 s 的元素个数（长度）
min(s)	序列 s 中最小元素
max(s)	序列 s 中最大元素
s. index(x)	序列 s 中第一次出现元素 x 的位置
s. count(x)	序列 s 中出现 x 的总次数

三、列表类型及操作

（一）列表的定义

列表是包含 0 个或多个对象引用的有序序列，与字符串、元组不同，列表的长度和内容都是可变的，可自由对列表中的数据项进行增加、删除、替换和查找等操作。

列表类型用方括号“[ ]”表示，所有元素都放在一对方括号“[ ]”里面，相邻元素之间用逗号“,”分隔，格式如下：

$$[\text{element}_1, \text{element}_2, \text{element}_3, \ldots, \text{element}_n]$$

上述格式中，element_1 ~ element_n表示列表中的元素，列表没有长度限制，元素类型可以不同，但必须是 Python 支持的类型，如数值类型、字符串、列表、元组等，示例如图 2 - 30 所示。

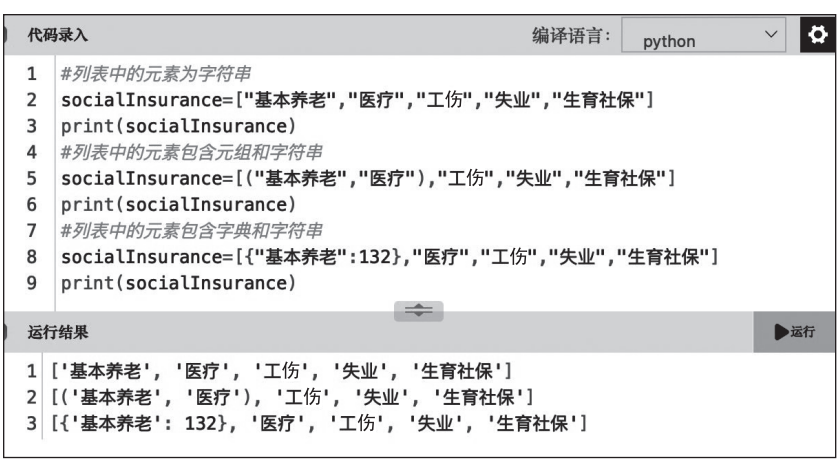


图 2 - 30　列表格式示例

在使用列表时，建议同一个列表中变量使用同一种类型的数据，这样可以提高程序的可读性和可维护性。

由于列表属于序列类型，所以列表支持序列类型的所有操作符和函数，此外列表之间支持通过比较操作符（<、<=、==、!=、>=、>）进行比较，比较的原理是单个元素之间逐个比较，示例如图 2 - 31 所示。

代码录入 编译语言：python

```
1  socialInsurance=[{"基本养老":132},"医疗","工伤","失业","生育社保"]
2  print(socialInsurance)
3  print("商业保险" in socialInsurance)#商业保险是列表中的元素
4  print("商业保险" not in socialInsurance)#商业保险不是列表中的元素
5  welfare=["公积金","补充医疗"]
6  socialInsurance=socialInsurance+welfare  #连接socialInsurance和welfare
   形成一个新的列表socialInsurance
7  print(socialInsurance)
8  print(welfare*2)#将列表welfare复制2次
9  print(len(socialInsurance))#列表中元素的个数
10 print(socialInsurance.index("工伤"))#列表中第一次出现元素"工伤"的位置
11 print(socialInsurance.count("工伤"))#列表中元素"工伤"出现的次数
12 print(socialInsurance == welfare)#判断socialInsurance与welfare是否相等
```

运行结果 运行

```
1 [{'基本养老': 132}, '医疗', '工伤', '失业', '生育社保']
2 False
3 True
4 [{'基本养老': 132}, '医疗', '工伤', '失业', '生育社保', '公积金', '补充医疗']
5 ['公积金', '补充医疗', '公积金', '补充医疗']
6 7
7 2
8 1
9 False
```

图 2－31　列表中操作符和函数应用示例

使用方括号“［ ］”作为索引操作符，用于获得列表的具体元素。该操作沿用序列类型的索引方式，即正向递增序号或反向递减序号，索引序号不能超过列表的元素范围，否则会产生 IndexError 错误，如图 2－32 所示。

代码录入 编译语言：python

```
1  socialInsurance=[{'基本养老': 132}, '医疗', '工伤', '失业', '生育社保', '公
   积金', '补充医疗']
2  #打印列表中的第2个元素
3  print(socialInsurance[1])
4  #打印列表中元素的个数
5  print(len(socialInsurance))
6  #打印列表中第7个元素
7  print(socialInsurance[6])
8  #打印列表中倒数第7个元素，由于列表总共7个元素，也即打印第1个元素
9  print(socialInsurance[-7])
10 #打印列表中倒数第8个元素
11 print(socialInsurance[-8])
```

运行结果 运行

```
1 医疗
2 7
3 补充医疗
4 {'基本养老': 132}
5 Traceback (most recent call last):
6   File "/copy_files_tmp/20240420/20240420223835ab491a9c-ff23-11ee-9691-
  d6733e8ebf3c/ab4915d8-ff23-11ee-9691-d6733e8ebf3c", line 28, in <module>
7     print(socialInsurance[-8])
8 IndexError: list index out of range
```

