普通高中教科书

信息技术

选择性必修 1

数据与数据结构

上海科技教育出版社

普通高中教科书

信息技术

选择性必修 1

数据与数据结构

上海科技教育出版社

编写人员名单

主　　编： 郑　骏

分册主编： 邓桂英

主要编写人员（以姓氏笔画为序）：

丁　祎　于洋鹏　申一頔

孙时敏　凌　玲

欢迎广大师生来电来函指出教材的差错和不足，提出宝贵意见。

上海科技教育出版社地址：上海市闵行区号景路 159 弄 A 座 8 楼

邮政编码：201101

联系电话：021-64702058

邮件地址：office@sste.com

写给学生的话

亲爱的同学：

今天，我们生活的时代处处离不开数据。每个人、每个企业、每个行业乃至整个社会每天都产生着各种各样的数据。日积月累，这些数据汇聚成的大数据，经过科学的管理与分析，从而产生新的价值并服务于各行各业和人们的日常生活。可以说，数据已成为了一种新的原材料、一种新的生产资料和一种新的基础设施。可能你每天都在不知不觉地采集数据、处理数据、利用数据，但是知道数据在计算机中究竟是如何存储的吗？有着怎样的结构？如何才能被计算机高效处理？

在《数据与数据结构》的学习中，我们将一起通过实例进一步探讨数据及其对社会各领域的影响；剖析日常生活中的实际项目，经历建立数学模型、抽象数据、选择数据结构、设计算法并编程实现等一系列的过程。你将在模拟解决真实问题的过程中，理解数据在计算机系统内部的组织和存储方式，掌握各种数据结构的概念，理解算法的基本思想以及算法与数据结构的关系，从而进一步提升自己的计算思维。

为了让你在学习《数据与数据结构》的过程中获得更大的成功，请浏览本书的栏目介绍。

单元引言、学习目标和单元挑战

从生活经验出发引入本单元将要学习的内容，提出本单元学习要达成的学习目标，预告学习完本单元后要接受的单元挑战。

项目引言和学习目标

描述项目产生的背景和意义，介绍项目学习的主要内容，并提出一些具体问题，引导你带着问题探究。

项目学习指引

通过剖析真实的项目实施过程，帮助你了解学科思想方法，理解相关概念，掌握具体技能。

核心概念和小贴士

解释一些重要概念和术语，或提示相关知识和技术，帮助你抓住重点，扫除认知障碍。

思考与讨论

提出若干问题引导你对技术背后的原理以及人、信息技术与社会的关系等进行思考和讨论。

数字化学习

引导你利用网络、数字化工具和数字资源进行学习。

活　动

提出活动任务，并引导你运用所学知识，使用信息技术工具进行探究、总结和展示。

知识链接

系统整理和归纳本项目的知识要点，方便你学习。

拓展阅读

补充更丰富的阅读材料，开阔你的视野。

单元挑战

布置面向真实情境的项目任务，希望你综合运用本单元所学的知识与技能去解决问题。

单元小结

用思维导图可视化呈现本单元的知识脉络，提供基于学科核心素养的评价表，为你的学习表现进行自我评价。

在学习过程中，希望你勤实践体验、多思考讨论，借助各种数字化工具、资源进行学习与创新，不仅要理解和掌握具体的信息技术知识与技能，还要把握用信息技术解决问题的思想方法，并思考将信息技术应用于社会时所引发的各种挑战，以开放、包容的心态与信息技术、信息社会一起进步。

编　　者

目 录

第一单元

走进数据时代

近年来，随着我国信息化工作的全面推进，以及智能感知技术、计算机技术、网络技术、云计算等信息技术的发展，数据的采集、传输、存储和处理等环节都发生了重大变化。如今各种观探测设备、实验设备、视频监控，如卫星、雷达、天文望远镜、粒子加速器、环境监测系统都装备了智能系统，实现了数据的自动存储和传输，产生了大量数据。智能手机、智能可穿戴设备、物联网，以及社交网络将人的一切“行为”自动记录下来，产生无数的“行为数据”。总之，随着智能技术和网络技术的发展，数据规模发生爆炸性的增长，人类迅速进入数据时代。如何做好海量数据的规范化存储管理、高时效分析运用，同时做好数据安全工作，是数据时代我们所面临的挑战。本单元将带领大家一起走进数据世界，感受数据的价值、数据管理分析和利用的意义，探讨迎接数据时代需要做好的准备。

学习目标

◆理解数字、数值和数据的基本含义。

◆感受数据对人们日常生活、社会经济发展、科学发现与技术进步的影响。

◆认识数据作为新的原材料、生产资料和基础设施的价值与意义。

单元挑战

完成网上购物数据初步分析

项目一

气象数据及其应用

——认识数据的价值

气象信息在保障人民群众生产生活和国民经济发展，促进生态环境保护中的作用日益显著。随着我国气象信息化事业的不断发展，气象领域积累了大量的数据，激增的数据背后隐藏着许多重要的信息。要充分利用这些数据并从中发现有用的信息，推动气象学科的发展和进步，为人民生活提供及时的个性化气象服务，为社会生产提供专业化的气象服务（图 1-1），为做好防灾减灾等公共安全预警和应急处置等提供决策支持，需要气象学家、计算机工程师以及其他相关职能部门的通力合作。

图 1-1　气象服务社会

项目学习目标

本项目将带领大家通过认识气象预报中各种形式的数据来理解数据的基本含义；通过气象预报对人民日常生活的作用，对政府的决策支持作用，对各行各业生产和发展的影响，来感受数据的价值以及数据管理分析的重大意义。

完成本项目学习，须回答以下问题：

1. 什么是数字、数值和数据，它们有何区别？随着信息技术的发展，“数据”的内涵和外延有何变化？

2. 气象数据有哪些方面的应用？有怎样的价值？

3. 数据的开放共享有什么意义？如何迎接数据时代的挑战？

项目学习指引

1. 从气象数据采集到天气预报

常言道："天有不测之风云。"预测天气是一件很难的事。到 19 世纪中叶，基于近代科学的天气预报方法才正式确立。20 世纪中叶，随着计算机技术的发展，世界各国相继采用了数值天气预报方法，预报的准确度得到了极大的提高。

图 1–2 所示天气预报信息中，28.1℃、999.9hPa、0.9m/s、87%、0.1mm、3～4 级，我们把它们称为数据，28.1、999.9、0.9、87、0.1 称为数值，组成数值的 0、1、2、3……9 称为数字。气象要素（温度）、数值（28.1）、单位摄氏度（℃），一起构成了温度这个数据。

图 1–2　常规天气预报信息

天气预报是怎么制作出来的？实际上，天气不仅受到各种气团的影响，还受到当地地形、水域状况等众多因素的影响，任何随机的因素变化都可能引起意想不到的天气变化。看似简单的天气预报信息，其背后都有非常庞杂的数据采集作支撑。从简单的温湿度计到复杂的能见度仪，从蓝天中的探空气球到大海上的浮标站，从矗立在地面的天气雷达到游弋于太空的气象卫星，各种各样的气象观测手段织成一张精密的大网，忠实记录着气象数据，如下页图 1–3 所示。我们每日接收到的天气预报信息，就是由如此庞杂的数据，再加上欧亚甚至全球的所有气象数据，通过筛选、运算、分析等一系列复杂的工作流程得到的。随着预报业务的不断发展，这些数据将更加精准，数量也将继续增加。

小贴士

数值天气预报是目前全世界广泛应用的一种天气预报方法。它根据描述大气运动规律的气体实验定律、水汽守恒定律和热力学能量守恒定律等多个物理定律建立方程组，确定某个时刻大气的初始状态和边界条件后，通过数学方法求解，计算出未来某个时间大气的状态，就是通常所说的天气形势及有关的气象要素如温度、风、降水、辐照度等。

核心概念

数据（data）是对客观事物属性的描述，是记录下的某种可以识别的符号。在计算机科学中，数据是指所有能输入到计算机中并被计算机程序处理的符号的总称。数据承载着信息，把采集到的数据进行加工、分析，可以产生信息。

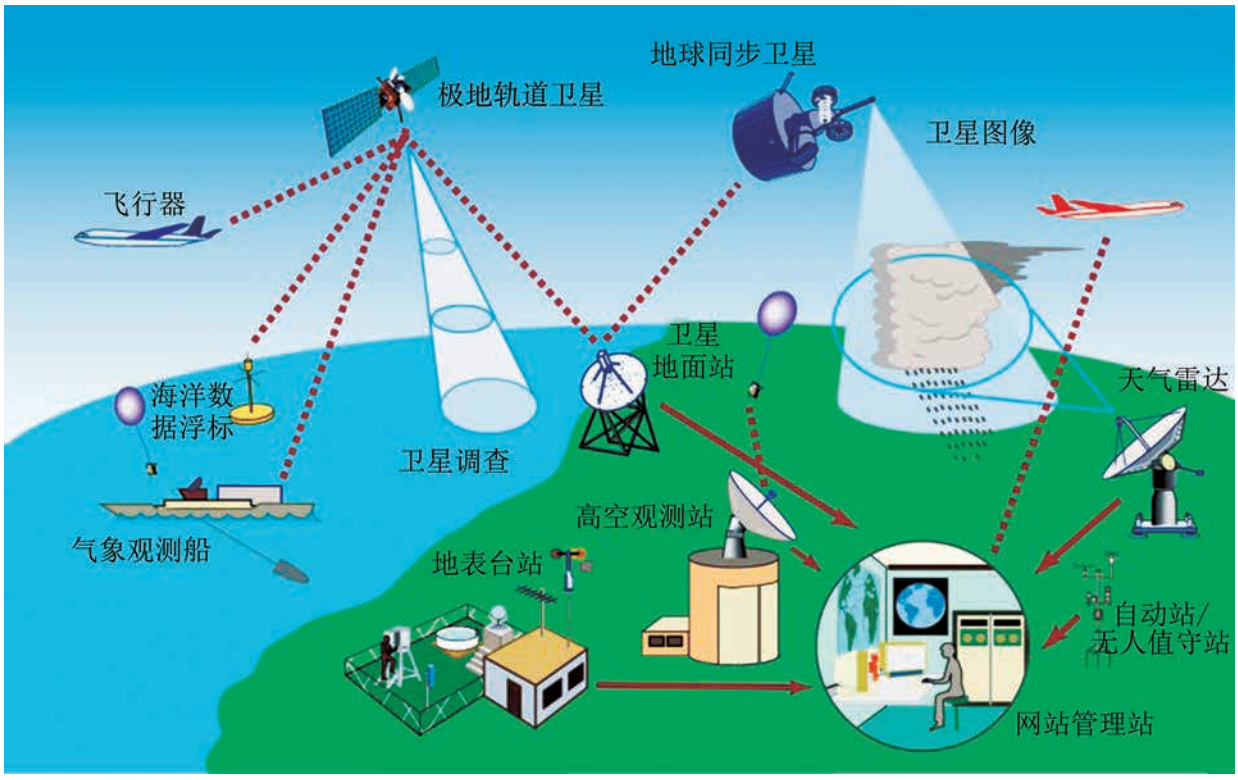


图 1-3　全球观测系统

近年来，随着气象部门观探测能力的不断提升，以及对气象数据的处理能力和可视化能力不断增强，气象部门向公众提供的预报信息越来越精细准确、生动形象，大大提高了气象服务的质量，如图 1-4 所示。

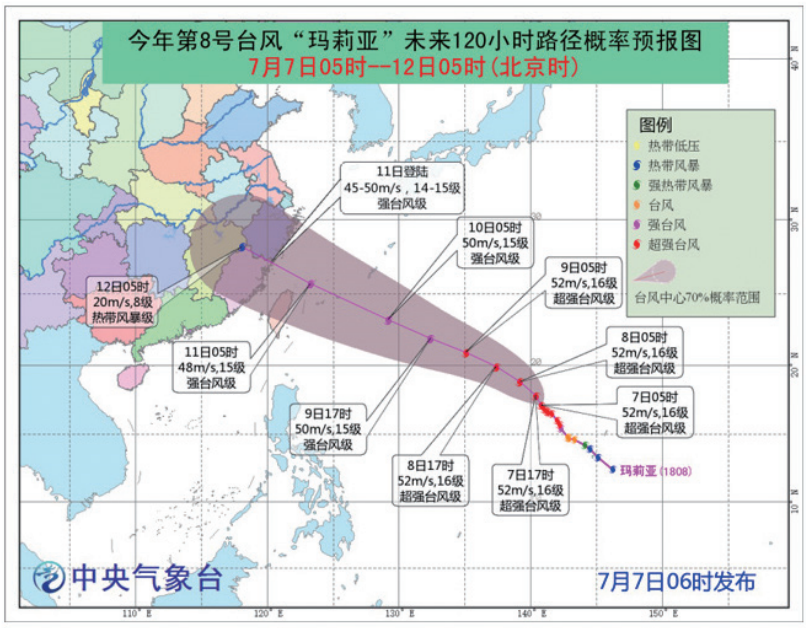


图 1-4　台风路径概率预报图

小贴士

随着信息技术的发展，计算机系统不仅可以处理数值、文本，还可以处理声音、图形、图像、视频等类型的数据。

活　动

1.1　竺可桢是我国当代著名的气象学家和地理学家，是物候学发展的推动者。

（1）收集资料，了解他组建早期中国气象观测网的经历，思考他对我国气象科学发展的贡献。

（2）读他的《大自然的语言》，了解物候观测的内容。思考并讨论：既然可以通过仪器设备采集气象数据，他为什么还要数十年如一日坚持开展物候观察和记录？

1.2　登录中央气象台或中国气象局网站，了解中央气象台或中国气象局向公众提供哪些形式和内容的气象服务。

1.3　收集资料，了解动态气象预报产品是如何制作的，在班级里向其他同学作介绍。

1.4　举例说明随着信息技术的发展，“数据”的内涵有了哪些变化。

2. 气象数据在社会各领域中的应用

随着社会经济的发展和人民生活水平的提高，社会公众、各行各业和国家防灾减灾部门对气象服务的需求日益增长，对天气预报的准确率、精细化程度和预报时效等提出新的、更高的要求。气象部门则充分利用气象观探测数据，提供更好的信息服务和决策参考。

参见 P11　知识链接“数据及其价值”

（1）人们日常生活

随着生活水平的提高，人们越来越注重生活质量，需要更多更方便的渠道获取更详细、更准确、更及时、更个性化的天气气象信息。为了更好地满足这种需求，除了常规天气预报信息外，现在气象部门还对各种气象数据进行综合分析，提供多种预报服务，如图 1-5 所示。

图 1-5　生活气象指数

小贴士

生活气象指数预报是气象部门根据公众普遍关心的生产生活问题和各行各业工作性质对气象敏感度的不同要求，引进数学统计方法，对气压、气温、空气湿度等多种气象要素进行计算而得出的量化预测指标。这些指数是对天气预报的进一步深化。

活　动

1.5　每个生活气象指数的计算需要选择影响因子进行数学建模，再通过计算得出结论。收集资料，了解“中暑指数”“晨练指数”“感冒指数”分别选用哪些气象数据作为影响因子。和同伴探讨交流：这些生活气象指数的计算合理吗？

（2）国民经济建设

我国正处在全面建设更高水平小康社会的重要历史时期，气象服务在国民经济各行各业发展中的作用越来越重要。气象观探测数据一直在为农业、交通、航空、航天、水利、环境、电信、电力和能源等行业提供决策参考。

气象观测数据服务海洋石油工程

海洋石油工程作业对安全保障有极高要求。当出现台风、大风、海雾、强对流天气时，海洋石油平台对准确精细的天气预报服务有着强烈需求，工程作业船迫切需要关于所在区域海上灾害性天气的针对性预警。截至 2016 年底，气象部门已经建设并纳入业务运行的有 373 个海岛自动气象站、41 个锚锭浮标自动气象站（图 1-6）、52 个船载自动气象站、46 个塔台自动气象站和 35 个海上石油平台自动气象站。中国海洋石油集团有限公司也在 16 个平台安装了风、浪、流水文综合观测系统，设法满足企业对海上作业精细化气象保障和海上气象观测资料的需求。

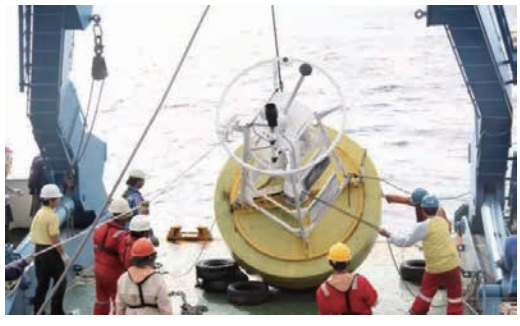


图 1-6　回收海洋气象观测浮标

气象数据助力农业经济发展

农业生产的每个环节都与天气、气候条件密切相关，我国气象部门一直把为农业服务作为基本业务（图 1-7）。气象部门要做到：为农业部门及广大农民提供旱、涝、低温、霜冻等灾害性天气的长、中、短期预报，提示农民在气象灾害到来之前做好防灾准备；利用卫星遥感和地面农业气象网数据，提供作物长势、灾情、土壤水分、天气气候条件等农业气象监测和预测信息，并分析气象条件利弊，提出趋利避害的农业生产管理建议；根据作物长势、面积及气象条件，进行农作物产量预报，定期向国家及各省市提供预报结果，从而为农业经济发展助力。

图 1-7　农业气象观测站

气象数据为青藏铁路提供决策参考

青藏铁路沿线平均海拔在 4000 米以上，闪电、雷暴、风雪和冰雹等灾害性天气事件时常发生（图 1-8）。这些恶劣天气直接加大了青藏铁路沿线的供电隐患。

为了保证青藏铁路的正常、安全供电，气象部门专门为用户加工处理了青藏铁路沿线各气象站多年来关于闪电、雷暴、雪冰雹等非常规气象的资料，形成专题数据集，为制定沿线冻土及融冻地区对输变电工程的接地要求、变电所的防雷标准、沿线多雷区输电线路的耐雷水平和雷击跳闸率等开展专项服务，确保了青藏铁路供电系统的可靠性。

图 1-8　青藏铁路沿线气候多变

（3）科学发现与技术创新

气象观测数据和信息是开展天气预警预报、气候预测预估及各类气象服务、科学研究的基础，是推动气象科学发展的原动力。

GRAPES“隔空指挥”风云四号加密观测

2018 年第 10 号台风“安比”于 7 月 22 日 12 时 30 分前后在上海登陆后，一路北上，数天内给华东、华北及东北地区带来较大风雨影响。

在这次对台风“安比”的路径及风雨预报中，一种创新性的方法首次启用，即 GRAPES 数值预报系统“隔空指挥”风云四号气象卫星（图 1-9）在特定区域内开展加密观测，回传数据实时进入 GRAPES 系统，最终成功改善了对台风“安比”的预报。根据评估，风云四号卫星与 GRAPES 进行配合，24 小时加密观测为精准预报台风“安比”的风雨影响提供了重要定量预报产品支撑，最终成功提高了目标区域内的预报效果。

图 1-9　风云四号气象卫星

数字化学习

通过登录中国气象局等专业网站，了解我国风云卫星的发展历史，思考卫星数据对提高气象预报的准确性和预报时长的积极作用，感受我国在气象科学技术方面的成就。

3. 气象数据，走向未来

近年来，随着人们对“数据是国家基础性战略资源”认识的不断加深，随着我国智慧化城市建设推进过程中对来自各行各业数据资源“融合应用”需求的加大，数据的开放共享工作被推上了快车道。“中国气象数据网”已被建成为气象数据汇集应用的权威大数据平台和中国气象局对外提供气象数据服务的官方门户网站，面向社会和公众提供气象数据服务（图 1-10）。2018 年 1 月，我国首个全球实时海洋观测网正式建成，中国因此成为 9 个有能力向全球 Argo（Array for real-time geostrophic oceanography，地转海洋学实时观测阵）资料中心业务化提交浮标观测资料的国家之一，帮助科学家同步获取全球海洋环境资料。

图 1-10　气象数据共享

气象数据的开放、共享，激发了社会公众、企事业单位、科研机构从海量数据中创造新价值、提升新能力、形成新业态和发现新知识的热情和行动。

基于共享的气象数据，一些瞄准“新零售”业的企业，将天气大数据与物流、仓储、营销管理等信息进行融合分析，从而提高企业的产能，减小和规避气象因素造成的风险。气象数据的开放还有效支撑了环境、国土、水利、农业、林业、海洋、国防和商业等各个领域的业务发展，促进了各行各业对气象数据的应用价值和效益的共同挖掘。

开放数据创新应用大赛（SODA）-2017- 未来之星奖

“航延预报——基于航延预测模型优化乘客航空出行选择”项目获 2017 年 SODA 未来之星奖，它的特点是让航班延误预测像天气预报一样简单。它主要基于气象数据、航班数据、机场流量数据等跟航班延误相关的大数据处理，引入深度学习、机器学习等技术建立航班延误预测模型，通过不同模型的综合运算及分析，得到较为理想的航班延误预测结果。航延预报可成为类似天气预报的便民应用，乘客可以便捷地使用网站、APP 等获取航班延误预测信息，理性选择出行航班及时间。保险公司可以根据航班延误预测模型设计航延险产品，由传统航延险的事后损失补偿理念升级为事前风险预警。航空公司也可以参考航班延误预测情况，合理调配运力，缓解旅客滞留情况。

活　动

1.6　登录“中国气象数据网”（国家气象科学数据中心官网），了解其主要提供哪些数据内容和形式的数据服务。

1.7　登录“中国气象数据网”下方的相关链接，调研其他数字共享中心。

	共享的数据种类	提供的数据产品	专题数据服务
国家气象科学数据中心			
国家农业科学数据共享中心			
地震科学数据共享中心			

1.8　收集资料，跟踪了解我国有关大数据发展的政策文件、大数据技术的发展前沿，以及大数据技术应用的相关案例，尝试在学校校园新闻中设立“大数据快讯”栏目。

4. 数据时代的机遇和挑战

数据，已经渗透到当今每一个行业和生活的方方面面，正在改变人们的生活、工作和思维方式。

数据被认为是一种新的原材料，可以用来加工、产生价值；是一种新的生产资料，可以提高生产的效率；是一种新

的基础设施，投资和利用它可以改善经济和民生。

参见 P12 知识链接“数据素养”

思考与讨论

结合前面关于气象数据应用的例子，思考：为什么说数据是一种新的原材料、一种新的生产资料和一种新的基础设施？

数据时代在给人类带来福音的同时，也对人类驾驭数据的能力发起了一场新的挑战。

首先，数据泄露对国家安全、组织机密和个人隐私保护等方面都构成巨大挑战，如何在利用数据价值的同时，保证数据安全，成了国家、企事业单位和个人不得不面对的难题。

其次，数据已渗透到每一个行业和业务职能领域。无论是数据的采集、高效存储和管理，还是对数据的有效挖掘、可视化展示和创造性应用，都需要具备数据素养的人才。作为一名高中生，我们要努力掌握一定的数据科学基础知识，提高数据素养，从容投身于数据时代。

活　动

1.9 阅读图 1-11，结合日常生活中所涉及的数据安全问题，谈谈现代公民应该具备怎样的“数据素养”。

图 1-11　数据安全报告

知识链接

数据及其价值

数据是计算机加工的基本对象，是现实世界中各种事物和现象的抽象化和符号化。伴随着信息技术的发展，计算机可处理的数据类型越来越多，现代计算机处理的不再是单纯的数值型数据，更多的是文本、图形、图像、音频、视频等非数值型数据。

数据是事物属性的刻画，反映出事物的信息。通过对数据的挖掘，可以发现数据里面所隐藏的各种信息，找到数据规律并挖掘出所隐含的自然或社会规律。因此，信息技术的高速发展所带来的海量数据和信息量无疑是一座重要的宝库。

从移动支付到共享经济，大数据正在加速重塑大众生活的诸多方面；从万物互联到智慧城市，大数据正在深刻影响着经济发展、社会治理、国家管理的各个领域。上海全面推广“健康云”，病历和就诊数据汇集“上云”，就诊记录一键查询，转诊信息顺畅共享；重庆江北区智慧城管系统为路灯、排水管网、环卫车等加装智能设备，采集到的运行状态数据由“智慧城管”进行精细的分析处理；全国多地交管部门联合高德地图，借助大数据分析技术为春运“摩托大军”提供最佳返乡路线、个性化路况提示，让春运更有“温度”；全国首个旅游大数据公共服务平台“杭州旅游数据在线”上线，游客通过手机便可了解景点实时拥堵度、酒店好评率等信息……

大量用户数据与信息催生了一系列消费者行为的研究与分析。企业利用用户数据可以给用户画像，对用户进行细分，精准感知用户的需求，从而基于数据优化产品设计，为用户提供更好的服务。制造业通过工业大数据与自动控制和感知硬件、工业核心软件、智能服务平台等融合发展，形成数据驱动的工业发展新模式，对每个产品从生产、销售到用户使用环节的数据采集、分析和应用，提高生产效率、减少库存、延伸产业链。

在科学研究领域，科学数据的采集、传输、存储和处理等环节都发生了重大变化。如今各种观测、实验设备都装备了智能系统，实现了数据的智能采集和管理。特别是数据可以实现远程共享。有专家提出，数据规模及其采集方式的不同，让数据挖掘成了科学发现的一种重要工具。“在 21 世纪，人们通过各种新工具不间断地采集着海量的科学数据，也通过计算机模型产生着大量的信息，其中大部分已经长期存储在各种在线的、可以公共获取的、得到有效管理的系统上，可以支持持续的分析，这些分析将引发许许多多新理论的发现。”

跨行业、跨领域数据的开放共享、综合利用和协同创新，给基于数据解决社会问题带来了新的希望。如通过分析气象报告、潮汐相位、地理空间、卫星图像等海量数据，优化风力发电机的布局，合理进行电力系统调度、提高风力发电效率，确保电力系统运行稳定；通过手机信令数据和呼吸道传染疾病医疗数据，分析该疾病暴发的时间、地点以及传播模型，为该病的预防和控制服务。

随着数据成为国家的战略资源、企业的核心资产，同时对个人也越来越重要，数据也成为违法犯罪分子的重点关注目标，数据安全问题受到全世界从政府到普通公众的重视。此外，规模庞大的数据中心在成为全球强大的经济引擎的同时，也消耗了巨大的能量。数据中心绿色建设、高效运营已是迫在眉睫的任务。

数据素养

在数据公开、数据交换、数据共享和数据利用成为时代特征时，不论是政府机构、企业还是个人，都在创造数据、管理数据和使用数据，数据素养的重要意义和价值日益凸显，并迅速引起学界关注。

国内关于数据素养并未形成统一的概念定义。有专家认为数据素养通常指的是研究者在工作中对科学数据的采集、组织管理、处理分析、共享等过程中应具备的能力，还应包括研究者在数据生命周期中普遍遵循的道德与行为规范；也有专家认为数据素养包括三个层次：数据意识、数据基本知识与技能、数据利用能力等。综合已有研究，数据素养应包括对数据的敏感性，数据收集、处理、分析、判断和利用的能力，尊重数据伦理、保证数据准确、安全和隐私的修养。数据素养与信息素养概念密切相关，数据信息素养是信息素养的深化与拓展。

每个公民只有具备必要的数据素养，才能紧跟社会发展的步伐。

拓展阅读

智慧城市建设需要“数据活化”

智慧城市在城市大发展中有两大功能，一是解决城市已经客观存在的“不健康”问题，二是提供让城市更舒适、更宜居的服务。智慧城市建设是利用信息技术，不断获取、活化和分析城市中的多种异构数据，从而解决城市所面临的各种挑战，如环境恶化、交通拥堵、能耗增加、规划落后等。智慧城市将无处不在的感知技术、高效的数据管理和分析算法，以及新颖的可视化技术相结合，以提高人们的生活品质、环境质量和城市运转效率。

如果将城市虚拟为一个“人”，那么采集城市感知数据的过程，就像为城市做全面的体检，将城市中的人口、交通、规划、能源、经济、环境等诸多信息详细地展示出来，为保障城市的健康发展以及“城市病”的及时、准确诊断提供了丰富的素材。在综合应用城市多源数据的基础上，机器学习、数据挖掘、大数据分析等计算机信息技术，可以在城市领域知识的指导下，深入挖掘并理解城市各种“不健康”问题的成因与机理，从而提出科学合理的调整方案。大数据与计算机技术同城市科学领域的专业知识相结合，是智慧城市走向科学的基础。

——摘自《国家治理》2015 年 18 期《以信息技术服务业推动智慧城市建设》

让“沉睡”的海洋观测资料活起来

全球海洋观测平台既有锚锭浮标、志愿观测船、石油平台等海表观测，也有剖面浮标（Argo）、深海潜标等深海观测；观测要素丰富，既包含常规的风、温、压、湿气象要素，也包

括海温、海浪、海流、盐度等水文要素。据国际海洋学与海洋气象学联合委员会（JCOMM）统计，2015 年业务运行的全球海洋观测平台有 8500 余个，这些观测资料通过世界气象组织全球通信系统（GTS）实现全球各大业务中心共享。长期以来，大部分资料都在资料库中“沉睡”，并没有被挖掘整理并应用于全球海洋监测业务中。

2015 年中国气象局、国家气象信息中心，组建海洋气象观测资料搜集整理小组。通过对目前信息中心资料库的海洋气象观测数据进行分门别类，共整理出漂流浮标、剖面浮标、波浪浮标、潮位站、石油平台站、海啸浮标、志愿船观测、潜标、锚锭浮标共九大类 8000 余个观测站，收集整理上述观测站的元信息数据，将成果通过“全球海洋气象共享系统”实现共享，方便了下游用户使用。在此基础上，进一步与国内锚锭浮标、海岛站、石油平台站、船舶站等观测数据整合，有效提升全球海洋气象监测能力，推进了我国海洋气象观测资料应用水平。