图 2－32　索引操作符示例

使用切片获取列表类型从 N 到 M（不包含 M）的元素组成新的列表，其中，N 和 M 为列表类型的索引序号，可以混合使用正向递增序号和反向递减序号，一般要求 N 小于 M，当 K 存在的时候，切片获取列表类型从 N 到 M（不包含 M），以 K 为步长所对应元素组成的列表，示例如图 2－33 所示。

图 2－33　索引操作符示例

列表的数据类型是 list，通过 type() 函数和 isinstance() 函数可以判断，如图 2－34 所示。

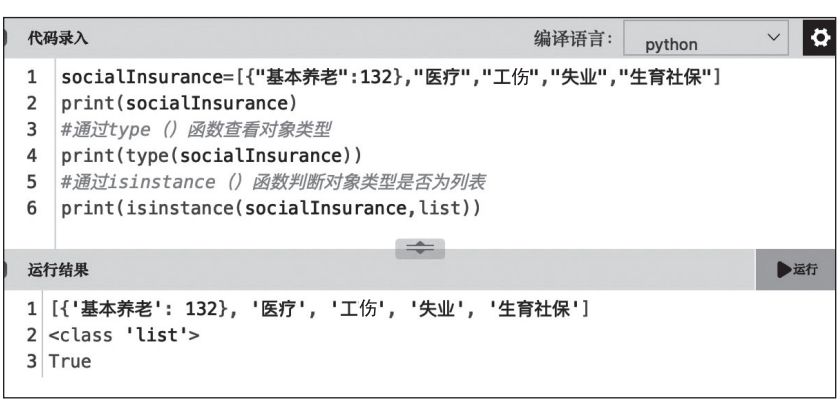


图 2－34　列表数据类型判断

（二）列表类型的操作

列表类型继承序列类型特点，因此有一些通用的操作函数，列表类型的操作函数如表 2－13 所示。

表 2－13　列表类型的操作函数

操作函数	描述
len（ls）	列表 ls 的元素的个数（长度）
min（ls）	列表 ls 中的最小元素
max（ls）	列表 ls 中的最大元素
list（x）	将 x 转换成列表类型

函数 min() 和 max() 分别返回列表的最小、最大元素，使用这两个函数的前提是列表中的元素类型可以进行比较，否则使用这两个函数会报错，如图 2－35 所示。

图 2－35　列表函数应用示例

list(x) 将变量 x 转变成列表类型，其中 x 可以是字符串、元组、集合类型，如图 2－36 所示。

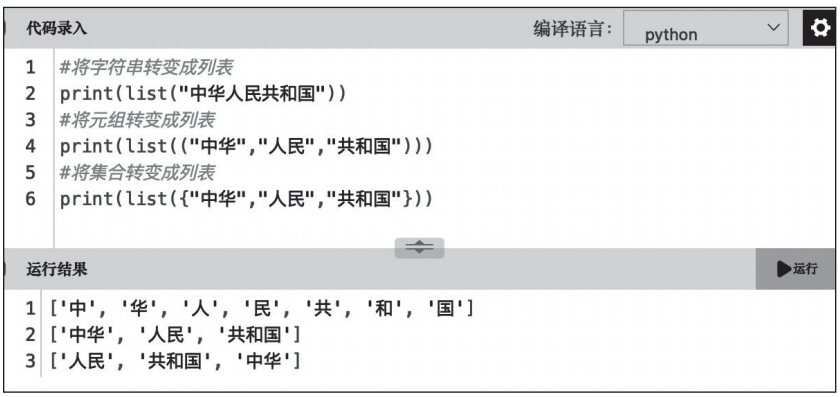


图 2－36　列表 list 函数应用

由于列表是可变的，列表类型特有的常用操作函数和方法如表 2－14 所示。

表 2－14　列表类型特有的常用操作函数和方法

操作函数和方法	描述
ls[i] = x	替换列表 ls 第 i 项数据为 x
ls[i:j] = lt	用列表 lt 替换列表 ls 中第 i 到第 j 项数据（不含第 j 项）
ls[i:j:k] = lt	用列表 lt 替换列表 ls 中第 i 到第 j 项以 k 为步数的数据（不含第 j 项）
del ls[i:j]	删除列表 ls 第 i 到第 j 项数据（不含第 j 项）
del ls[i:j:k]	删除列表 ls 第 i 到第 j 项以 k 为步数的数据（不含第 j 项）
ls += lt 或 ls.extend(lt)	将列表 lt 元素增加到列表 ls 中
ls *= n	更新列表 ls，其中其元素重复 n 次
ls.append(x)	在列表 ls 最后增加一个元素 x
ls.clear()	清除 ls 列表中的所有元素
ls.copy()	生成一个新列表，复制 ls 中的所有元素
ls.insert(i,x)	在列表 ls 的第 i 位置增加元素 x
ls.pop(i)	将列表 ls 中的第 i 项元素取出并删除该元素
ls.remove(x)	将列表中出现的第一个元素 x 删除
ls.reverse()	将列表 ls 中的元素反转

上述操作符和方法主要是对列表元素的增、删、改等操作，示例如图 2－37 所示。实际开发中并不经常使用 del 来删除列表，因为 Python 自带的垃圾回收机制会自动销毁无用的列表，即使开发者不手动删除，Python 也会自动将其回收。