——摘自中国海洋网

单元挑战　完成网上购物数据初步分析

一、项目任务

张女士在某网上书店为自己买了一本《优雅女人的投资理财》，给 6 岁女儿买了一本《写给儿童的成语故事》。张女士此次购买数据信息被商家共享给了合作伙伴，被共享的数据为：（1）张女士对书籍感兴趣；（2）她家有个孩子；（3）她是通过网络购买的；（4）她通过网络看到广告；（5）住在上海；（6）……

通过对上面初步记录的数据进行分析可推测：（1）张女士可能会购买理财产品；（2）张女士可能会购买健身会所的会员卡；（3）张女士可能会为女儿报兴趣班；（4）……

1. 分小组讨论，为上述案例再添加一条被共享的数据和推测结果。

2. 请将本案例中有可能涉及张女士个人隐私的信息（至少三条）列在下表中。

序号	可能的隐私信息
1	
2	
3	
4	

3. 某购物平台内部有一个专门的数据安全小组，负责监管隐私问题，如要查某年用户有多少，数据查询结果不会显示个人的全部身份证号码；在对外合作中，提供脱敏的数据，即只告诉数据分析运算的结果，而不提供元数据。结合本案例，列出大数据存在哪些风险，分享你或你们小组关于如何有效保护数据隐私权的建议。

二、项目指引

1. 从自己或身边人的生活体验中，了解网上购物的过程。

2. 了解在网上购物的过程中会产生的数据。

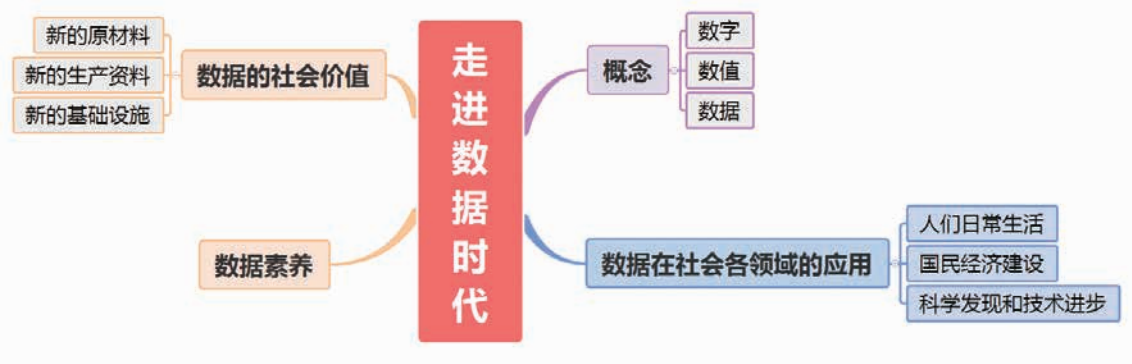
3. 收集身边或媒体关于个人隐私泄密的案例。

三、交流评价与反思

以自己熟悉的信息表达工具（如演示文稿等）制作电子作品，通过网络或课堂展示交流分析结果，并对他人的作品进行评价，谈谈数据的应用价值和负面影响。

单元小结

一、主要内容梳理



二、单元练习

1. 当气温降到 16℃时，短大衣开始畅销；气温降到 10℃时，长大衣开始畅销；某啤酒商发现，夏季气温每上升 1℃，啤酒销量就会增加 230 万瓶。以上案例说明，气象数据可以是预测生产流通和消费的风向标，反映了气温变化与销售额增减的关系。请从身边寻找相关气象数据，分析该数据与社会某些领域的关系。

2. 有些创新企业的生产原材料就是数据，通过技术手段将收集到的数据加工，进行数据分析，生产出形形色色的“数据产品”，从而获得收益。某订餐网站的外卖时段占比和订单量占比，如图 1-12 和图 1-13 所示。请从图中分析，这些数据可以给企业提供哪些方面的决策参考。

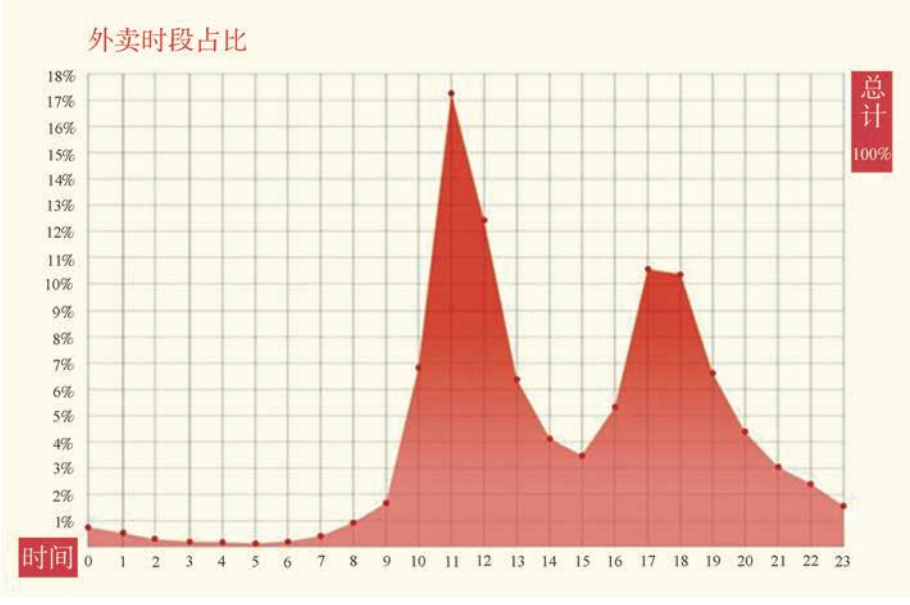


图 1-12　外卖时段占比

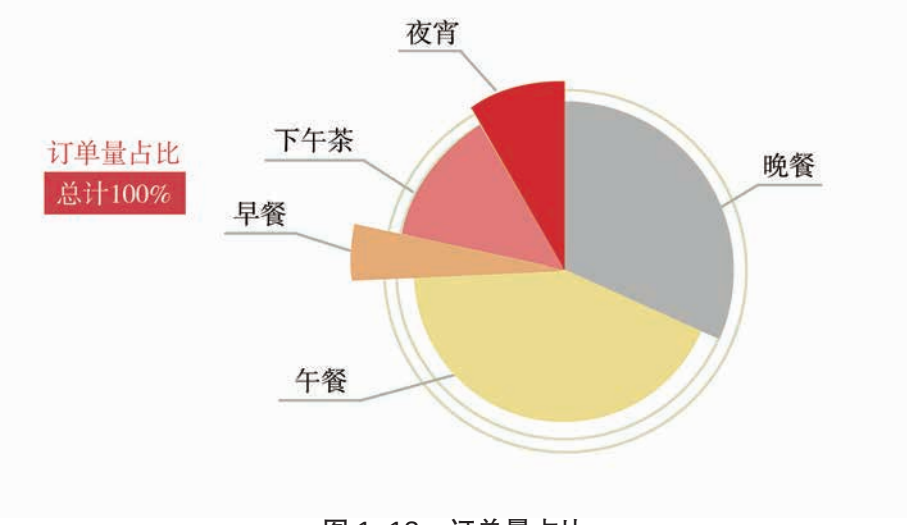


图 1-13　订单量占比

3. 汽车生产企业的产业全过程都与数据密不可分，这些数据让其在开发新车和开拓市场的核心业务上能够及时采取相应的行动并获益。查阅资料，请从市场角度说明可以收集哪些数据，为汽车生产企业的新车研发提供依据。

三、单元评价

评 价 内 容	达 成 情 况
能说出数字、数值和数据的涵义及它们的区别（T）	
能列举气象数据在人们日常生活中的应用（A）	
能列举气象数据在国民经济建设中的应用（A）	
能列举气象数据在科学发现和技术进步中的应用（A，I，R）	
能理解数据开放共享的意义（A，R）	
能洞悉数据时代的机遇和挑战（A，R）	
能养成良好的数据素养（A，R）	
能理解数据是一种新的原材料，一种新的生产资料和一种新的基础设施（A，R）	

说明：A—信息意识，T—计算思维，I—数字化学习与创新，R—信息社会责任

第二单元

初识数据结构

随着信息技术的不断发展，计算机的应用越来越广泛，已经从最初的科学计算发展到数据处理、自动控制、办公自动化、人工智能等许多非数值计算领域。计算机加工处理的对象也从纯粹的数值发展到字符、表格、图片等各种具有一定结构的数据。为了对大量数据进行高效处理，必须要先分析数据的内在联系，合理地组织、存储数据，然后设计算法编写程序实现相关处理。而如何合理地组织、存储数据就是“数据结构”主要研究的问题。本单元将从实例出发，学习数据结构的基本概念，初步认识数据结构在问题解决中的重要作用，进一步提升计算思维。

学习目标

◆理解数据结构的概念，认识数据结构在解决问题过程中的重要作用。

◆理解线性表数据结构的概念，并能编程实现其相关操作。

◆比较数组和链表的区别，明确上述两种数据结构在存储不同类型数据中的应用。

◆理解抽象数据类型的概念，认识抽象数据类型对数据处理的重要性。

单元挑战

实现学校学生健康情况登记表的操作

项目二

研究学校教学管理相关数据的组织处理
——初识数据结构

随着学校管理信息化的不断发展，使用计算机进行教学管理逐步成为常态。学校教学管理涉及许多方面的数据，如学生数据、教职员工数据等。如果要用计算机处理这些数据，就要先分析它们各自的内在关系（图 2-1），然后采用相应的存储方式将这些数据存储到计算机中，在此基础上进行数据的相关操作，如查找、插入、删除、合并等，以实现管理需求。

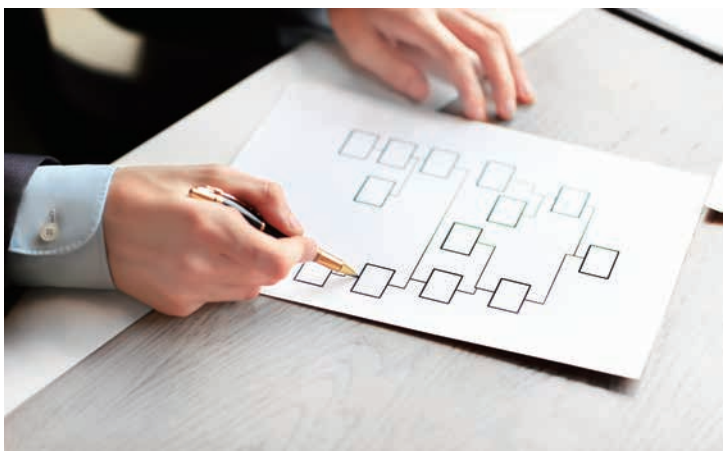


图 2-1　分析数据内在关系

项目学习目标

在本项目中，我们将一起通过研究学校教学管理相关数据的组织处理来初步认识数据结构。

完成本项目学习，须回答以下问题：

1. 学生基本数据之间有何种内在关系，处理这些数据的时候可以如何存储？

2. 班级管理组织结构属于何种逻辑结构？

3. 什么是数据对象、数据元素和数据项？

4. 什么是数据结构？数据结构在解决问题过程中有何作用？

5. 什么是抽象数据类型？抽象数据类型的作用是什么？

项目学习指引

1. 从教学管理相关数据认识数据的逻辑结构

学校为了对学生进行管理，每年新生入校都要登记注册各种信息，诸如姓名、性别、出生日期、家庭地址等。学校要为每位新生分配班级学号。中途学生转学转班，学校要删除或修改学生信息。学生的基本情况，可以用学校编制的“学生信息表”表示，如表 2-1 所示。

表 2-1　某学校学生信息表

学号	姓名	性别	出生日期	……
20140111	杨阳	男	1998.10	……
20140112	卢声凯	男	1999.03	……
20140113	林德康	男	1999.01	……
20140114	王诗萌	女	1999.07	……
20140115	冯子晗	女	1999.04	……
……	……	……	……	……

表中的每一行构成一个学生的一条信息，包括学号、姓名、性别、出生日期等。可以看出，除第一个和最后一个学生外，每个学生都分别与前一个学生和后一个学生相邻，形成一对一的关系。这张按一定顺序排列的表，就是解决学生管理问题的模型（线性表，将在后续项目详细展开介绍）。

当学生信息被输入计算机后，就成为计算机处理的对象。表中每一行代表一条学生基本数据，称为数据元素，它由学号、姓名、性别、出生日期等组成，其中的每一项称为数据项。所有各条学生基本数据一起构成一个集合（学生基本数据表），称为数据对象。数据元素之间存在一对一的关系的逻辑结构称为线性结构。

生活中还有很多这样的例子，如员工管理系统、订票系统等。在这类问题中，一个共同特点是所处理的对象之间存在简单的一对一的线性关系。基于此，可以获得解决该类问题的数学模型。通过设计算法，计算机能够完成对这些数据

小贴士

数据元素（data element）是数据的基本单位。一个数据元素由若干个数据项组成。在不同的条件下，数据元素又可称为元素、结点、顶点、记录等。

数据项（data item）是组成数据元素的最小单位（也可称为字段、域、属性）。

数据对象（data object）是性质相同的数据元素的集合。

逻辑结构是指数据元素之间的相互关系。

参见 P26 知识链接“基本概念术语”

小贴士

数学模型是指，从实际问题中提取操作对象，并找出这些操作对象之间的关系，然后用数学语言做出描述。有些问题的数学模型可以用具体的数学方程表示，更多的实际问题无法用数学方程表示，这就需要对数据进行分析得到解决问题的方法。数据的逻辑结构也是从具体问题抽象出来的数学模型。

元素查找、插入和删除等操作。这就是一类数据结构——线性数据结构。

除了学科教学工作外，学校还有许多教学管理工作。为了提高管理效率，须按照一定的工作任务和目标，将成员按不同的工作性质、职务、岗位组合起来，形成层次恰当、结构合理的有机整体。以班级管理为例，一般学校都设有如图 2-2 所示的组织管理结构：

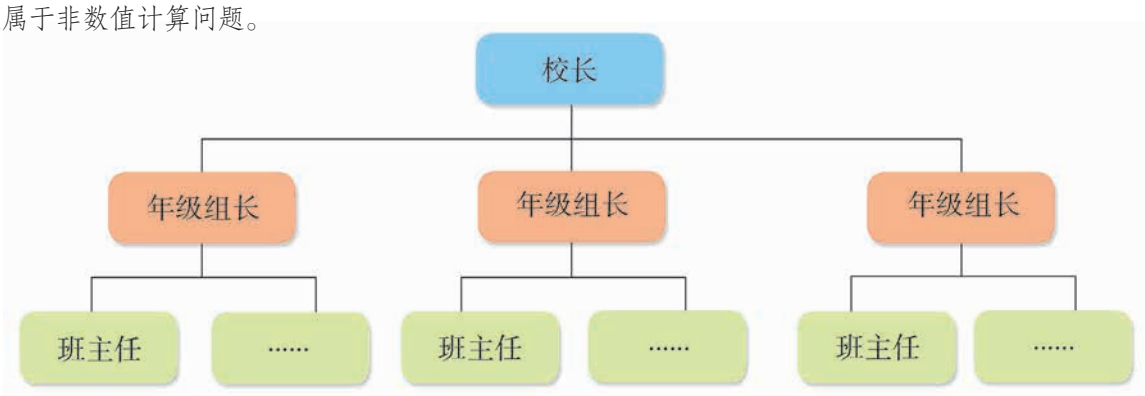


图 2-2　班级管理组织结构示例

其中，校长（分管校长）领导各年级组长，各年级组长分别领导本年级各班班主任。他们之间存在着一对多的关系，所构成的逻辑结构称为树形结构。逻辑结构为树形结构的数据结构属于非线性数据结构。

参见 P27 知识链接“逻辑结构”

核心概念

数据结构（data structure）是相互之间存在一种或多种特定关系的数据元素的集合，涉及逻辑结构、存储结构及运算（操作）三个方面。

小贴士

学生管理、班级管理等问题属于非数值计算问题。

活　动

2.1　请列举生活中其他常见的线性结构。

2.2　请了解本校学科教学管理的组织结构，并画出结构图。

2.3　在计算机和人下井字棋的游戏中，计算机操作的对象是对弈过程中可能出现的棋盘状态，称为格局，每下一步产生的格局都可以派生出多个格局（下一步的可能走法），请以图 2-3 为当前格局画出后续所有的格局关系图，说说该图所示的是一种什么逻辑结构，为什么？

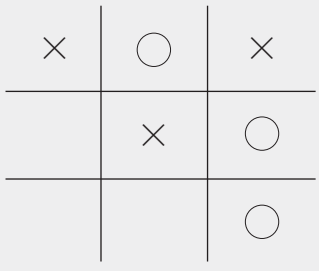


图 2-3　井字棋，格局

2. 了解教学管理相关数据的存储结构

上述学生信息表存储到计算机内时，不仅要存储每一条学生基本数据，还要借助它们在存储器中的相对位置来表示线性关系。表 2-2 所示的是学生信息表的一种存储结构。

小贴士

存储结构是指数据的逻辑结构在计算机中的表示，也称为物理结构。

表 2-2　“某学校学生基本数据”存储结构（一）

存储地址	某学校学生基本数据				
200	20140111	杨阳	男	1998.10	……
230	20140112	卢声凯	男	1999.03	……
260	20140113	林德康	男	1999.01	……
290	20140114	王诗萌	女	1999.07	……
320	20140115	冯子晗	女	1999.04	……
……	……	……	……	……	……

假设存储地址为一个十进制数（实际存储地址是一串二进制数），首地址为 200，每条学生基本数据占用 30 个存储单元（一个存储单元的大小为一个字节），即一条数据的存储空间大小为 30 个字节。

从图中看到数据元素是按照学生信息表中的顺序依次存储在地址连续的存储空间中的，一对一的逻辑关系从存储位置的前后顺序能直接反映出来。如第一个学生杨阳的数据存储在地址为 200 的存储空间内，它后续的卢声凯的数据就存储在地址为 230 的存储空间内，以此类推，依次由低地址向高地址存储，直到存储好最后一个数据元素。

参见 P27 知识链接“顺序存储结构”

思考与讨论

如果存储地址不连续，是否能表示线性关系？

除了顺序存储结构外，还可以用其他方式来存储数据元素。假定给数据元素（每条学生记录）附加一个后继结点的地址，用于存放下一个数据元素的存储地址，则可得到表 2-3 所示的存储结构。

表 2-3 “某学校学生基本数据”存储结构(二)

地址	某学校学生基本数据					后继结点的地址
100	20140111	杨阳	男	1998.10	……	350
150	20140115	冯子晗	女	1999.04	……	0
210	20140114	王诗萌	女	1999.07	……	150
350	20140112	卢声凯	男	1999.03	……	500
500	20140113	林德康	男	1999.01	……	210
……	……	……	……	……	……	……

参见 P28 知识链接“链式存储结构”

在这种存储方式下，数据元素的存储地址可以是连续的，也可以是不连续的。每个存储空间存储了数据元素(称为数据域)和后继结点(下一个数据元素)的存储地址(称为指针域)，地址 0 表示结束。这一存储方式也可以用图 2-4 表示。这种存储结构称为链式存储结构。

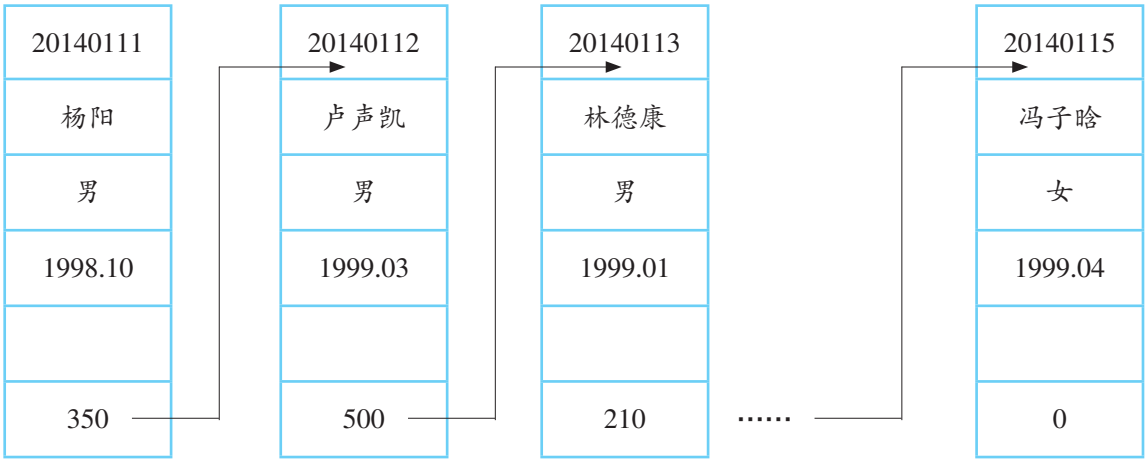


图 2-4 链式存储结构

思考与讨论

链式存储结构是否一定需要从低地址向高地址存储?

明确了学生基本数据的逻辑结构和存储结构之后，就可以对其进行操作，如插入、删除、查找等(在后续项目中展开学习)。

活　动

2.4　请画出采用顺序存储结构和链式存储结构两种不同的方式存储本班学生信息的示意图。

顺序存储结构	
链式存储结构	

3. 了解数据类型和抽象数据类型

计算机进行教学管理需要做诸如学生信息的增、删，或学生成绩统计等工作。这些工作完成的前提是要把学生信息或成绩等数据储存在计算机内存中，同时要给出指令，“告诉”计算机针对不同的数据对象“做什么”和“怎么做”。

计算机的内存容量是有限的，而做两个个位数的加法或两个位数不同的小数的加法，显然需要的空间大小可以不同。计算机研究者通过对不同数据进行分类的方法——数据类型，来描述不同数据的集合，为不同类型的数据分配了大小恰当的内存空间。

所有高级语言都定义了一系列的数据类型。

以 Python 语言为例，基本数据类型也可以分为：

• 原子类型——数字型（numbers，包括整型 int 和实数型 float）、字符串型（string）

• 结构类型——元组（tuple）、列表（list）、字典（dict）

教学管理数据中，班级学生人数是整型，学生成绩是实数型，学生的姓名是字符串型等。

思考与讨论

你还知道教学管理数据中哪些是整型？哪些是实数型？哪些是字符串型？

核心概念

数据类型是指一组性质相同的值的集合及定义在此集合上的一些操作的总称。

参见 P28 知识链接“数据类型”

小贴士

数据类型用来说明一个数据在数据分类中的归属。它是数据的一种属性。这个属性限定了该数据的变化范围。数据类型是被定义在程序设计语言中的，尽管不同的高级语言所定义的数据类型不尽相同。

除了上述基本数据类型外，Python 语言还通过定义类（class）来实现结构类型。例如，用“class student：”就可以定义包含学号、姓名等多个数据项的结构类型。这时，student 就相当于是一种记录类型，student 的变量（一般称对象）就可以存放学生信息数据元素了。

数据类型还有一个作用是定义了对数据的一些操作。这些操作在程序设计语言中是直接使用运算符或函数来实现的，如将班级学生人数相加得出年级学生人数，在 Python 中为 T=a1+a2+a3+a4（假设有 4 个班级，每个班级的人数分别为 a1、a2、a3、a4），这就是基于整数类型上的一种操作（加法运算）。计算机编程者在编程时，不需要关心整数在计算机中是如何表示的，计算机是如何分配相应的存储空间，如何实现加法操作的。

思考与讨论

整数型、实数型、字符串型通常定义了哪些操作？你使用过哪些？

事实上，各种计算机，不管是大型机、小型机、PC、平板电脑、PDA，甚至是智能手机都拥有“整数”类型，也需要整数间的运算，实现方法可能有所不同，但在计算机编程者看来，它们都是相同的，原因就在于整数类型定义的数学特性相同。这就是抽象的意义。从这个层面来看，整型其实是一个抽象数据类型。

思考与讨论

其他数据类型是抽象数据类型吗？为什么？

当然，抽象数据类型不仅仅指已经定义并实现的数据类型（如整型、字符串型等），还可以是计算机编程者对现实问题进行抽象后，在设计软件程序时自己定义的数据类型。以学生信息管理问题为例，对其进行抽象后，可以得出数据对象是学生信息这一数据元素的集合，此集合中数据元素之间的关系是一对一的线性关系。如果在此数学模型基础上定义插入、删除等一组基本操作，就形成一种抽象数据类型。该抽象数据类型可以如下所示：

小贴士

抽象是指抽取出事物具有的普遍性的本质。它是抽出问题的特征而忽略非本质的细节。是对具体事物的一个概括。抽象是一种思考问题的方式，它隐藏了繁杂的细节，只保留实现目标所必需的信息。

核心概念

抽象数据类型（abstract data type，ADT）是指一个数学模型以及定义在此数学模型上的一组操作。

```
ADT List:
    数据对象：D={a_i |a_i ∈ ElemSet, i=1,2,…,n,n>=0}
    数据关系：R={<a_i-1,a_i>| a_i-1,a_i ∈ D, i=2,…,n}
    基本操作：
        def InitList(self) # 建立一个空的表
        def GetElem(self,i) # 返回表的第 i 个元素
        def Length(self) # 求表的长度
        def LocateElem(self,x)
              # 求元素 x 在表中的位置；若不存在 x，则返回 0
        def Insert(self,i,x) # 在表的第 i 个位置上插入一个新元素 x
        def Delete(self, i) # 删除第 i 个元素
```

数据对象的定义

数据关系的定义

基本操作的定义

抽象数据类型中被定义过的基本操作，是通过编写出相应的程序模块实现的。以后用计算机解决此类问题时，凡是要用到这些基本操作，直接调用这些程序模块即可，而不必每次重新编写程序，大大降低了程序员的重复劳动。

参见 P29 知识链接“抽象数据类型”；P29 知识链接“抽象数据类型的表示”

例如，将某两个班级的学生数据定义为上述抽象数据类型后，调用其基本操作（如下面代码中框出的部分）就可以实现将两个班学生数据的合并（用 Python 语言编写）。

```
def union(La, Lb):    # 本函数功能是将表 Lb 合并到表 La 中
    La_len = Length(La)    # 求表 a 的长度
    Lb_len = Length(Lb)    # 求表 b 的长度
    for i in range(1, Lb_len+1):
        x = GetElem(Lb, i) # 获取表 b 的第 i 个数据元素
        if LocateElem(La, x)==0:
            La_len=La_len+1
            La.Insert(La, La_len,x) # 将数据元素插入至表 a 中
    return
```

思考与讨论

1. Python 中列表属于抽象数据类型吗？为什么？

2. 使用抽象数据类型有何好处？

活　动

2.5　假定把矩形定义为一种抽象数据类型，其数据部分包括矩形的长度和宽度，操作部分包括初始化矩形的尺寸、求矩形的周长和矩形的面积。请完成矩形的 ADT（抽象数据类型）描述。

提示：假定该抽象数据类型名用 Rectangle（矩形）表示，定义矩形长度和宽度的数据用 length 和 width 表示，并假定其类型为浮点（float）型，初始化矩形数据的函数名用 InitRectangle 表示，求矩形周长的函数名用 Circumference（周长）表示，求矩形面积的函数名用 Area（面积）表示。

知识链接

基本概念术语

1. 数据

数据是对客观事物的描述，是记录下来的某种可以识别的符号，在计算机科学中，数据是指所有能被输入计算机中，且能被计算机处理的符号的集合，是计算机加工处理的对象。这些符号必须具备两个前提：可以输入到计算机中和能被计算机程序处理。

例如，学生基本信息输入到计算机中后，可以通过计算机程序进行插入、修改等处理。

数据不仅仅包括数值型数据，还包括字符、图像等非数值型数据。

2. 数据元素

数据元素是组成数据的、有一定意义的基本单位，是数据这个集合中的个体，也被称为记录。如表现在“学生基本信息表”中，就是某一学生的一条记录。

3. 数据项

数据项是组成数据元素的、有独立含义的、不可分割的最小单位。例如，“学生基本信息表”中每个学生的学号、姓名、性别等都是数据项。

4. 数据对象

数据对象是性质相同的数据元素的集合，是数据的子集。例如，整数数据对象是集合 N={…，−2，−1，0，1，2，…}，字母字符数据对象是集合 C={A，B，…，Z，a，b，… z}，而学生基本信息表也是一个数据对象。

数据结构

数据结构是相互之间存在一种或多种特定关系的数据元素的集合，涉及逻辑结构、存储结构及运算（操作）三个方面。

1. 逻辑结构

逻辑结构是指数据对象中数据元素之间的相互关系。它与数据的存储无关，是独立于计算机的。数据的逻辑结构可以看作是从具体问题抽象出来的数学模型。

根据数据元素之间关系的不同特性，通常有四类基本结构，如图 2-5 所示。它们的复杂程度依次递进。

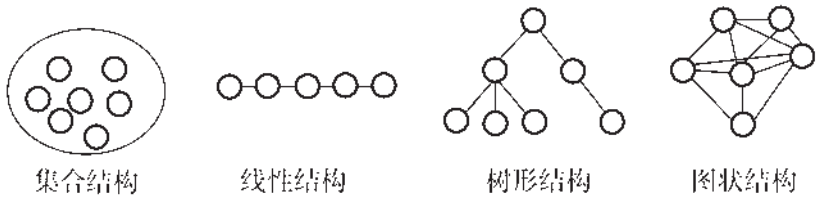


图 2-5　四类基本逻辑结构关系图

（1）集合结构。

这种结构的数据元素除了同属于一个集合外，它们之间没有其他关系。各个数据元素是“平等”的，它们的共同属性是“同属于一个集合”。例如，一组随机没有规律的数字组成的集合，就是一个集合结构。

（2）线性结构。

这种结构的数据元素之间是一对一的关系。例如，把学生信息数据按照其入学报到的时间先后顺序进行排列，将构成一个线性关系。

（3）树形结构。

这种结构的数据元素之间存在一种一对多的关系。例如，在班级的管理体系中，班长管理多个组长，每位组长管理多名组员，从而构成树形结构。

（4）图状结构或网状结构。

这种结构的数据元素是多对多的关系。例如，若任意两个城市之间有直线或间接的通信线路，就可构成图状结构。

其中，树形结构和图状结构属于非线性结构。

2. 存储结构

存储结构是指数据的逻辑结构在计算机中的表示，即数据元素及其之间的关系在计算机中的表示，也称为物理结构。

如何反映数据元素之间的逻辑关系，是实现存储结构的重点和难点。

数据元素在计算机中有两种最基本的存储结构：顺序存储结构和链式存储结构。数据元素在计算机内可以用一个结点来表示。

（1）顺序存储结构。

顺序存储结构是把数据元素按顺序存放在地址连续的存储单元中，其数据之间的逻辑关系和存储关系是一致的，即借助数据元素在存储器中的相对位置来表示数据元素之间的逻辑关系。计算机会在内存储器中开辟一段地址连续的存储单元空间依次存放数据，即第一个数据放在第一个位置，第二个数据放在第二个位置……如图 2-6 所示，其中 1001，

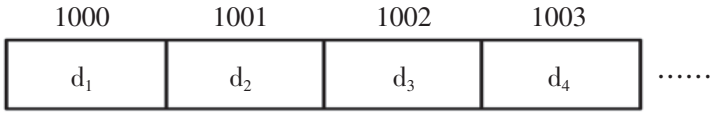


图 2-6　顺序存储结构示意图

1002…表示存储地址，d_1，d_2…表示数据。由于只要知道了首地址，就可以随机存取任意位置上的数据元素，所以也可以称顺序存储结构为随机存取存储结构。

（2）链式存储结构。

链式存储结构无须占用一整块存储空间，它把数据元素存放在任意的存储单元中。链式存储结构不要求逻辑上相邻的数据元素在物理位置上也相邻。数据元素之间的关系借助于指针来表示，即给每个数据元素附加一个指针用于存放后继数据元素的存储地址，这样通过这个存储地址就可以找到相关数据元素的位置，如图 2-7 所示。

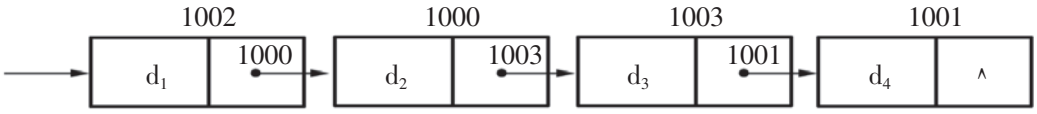


图 2-7　链式存储结构示意图

显然，链式存储结构存放数据时要比顺序存储结构灵活，不用关心数据存在哪里，只要有一个相应的存储地址就能找到它了。

总之，逻辑结构是面向问题的，而存储结构是面向计算机的，其目的是将数据及其逻辑关系存储到计算机内存中。

3. 运算

数据的运算也称为操作，主要包括对数据进行删除、插入、访问、修改和查找等。

数据结构的作用

如今需要用计算机解决大量的非数值计算问题，这些问题通常不能通过列方程、解方程等数学方法求解，而是需要用诸如线性表、树和图之类的数据结构来描述。求解这类问题的做法通常是：

（1）从具体问题抽象出一个适当的数学模型；

（2）设计一个解此模型的算法；

（3）编程并进行调试，直至最终得到解答。

其中，抽象出数学模型，其实质是分析问题，从中提取出操作对象，并找出这些操作对象之间的关系，而数据结构正是实际问题中的操作对象以及这些对象间关系的数学抽象。因此，要解决非数值计算问题，必须研究如何合理组织数据，并采用适当的存储结构来存储，才能设计出合理的算法，提升程序的运行和存储效率。这正是数据结构的作用所在。

数据类型

数据类型是一组性质相同的值的集合及定义在此集合上的一些操作的总称。高级程序

设计语言的每一个变量、常量、表达式都有一个所属的确定的数据类型。类型明显或隐含地规定了在程序执行期间变量或表达式所有可能取值的范围，以及在这些值上允许进行的操作。例如，某语言整数类型取值范围 [–maxint,maxint]（maxint 是依赖特定的计算机的最大整数），定义在其上的一组操作为：加、减、乘、除、整除和取模等。

按“值”的不同特性，高级程序语言中的数据类型可分为两类：一类是非结构的原子类型，如整型、字符型等，原子类型的值是不可分解的；还有一种是结构类型，是由若干成分按某种结构组成的，因此是可以分解的，并且每个成分可以是非结构的，也可以是结构的。例如，定义一个记录类型，记录包含姓名、性别等。

抽象数据类型

抽象数据类型是指一个数学模型以及定义在此数学模型上的一组操作。抽象数据类型的定义与其在计算机内部如何表示和实现无关，即只考虑在数据元素集合上能完成什么操作，而不考虑如何完成。例如，前面所说的整数类型，无论加减乘除操作是如何运算的，对于用户来说，这些操作的性质不会变。这就是抽象的意义。

抽象数据类型中对数据对象和数据运算进行了声明，对数据对象的表示和数据运算的实现进行了分离。

抽象数据类型的两个重要特征：

- 抽象

用抽象数据类型描述程序处理的实体时，强调的是其本质的特征、其所能完成的功能，以及它和外部用户的接口。

- 封装

将实体的外部特征和内部实现细节分离，并且对外部用户隐藏其内部实现细节。

抽象数据类型的表示

本书按如下格式表示抽象数据类型：

```
ADT <抽象数据类型名>:
    数据对象:<数据对象的定义>
    数据关系:<数据关系的定义>
    基本操作:<基本操作的定义>
```

其中，数据对象和数据关系用集合或自然语言来描述；基本操作用 Python 语句描述，具体格式如下所示：

```
def 基本操作名(参数表)  #功能说明
```

例如，有理数抽象数据类型表示如下：

```
ADT Rational：
数据对象：D={ e1,e2 | e1,e2 均为整数 }
数据关系：R={ <e1,e2> | e1 是分子 ,e2 是分母 }
基本操作：
def  __init__(self,num,den ）  # 构造有理数 num/den
def __add__(self, v1, v2)   # 求出有理数 v1+v2 的和
def __sub__ (self, v1, v2)   # 求出有理数 v1−v2 的差
def __mul__(self, v1, v2)   # 求出有理数 v1*v2 的积
def __div__(self, v1, v2)  # 求出有理数 v1/v2 的商
def num(self, v1)          # 求得有理数 v1 的分子
def den(self, v1 ）        # 求得有理数 v1 的分母
```

抽象数据类型把实际生活中的问题分解为多个规模小且容易处理的问题，然后建立一个计算机能处理的数据模型，并把每个功能模块的实现细节作为一个独立的单元，从而使具体实现过程隐蔽起来。因此，抽象数据类型体现了程序设计中问题分离、抽象和信息隐蔽的特性。

项目二

探索商品基本信息表的实现
——线性表的应用

通常，超市销售的商品品种繁多，每种商品都含有品名、零售价、库存数等多种信息，如表 2-4 所示。为了更好地管理，超市必须保存这些商品信息，还要做到及时更新，如添加、删除和修改商品信息。通常超市都采用数据库来实现管理，而实际编程实现这些数据管理功能时，必须使用有效的数据结构技术来支持各种数据操作的执行。

表 2-4　商品信息表（简化）

序号	商品条形码编码	品名	库存数（件）	零售价（元）
1	6956416200029	1.25L 果粒橙	80	8
2	6956416200265	450mL 果粒芒果	80	3.5
3	6921311196364	1L 冰红茶	80	4
4	6921311196494	1L 绿茶	80	4
5	6924862101467	2L 百事可乐	100	6.5
6	6900451893036	2L 芬达	100	6.5
7	6900451891032	2L 可口可乐	100	6.5

项目学习目标

在本项目中，我们将通过小超市某类商品信息表的插入、删除操作的实现，体验线性表的应用。

完成本项目学习，须回答以下问题：

1. “商品信息表”采用什么数据结构？
2. 线性表可采用哪些存储结构？分别有什么优缺点？
3. 如何编程实现线性表数据结构的插入和删除等操作？

项目学习指引

1. 问题分析

为了实现使用计算机对超市商品进行管理，需要按照商品的不同分类，建立“商品信息表”，然后通过条码扫描器读取商品条码，录入每一种商品的具体数据。

上页的商品信息表中，每一个数据元素有相同的数据项。若将表中的每一个数据元素表示为 商品i（i 为原表中的商品序号），可以得到按原序号顺序排列的序列，如图 2-8 所示。

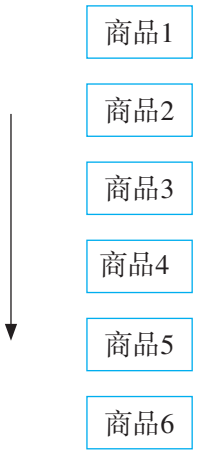


图 2-8　顺序排列的数据元素

从表中可以看到，各个数据元素（商品）之间存在着一对一的关系，自上而下顺序排列。此表也是线性表。

小贴士

为了便于理解，此处“商品信息表”设定为只包含序号、条形码、品名、库存数、零售价等信息。

核心概念

线性表（linear list）是 n（n ≥ 0）个数据特性相同的元素构成的有限序列。

小贴士

线性表的特点是存在唯一的一个被称作“第一个”的数据元素；存在唯一的一个被称作“最后一个”的数据元素；除第一个外，结构中每个数据元素只有一个前驱；除最后一个外，结构中每个数据元素只有一个后继。

参见 P39 知识链接“线性表”

思考与讨论

1. 在日常生活中，线性表的例子比比皆是。例如，26 个英文字母的字母表：A，B，C，D，...，Z 就是一个线性表。想一想，生活中还有哪些这样的例子？

2. 上一个项目中给出的抽象数据类型是否适用于商品信息表？为什么？

活　动

3.1 请尝试写出线性表抽象数据类型的定义。

3.2 请分析以下生活中的案例，判断哪个是线性表。

（1）公司的组织结构表：总经理管理数名总监，每个总监管理数名经理，每个经理都有各自的下属和员工。

（2）学生信息表，如表 2-5 所示。

表 2-5　学生信息表

学号	姓名	性别	出生年月	家庭地址
1	张一帆	男	2001.3	东街西巷 1 号 203 室
2	李小红	女	2001.8	南大北路 4 弄 5 号 6 室
……	……	……	……	……

3.3 下面哪些事物的相关信息适合用线性表存储和管理，为什么？

（1）在银行排队等候服务的顾客；

（2）书架上的一排书籍；

（3）计算机桌面上的各种图标及其相关信息；

（4）计算机的文件和文件夹；

（5）个人的电话簿；

（6）一辆汽车的所有部件和零件。

2. 设计算法

“商品信息表”——线性表不能被计算机直接处理，需要为它选择合适的存储结构，然后设计算法编程实现相关功能，如插入一条商品信息或者删除一条商品信息等。

线性表既可以采用顺序存储结构存储，也可以采用链式存储结构存储。

参见 P39 知识链接“线性表的存储”

（1）采用顺序存储结构

线性表采用顺序结构存储时，称之为顺序表。很多高级程序设计语言的数组（array）在计算机内的表示也是顺序结构，为此，可以用数组来表示顺序表。假设用数组 s 来存放商品信息表的数据元素，则每个数据元素都可以采用 s[0]、s[1]、s[2]、s[3]……表示，其中 s 为数组名，0，1，2，3……为数组下标，用来标识数据元素的位置和顺序。如图 2-9 所示（为方便说明，假

图 2-9　数组

设目前只有 4 条商品数据）。

若要在商品 1 后插入（insert）商品 5，则插入位置后的每个数据元素都要向后移动空出插入位置然后插入商品 5，具体过程如图 2-10 所示。

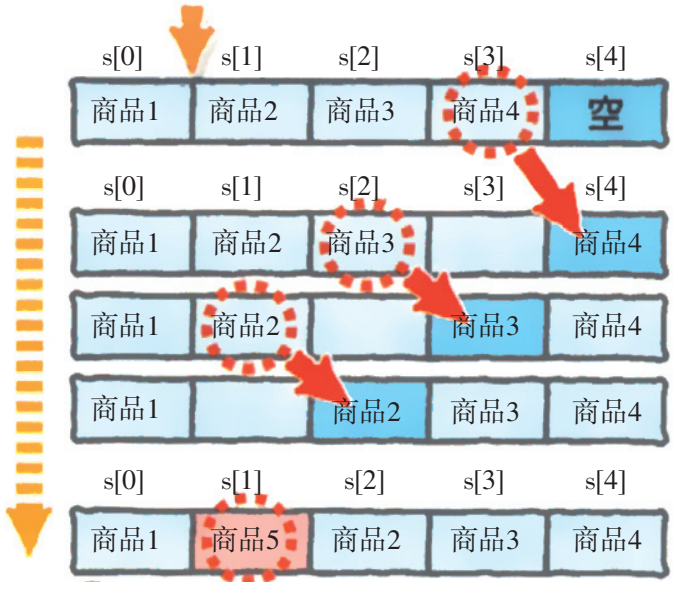


图 2-10　数组的插入操作

而删除（delete）操作恰好是插入操作的逆过程。若需将插入的商品 5 删除，则须将该数据元素后的所有数据元素依次向前移动，使后一数据元素覆盖前一数据元素。最终，依旧保证了数据元素存储的连续性，如图 2-11 所示。

小贴士

一个元素覆盖前一个元素后，该元素本身依旧存在于原先的位置上，要等到后一个元素将其覆盖。最后一个元素覆盖前一个元素后，最后一个位置须去掉。

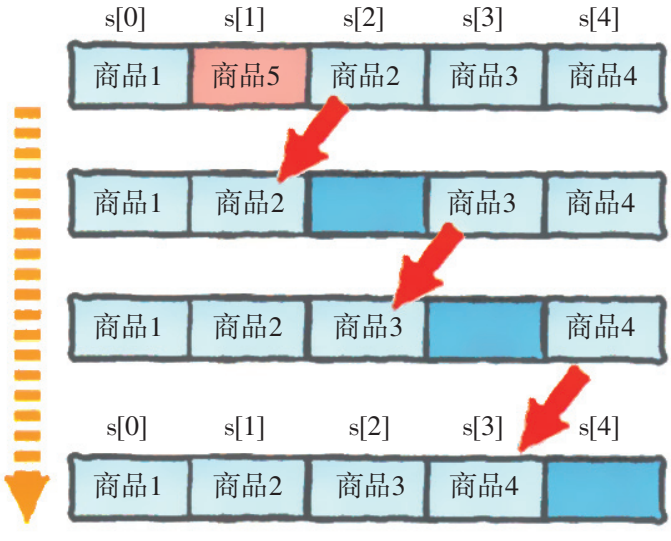


图 2-11　数组的删除操作

思考与讨论

1. 使用数组存储插入和删除移动数据元素的次数与什么相关？

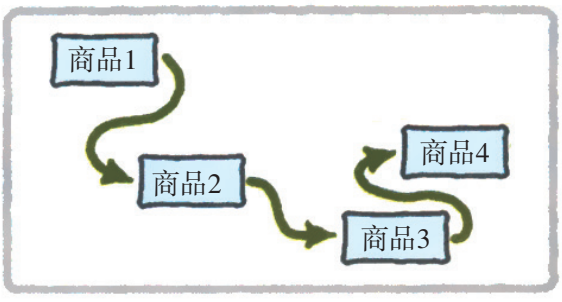
2. 使用数组存储，确定插入和删除元素的位置是否方便？

（2）采用链式存储结构

线性表采用链式存储结构存储时，称为链表。链表中的数据可以是不连续存储的。链表不要求逻辑上相邻的数据元素在物理位置上也相邻，数据元素之间通过指针链接，如图 2-12 所示。

小贴士

当一个序列中只含有指向它的后继结点的链接时，就称该链表为单链表。



用有方向性的链连接各结点，表示数据的顺序；即使数据存储位置改变，顺序还是不变的

图 2-12　链表

若要在商品 1 后插入商品 5，先要切断插入位置前后的数据链，然后将断开的链连接到插入元素，具体过程如图 2-13 所示。

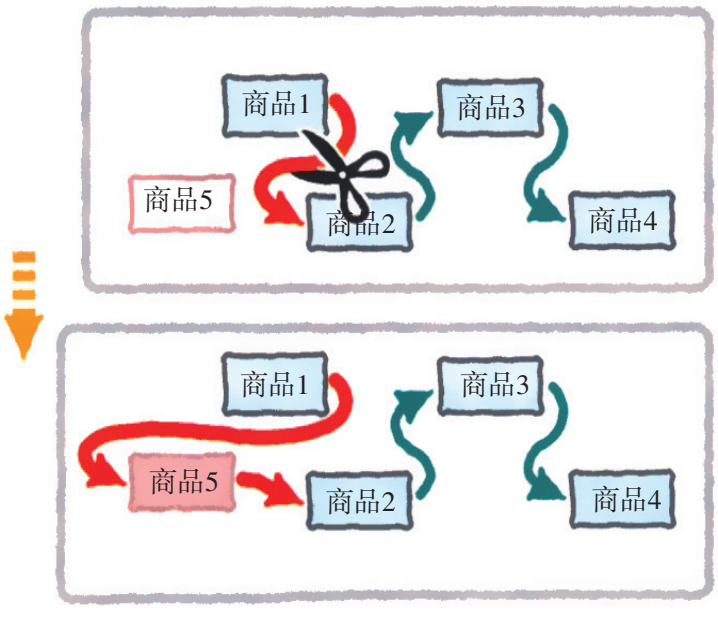


图 2-13　链表的插入操作

若要删除一个商品，只要将须删除数据元素的前一个数据元素的指针指向下一个数据元素即可，如图 2-14 所示。

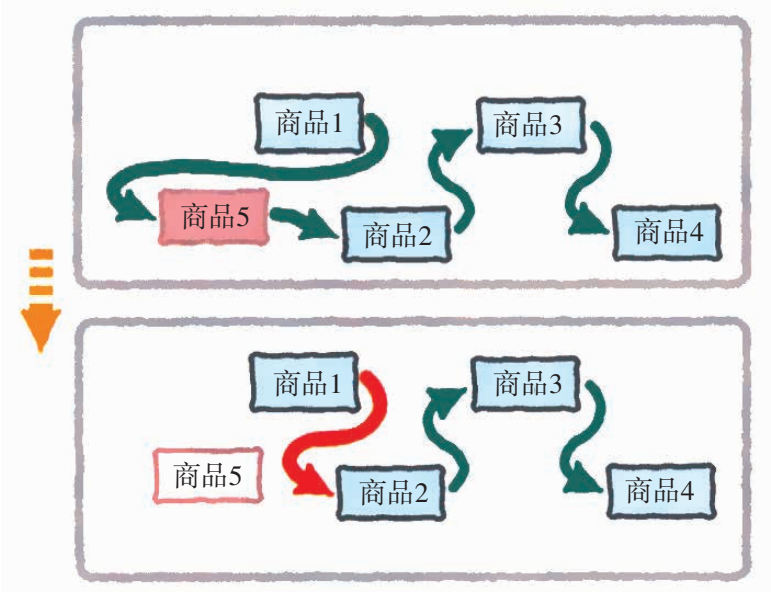


图 2-14　链表的删除操作

思考与讨论

1. 数组和链表有何区别？

2. 你觉得本例中，链表插入操作方便还是数组方便？为什么？如果编程实现，哪个执行操作的次数可能多些？

活　动

3.4　现需要删除商品 3 数据元素，请在图 2-15 的基础上，用图示方式呈现全过程（数组和单链表两种方式）。

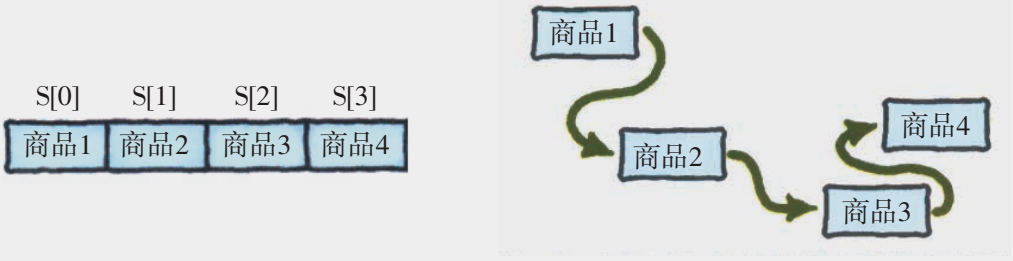


图 2-15　数组和单链表

3.5 完成在商品 3 后插入商品 5 的算法步骤设计。

算法步骤（数组）
① 判断插入位置 i 是否合法，若不合法则返回 ERROR。 ② 判断数组存储空间是否已满，若满则返回 ERROR。 ③ _____ ④ _____ ⑤ 数组长度加 1。

算法步骤（链表）
① 查找商品序号为 i 的结点并由指针 p 指向该结点。 ② 生成一个新结点。 ③ 将新结点的数据域信息 _____。 ④ 将新结点的指针域信息 _____。 ⑤ 将 p 结点的指针域指向新结点。

3. 程序实现

根据上述算法，可以编写程序实现“商品信息表”的数据插入与删除。

用 Python 语言实现顺序存储结构类型可以采用定义类的方式。商品信息定义如下：

```
class goods:
    def _init_(self,bar_code='',name='',s_number=0,price=0):
        self. bar_code= bar_code # 条形码
        self.name=name # 品名
        self.s_number=s_number # 库存数量
        self.price=price # 零售价
```

思考与讨论

如果要加入生产日期，该如何定义？

小贴士

数组在不同高级语言定义是不同的，有的可以使用记录类型即结构类型的数组存储数据元素（将多种数据类型集合在一个用户自定义的数据类型中）。在 Python 中可以用列表嵌套、定义类类型的对象列表等方式。为了方便理解算法思想，这里采用定义类类型的对象列表模拟数组的方式。

活 动

3.6 若采用数组，请参考配套资源中的“商品表（数组）.py”，完成以下程序，实现“商品信息表”的数据插入和删除，并上机运行。

数据插入程序

```
# 表任意位置插入操作：
    def insert_sq(self,i,elem):
        if not isinstance(i,int):
            raise TypeError
        if i < 0 or i > self.num:    #num 是商品个数
            raise IndexError
        self.data.append(goods)
        for j in range(self.num,i-1):
            ____________________
        ____________________________
        ____________________________
```

数据删除程序

```
# 删除表任意位置上的元素操作：
def delete_sq(self,i):
    if not isinstance(i,int):
        raise TypeError
    if i < 0 or i >=self.num:
        raise IndexError
    if self.num>0:
        for j in range(i,self.num-1):
            ____________________________
        del self.data[self.num-1]
    ____________________________
```

*3.7 若采用链表，请参考配套资源中的“商品表（链表）.py”，完成以下实现“商品信息表”的数据插入和数据删除的程序。

数据插入程序

```
# 定义将元素 item 插入表中作为第 index 个元素
    def insert_lk(self,index,item):
        if index<0 or index >self.getlength():
            print ('index error.')
            return
        if index ==0:
            q = Node(item,self.head)
            self.head = q
        else:
            post  = self.head    #post 为指针前驱
            p = self.head.next
            j = 1
            while p!=0 and j<index:
                post = p
                ____________
                ______
            if index ==j:
                q = Node(item,p)
                ____________
                ____________
```

数据删除程序

```
# 定义删除表中第 index 个元素的方法
    def delete_lk(self,index):
        if self.is_empty() or index<0 or index
>self.getlength():
            print ('Linklist is empty.')
            return
        if index ==0:
            self.head = self.head.next
        else:
            post = self.head
            p = self.head.next
            j = 1
            while p!=0 and j<index:
                post = p
                ____________
                ______
            if index ==j:
                ______
```

知识链接

线性表

线性表是 n（n ⩾ 0）个数据特性相同的元素构成的有限序列。记为（a_1, a_2, …, a_{i-1}, a_i, a_{i+1}, …, a_n），其中 a_{i-1} 是 a_i 的直接前驱元素，a_{i+1} 是 a_i 直接后继元素，a_1 无前驱元素，a_n 无后继元素。n 称为线性表的长度，当 n = 0 时，称为空表。

例如，商品信息表 =（商品 1，商品 2，商品 3，商品 4，商品 5，商品 6……）

线性表的抽象数据类型表示如下：

```
ADT List:
    数据对象: D={a_i |a_i ∈ ElemSet, i=1,2,…,n,n>=0}
    数据关系: R={<a_i-1, a_i >| a_i-1, a_i ∈ D, i = 2,…,n}
    基本操作:
      def InitList(self)   # 建立一个空的线性表
      def GetElem(self,i)   # 返回线性表的第 i 个元素
      def Length(self)   # 求线性表的长度
      def LocateElem(self,x)   # 求元素 x 在线性表中的位置；若不存在 x，则返回 0
      def Insert(self,i,x)   # 在线性表的第 i 个位置上插入一个新元素 x
      def Delete(self, i)   # 删除线性表的第 i 个元素
```

线性表的存储

线性表可以使用顺序存储结构存储（此时称为顺序表），也可以使用链式存储结构存储（此时称为链表）。

1. 线性表的顺序存储

大多数高级程序语言的数组在计算机内的表示是顺序结构。为此可以用数组来表示顺序表。

数组（array）是由数据类型相同的数据元素构成的有序集合。它被用来存放大量数据类型相同的数据。

数组与变量一样，每个数组都要有一个唯一的名称，即数组名，命名规则和变量相同。数组元素是组成数组的基本单元，每一个存储单元对应于一个数组元素。数组元素用数组的名字和它自己在数组中的顺序位置来表示。例如，d[0] 表示名字为 d 的数组中的第一个元素，d[1] 代表数组 d 中的第二个元素，以此类推，如图 2–16 所示。

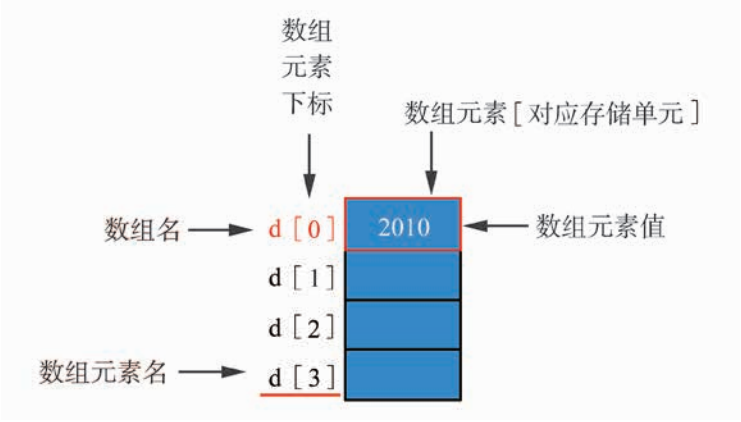


图 2-16　数组 d

（1）插入操作

如果要实现在线性表 L 中的第 i 个位置插入新元素 e，插入算法的思想为：

如果插入位置不合理，抛出异常；
如果线性表长度大于等于数组长度，则抛出异常或动态增加数组容量；
否则从最后一个元素开始向前遍历到第 i 个位置，分别将它们都向后移动一个位置；
将要插入元素 e 填入位置 i 处；
线性表长度加 1。

（2）删除操作

删除算法的思想为：

如果删除位置不合理，抛出异常；
否则取出删除元素；
从删除元素位置开始遍历到最后一个元素位置，分别将它们都向前移动一个位置；
表长减 1。

数组的优点是具有随机访问的特性，可以快速地存取任意位置的元素。缺点是为了保持存储的连续性，插入和删除操作可能会移动大量元素。因此，数组适合查找和存取数据的操作，而不适合插入和删除的操作。

2. 线性表的链式存储

链式存储是将数据元素存储在地址任意的存储空间中，用指针来指出元素间的逻辑关系，所以每个结点空间由数据域和指针域两部分组成，数据域用来存放数据元素，指针域用来存放指向后继的指针。链式存储是一种动态存储结构，即每当要存储元素时就去申请一个存储空间，存放数据元素及指针。

链表中的结点形式用以下方式表示：

数据	指针

（1）单链表的插入

假设存储元素 e 的结点为 s，若要实现将 e 插入到 a_i 和 a_{i+1} 之间，如图 2-17 所示，该如何做呢？

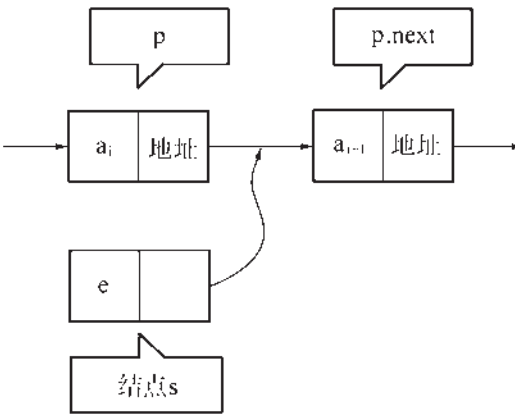


图 2-17　将要插入结点 s

只须让 s.next 和 p.next 的指针做一点改变即可实现插入。即：

```
s.next = p.next;
p.next = s;
```

通过图 2-18 可以理解以上两句代码。

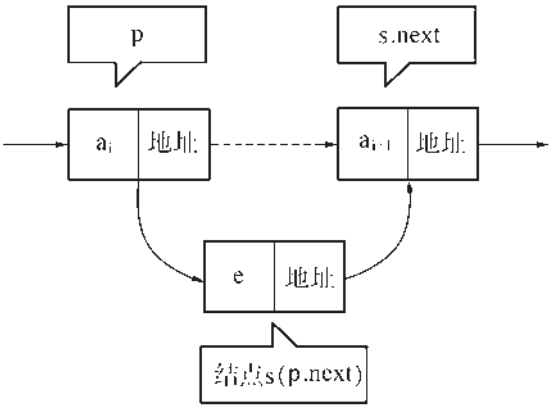


图 2-18　插入结点 s

那么，这两句代码的顺序是否可以交换位置？

即先 p.next = s，再 s.next = p.next。

如果先执行 p.next = s 的话，p.next 会被 s 覆盖，s.next = p.next 就等于 s.next = s 了，所以这两句代码是不能颠倒次序的。

在单链表第 i 个位置上插入结点 e 的算法思想为：

若 i = 0 则结点 e 插在表头（假设链表从第 0 个开始计数）；
否则指针 p 从表头开始沿着链往后查找第 i 个结点（为便于插入，在查找时始终要记住 p 的前驱，假设用 post 指向）；
当 p = 0 时，说明 i 的值超出链表长度，即不存在第 i 个结点，不插入，返回。
否则当 p 指在第 i 个结点上时，在 post 与 p 之间插入新结点 s。

（2）单链表的删除

单链表的删除操作如图 2-19 所示。

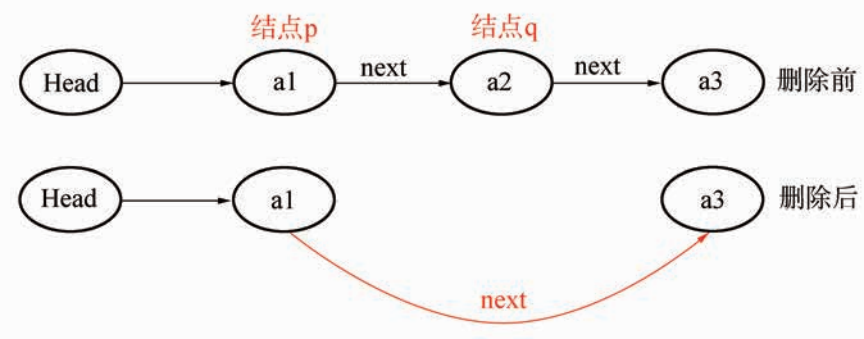


图 2-19　单链表的删除

假设元素 a2 的结点为 q，要实现删除结点 q 的操作，其实就是将它的前驱结点的指针指向后继结点即可。

可以是：p.next = p.next.next; 也可以是：q=p.next; p.next=q.next;

删除单链表第 i 个结点的算法思想为：

若 i=0，则删除首结点，即 head= head.next；
否则指针 q 从链表头开始沿着链往后查找第 i 个结点（为便于删除，在查找时始终要记住 q 的前驱，假设用 p 指向）；
当 q=0 时，说明 i 的值超出链表长度，即不存在第 i 个结点，不用删除，返回。
否则当 q 指在第 i 个结点上时，删除 q，即 p.next=q.next。

无论是单链表插入还是删除算法，它们其实都是由两部分组成：第一部分是从第一个元素开始逐个查找第 i 个元素，第二部分是实现插入和删除元素。

链表的优点是插入和删除的速度快，链表长度不固定，拓展灵活。缺点是不能随机查找，必须从第一个开始逐个查找，效率很低。因此，链表比较适合插入或删除数据频繁的操作，而不适合查找操作。

总之，线性表的顺序存储结构和链式存储结构各有其优缺点，不能简单地认为哪个好，哪个不好，需要根据实际情况，来综合判断采用哪种存储结构更能满足需求。

单元挑战　实现学校学生健康情况登记表的操作

一、项目任务

现有某学校的学生健康情况登记表如表 2-6 所示，表中每个学生的情况为一个记录，它由姓名、性别、年龄、班级和健康状况等 5 个数据项组成。

表 2-6　学生健康情况登记表

姓名	性别	年龄	班级	健康状况
李林	男	17	高二 6 班	良好
汤晨	女	16	高二 6 班	良好
王平平	男	17	高二 6 班	一般
陈小莉	女	17	高二 6 班	良好
……	……	……	……	……

若用数组存储表中数据，假设班级人员变化，新进来“常龙”同学（男生、17 岁，高二 6 班、健康状况良好），需要添加在“王平平”和“陈小莉”之间；又由于“李林”转学，需要将他的记录删除，请编程实现相应的操作。

二、项目指引

1. 给出程序设计的基本思想、原理和算法描述。

2. 画出流程图，并编写程序。

（1）编写删除操作函数。

（2）编写插入操作函数。

（3）实现以下功能：

① 调用插入操作函数，在上表中“王平平”和“陈小莉”之间插入“常龙”同学；

② 调用删除操作函数，删除表中“李林”同学；

③ 输出最终的表。

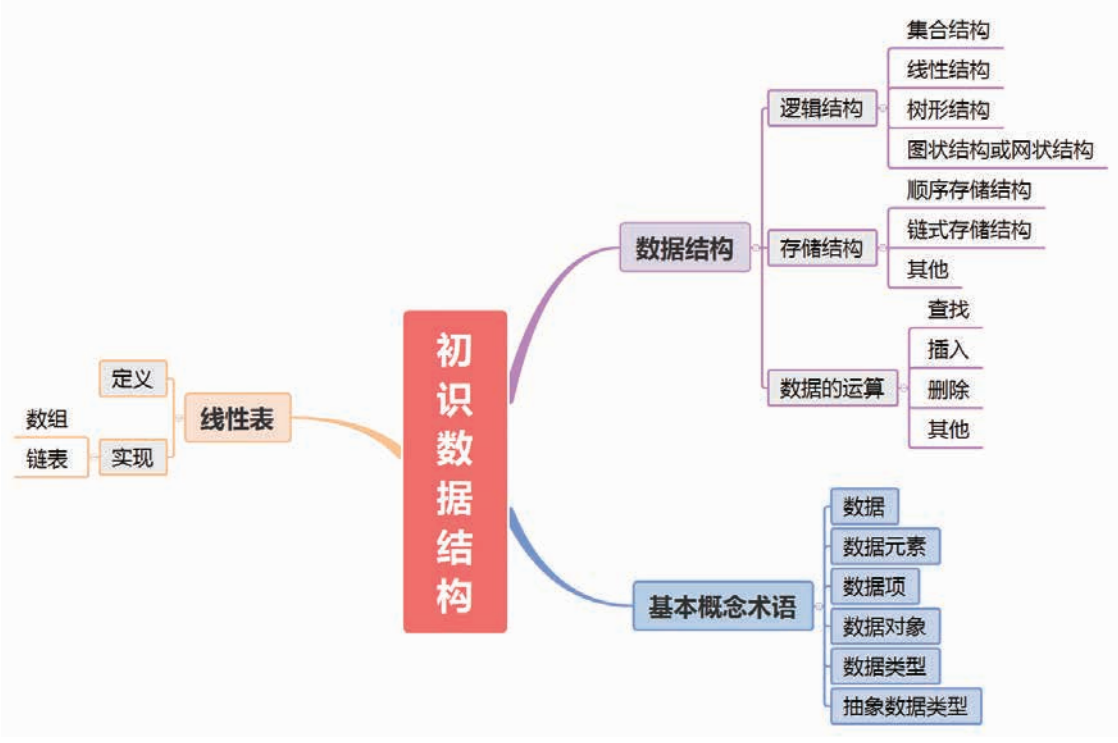
3. 给源程序添加注释，保存并打印出程序及运行结果。

三、交流评价与反思

以自己熟悉的信息表达工具（如演示文稿等）制作电子作品，通过网络或课堂展示交流自己的程序，并对他人的程序进行评价。

单元小结

一、主要内容梳理



二、单元练习

1. 设有一个正整数序列组成的有序表（按递增次序有序，且允许有相等的整数存在），请编写能实现下列功能的程序：

（1）确定在序列中比正整数 x 大的数有几个（相同的数只计算一次，如序列{20，20，17，16，15，15，11，10，8，7，7，5，4}中比 10 大的数有 5 个）；

（2）将比正整数 x 小的数按递减次序排列；

（3）将比正整数 x 大的偶数删除。

（4）比较使用数组和链表实现上述功能的优劣。

2. 把二次多项式 ax^2+bx+c 设计成一种抽象数据类型，该类型的数据对象为三个系数项 a,b 和 c，操作部分为：

（1）初始化 a，b 和 c 的值；

（2）做两个多项式加法；

（3）根据给定 x 的值计算多项式的值；

（4）计算方程 $ax^2+bx+c=0$ 的两个实数根；

（5）按照 ax**2+bx+c 的格式输出二次多项式。

3. 某航空公司有一个自动预订飞机票的系统，假设该系统中有一张用单链表表示的乘客表，见表 2-7。表中结点按乘客姓氏的字母次序进行链接（指针暂用序号表示），请为该系统编写有新乘客订票时修改乘客表的程序。

表 2-7　乘客表

data	Link
Liu	7
Chen	4
Wang	5
Bao	2
Mai	8
Dong	6
Xi	0
Deng	5
Chang	3

*4. 约瑟夫环（约瑟夫问题）是一个数学的应用问题：已知 n 个人（以编号 1，2，3，…，n 分别表示）围坐在一张圆桌周围。编号为 k 的人从 1 开始报数，数到 m 的那个人出列；他的下一个人又从 1 开始报数，数到 m 的那个人出列；依此规律重复下去，直到圆桌周围的人全部出列。请用单链表设计一个程序求出出列顺序（单链表最后一个结点的指针指向链表的首结点）。

程序：

三、单元评价

评 价 内 容	达 成 情 况
能从教学管理相关数据认识数据元素、数据项、数据对象等基本概念（T）	
能从学生信息表、班级学生管理组织结构认识数据结构的逻辑结构（T）	
理解数据结构的四种逻辑结构及两种最基本的存储结构（T）	
理解什么是数据结构，了解数据结构在解决问题过程中的重要作用（T，A）	
了解抽象数据类型及其重要性（T）	
知道商品信息表是一种线性表（A）	
会使用数组实现顺序存储结构存储商品信息表；会使用链表实现链式存储结构存储商品信息表（T，I）	
会使用顺序存储结构和链式存储结构实现商品数据的插入与删除（T，I）	
能说出数组和链表的区别（A）	
关注线性表的实际应用（A）	

说明：A—信息意识，T—计算思维，I—数字化学习与创新，R—信息社会责任

第三单元

特殊的线性表

线性表是最常用的数据结构，也是最简单最基本的一种数据结构。有一些特殊的线性表，它们与线性表的逻辑结构完全相同，但是在数据类型设定上，或者操作上有所不同，在实际应用中比较广泛。例如，日常生活中，经常会遇到排队预订系统等，这可以借助特殊的线性表——队列来实现其功能；文字处理软件一般都具有对文本字符进行编辑、查找、选择、剪切、粘贴、撤消操作等各种编辑处理功能，这些功能的实现很多也都会用到特殊的线性表——栈、字符串。在本单元中，我们将通过实例探究来学习这些特殊的线性表。

学习目标

◆理解队列的概念，并能编程实现顺序队列的进队、出队操作。

◆理解栈的概念，并能编程实现顺序栈的进栈、出栈操作。

◆理解字符串的概念及其基本操作。

单元挑战

按解密规则提取情报

项目四

探索电子排队预订功能的实现
——队列的应用

日常生活中，商品和服务需求大于供给的情况时有发生，例如限量商品发售、医院专家门诊号等。为了避免商场、医院或营业厅里人潮拥挤，等候的客户排起长龙，一些企事业单位在相应网站或手机 App 上提供了预约服务，如图 3-1 所示。客户输入身份、账号信息提交后自动进入电子排队预订系统。你知道电子排队预订系统是如何实现排队预约的吗？

图 3-1　电子排队预订系统

项目学习目标

在本项目中，我们一起使用一种特殊的线性表结构来模拟实现排队功能。

完成本项目学习，须回答以下问题：

1. 排队预订遵循的是什么规律？
2. 什么是队列？队列遵循的原则是什么？
3. 队列在计算机中如何实现？

项目学习指引

1. 分析问题

电子排队预订系统是计算机自动控制的排队系统，排队的特点是先来先服务，那计算机是怎么实现自动排队的呢？

当客户在电子排队预订系统提交信息后，客户的预订数据就进入队列，称为进队（也称入队），即在队尾（rear）插入一个客户账号数据；当正式购买商品时就到队列中取出客户账号数据，称为出队，即在队首（front）取出一个客户账号数据（客户账号数据暂用 A0001 形式替代），如图 3-2 所示。

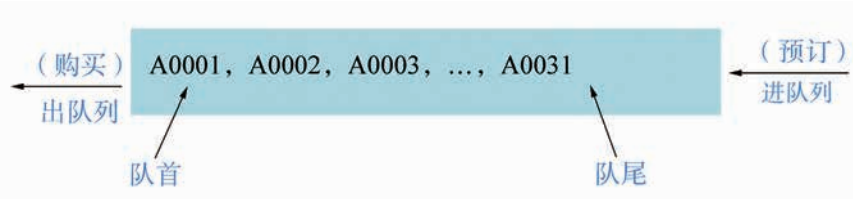


图 3-2　预订—购买队列

用先进先出的数据结构来存储排队的客户账号数据，就可以方便地实现先来先服务，这种数据结构称为队列。

思考与讨论

生活中还有其他类似的先进先出的例子吗？

核心概念

队列（queue）是一种只允许在表的前端（队头）进行删除操作，而在表的后端（队尾）进行插入操作的线性表。队列又称为先进先出（first in first out，FIFO）表。

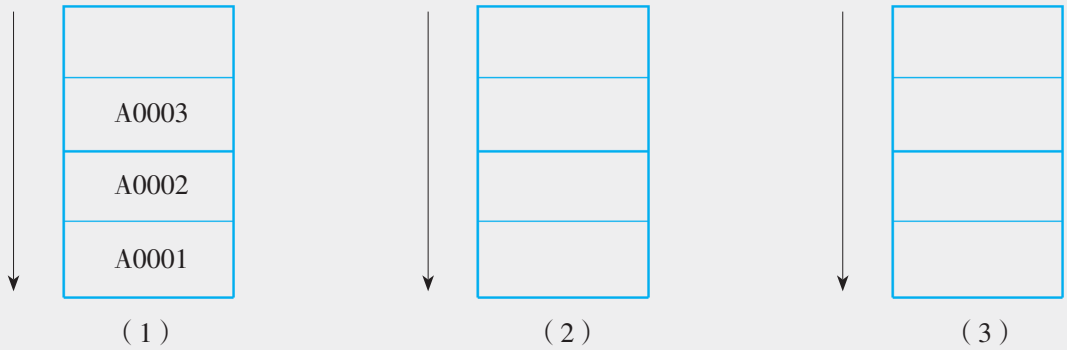
小贴士

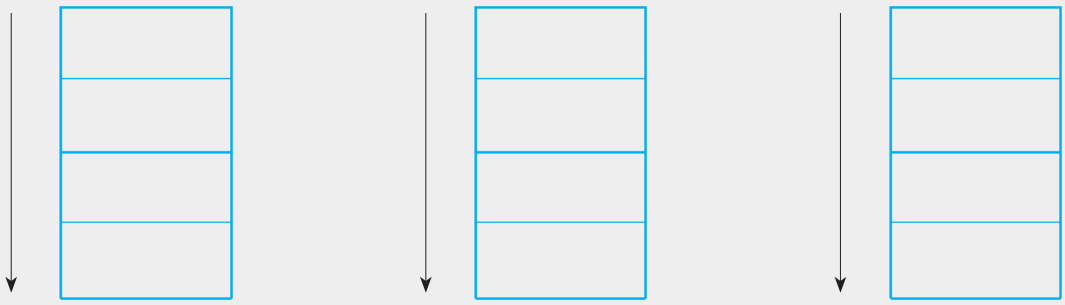
队列是一种操作上受限制的线性表，只允许在队首和队尾进行操作。

参见 P53 知识链接“队列”

活　动

4.1　假设排队预订客户账号数据为 A0001、A0002、A0003、A0004，下面空的队列中依次 A0001 进队，A0002 进队，A0003 进队，叫号出队，A0004 进队，叫号出队，叫号出队，叫号出队，画出队列的变化过程。





4.2 请尝试写出队列的抽象数据类型定义。

小贴士

进队在队列尾端插入一个新元素。

出队从队列首端取出(或称删除)一个元素。

front 为头指针，rear 为尾指针。

2. 设计算法

排队预订的队列变化过程如图 3-3 所示(为图示方便，暂定图中队列空间只有 4 个)。客户预订即为进队，假设 A0001 表示第一个客户账号数据，购买即为队首出队。图中 rear 指向队列的尾端，图中 front 指向队列的首端。

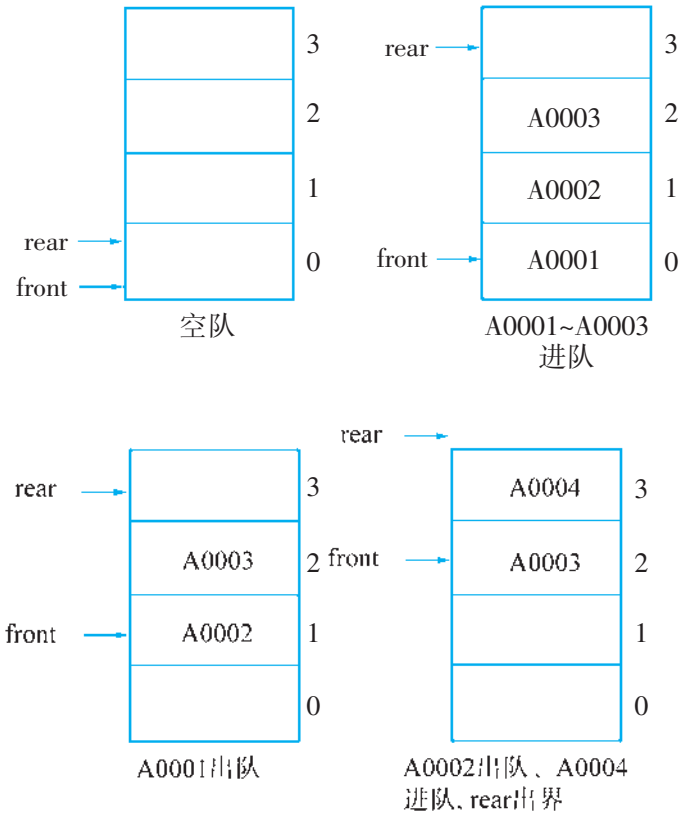


图 3-3 进队、出队的过程示意

参见 P54 知识链接“队列的常用基本操作”

思考与讨论

1. 队尾指针是否一定要指向队尾下一个元素？
2. 队列在反复进队、出队后会出现尾指针 rear 和头指针 front 出界的情况，有什么解决方法？

队列在计算机里怎样表示呢？队列是操作上有限制的线性表，既可以用数组表示一个队列，也可以用链表表示一个队列。一般若问题规模已知，即总的进队元素个数已知的话，队列可以用顺序存储结构存储，即用数组表示队列；若进队元素个数无法预计，则队列可以用链式存储结构存储，即用链表表示队列。

思考与讨论

用数组方式和用链表方式存储队列，队列指针的形式有何不同？

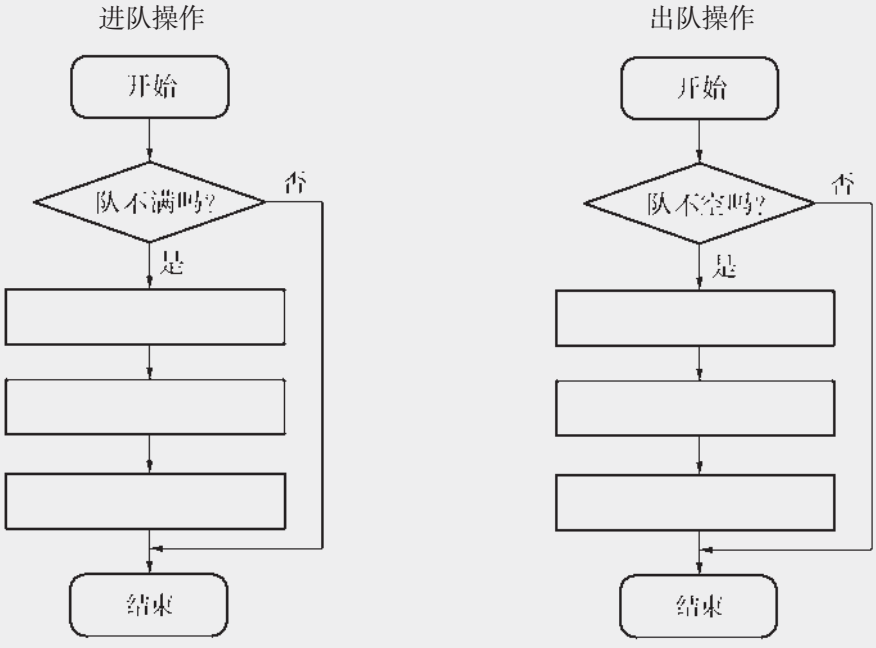
小贴士

用数组存储队列时，可能会遇到指针溢出的问题，最简单的解决方法是将 rear 增 1 改成 rear = (rear+1)%M；front 的增 1 改成 front = (front+1)%M。其中 M 为队列的空间数，% 是取余运算符，这时的顺序队列被称为循环队列。

为了操作方便增加一个计数器 number，记录队列中的元素个数。

活　动

4.3 在算法流程框图中完成进队和出队操作（数组名和变量名可以自取）。



3. 程序实现

根据上述算法，可以利用学过的编程知识来编程实现排队预订。首先要定义队列的类型并进行初始化（即置空）操作，指针变量要设定初始值。用列表表示队列的类型定义如下：

```
class SqQueue:
    def __init__(self, size):    # 队列初始化
        self.size =size    # 定义队列长度
        self.queue = [' ']*size    # 存储队列元素的列表
        self.front=0    # 头指针
        self.rear=0    # 尾指针
        self.number=0    # 计数器
```

数字化学习

观看配套资源中的循环队列演示动画。

活　动

4.4 打开配套资源中“循环顺序队列 .py”程序，补充完整以下代码，并进行运行测试，模拟实现排队预订功能。

```
def EnQueue(self,e):  # 进队程序
    if(self.number==self.size):
        print（" 队满，不能进 "）
    else:
        self.queue[self.rear]=e
        ____________________
        self.number=self.number+1

def OutQueue(self):  # 出队程序
    if self.number==0:
        print（" 队空 "）
        return -1
    else:
        e = self.queue[self.front]
        ____________________
        self.number=self.number-1
        return e
```

知识链接

队列

排队的队伍在计算机中称为队列，队列是一种只能在表的首端取出数据，在表的尾端添加数据的线性表，即队列是操作上有限制的线性表，队列是一种先进先出数据结构。

队列的抽象数据类型表示如下：

```
ADT Queue：
数据对象：D={a_i |a_i ∈ ElemSet, i=1,2,…,n,n>=0}
数据关系：R={<a_i-1, a_i>| a_i-1, a_i ∈ D, i=2,…,n}
基本操作：
    def __init__(self)  # 初始化一个空队列
    def QEmpty(self)  # 若队列空，则返回 True，否则返回 False
    def QLength(self)  # 返回队列的元素个数
    def GetHead(self)  # 返回队列的队头元素
    def EnQueue(self,e)  #e 元素进队
    def OutQueue(self)  # 元素出队
```

队列有两种存储方式，一种是用数组存储，这种方式存储的队列称为顺序队列；另一种用链表存储，这种方式存储的队列称为链队列，如图 3-4 所示。元素插入队尾称为进队，队首元素取出称为出队。队首用 front 指针指示，队尾用 rear 指针指示。

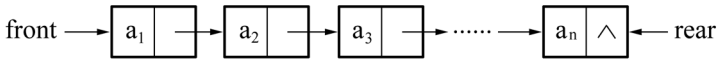


图 3-4　链队列

用数组存储队列元素时，会遇到指针溢出问题。即当队满时，再有元素进队，就会产生指针的溢出（上溢出）；当队空时，再要取出元素，也会产生指针的溢出（下溢出）。

另外，还有“假溢出”问题。随着元素的进队和出队，整个队列是向数组下标较大的位置移动。当移动到数组中下标最大的位置后，队列的空间就用尽了。此时，即使数组下标较小的位置处还有空闲空间，元素也无法进队了，这种现象就叫“假溢出”。

解决以上问题可以有两种方法：一是每次队首元素出队，后面的元素都向前移动一次。但这种做法会使出队效率较低。另一种方法是采用循环队列，即把队列的首尾相连，当队尾指针超出数组长度时，就将其设回到最初的队首位置（数组下标为 0）。如图 3-5 所示，图中为 8 个空间的队列形成的循环。为了能实现循环，可以采用指针加 1 除队列空间数后

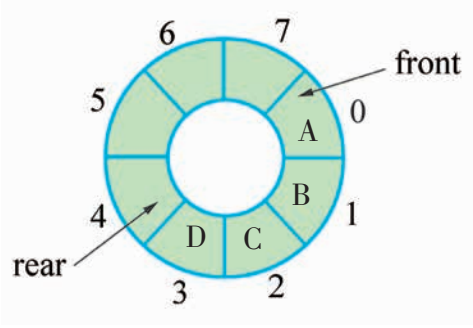


图 3-5　循环队列

取余，替代指针加 1 的方法，即 rear= (rear+1)%size（队尾指针循环），front=(front+1)%size（队首指针循环），其中，% 就是取余运算，size 为队列的空间数。

队列的常用基本操作

1. 进队

进队就是从队尾添加数据的操作。

顺序队列进队的算法思想：若队列不满，则将进队的元素送入尾指针 rear 所指空间，元素个数计数器增 1，然后将尾指针 rear 往尾部方向移动一位即指向新的队尾。

```
def EnQueue(self,e):
    if(self.number==self.size):
        print (" 队满，不能进 ")
    else:
        self.queue[self.rear]=e
        self.rear=(self.rear+1)%self.size
        self.number=self.number+1
    return
```

2. 出队

出队就是在队首取出数据的操作。

顺序队列出队的算法思想：若队列不空，则将队首指针 front 所指空间的内容取出赋予变量，元素个数计数器减 1，然后将首指针 front 往后移动一位即指向新的首端。

```
def OutQueue(self):
    if(self.number==0):
        print (" 队空 ")
    else:
        e=self.queue[self.front]
        self.front=(self.front+1)%self.size
        self.number=self.number-1
        return e
```

项目五

模拟实现软件的撤消功能
——栈的应用

队列是一种操作受限制的线性表，只允许先进先出。还有一种操作受限制的线性表，即栈，它的特点和队列恰好相反，是后进先出。日常学习和工作中，人们使用的许多应用软件，如办公软件，大多提供了“撤消”功能。该功能允许用户撤消有限的操作步骤，方便恢复到误操作之前的状态（图 3-6）。而这一功能就可以借助栈来实现。

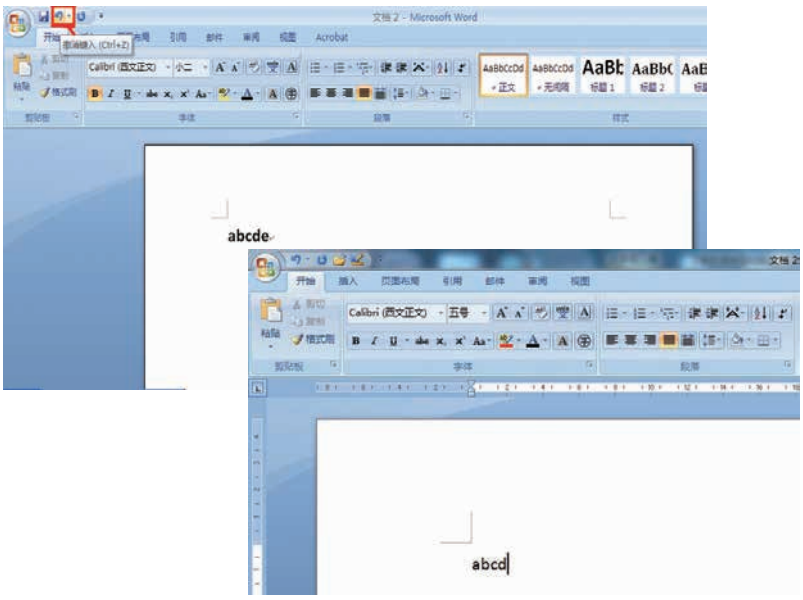


图 3-6　撤消键入示意图

项目学习目标

在本项目中，我们将一起来探索如何使用栈这种数据结构来编程模拟实现“撤消”功能。

完成本项目学习，须回答以下问题：

1. 软件撤消操作的过程是怎样的？
2. 什么是栈？为何可以使用栈来实现撤消操作？
3. 栈有哪些基本操作？

项目学习指引

1. 分析问题

核心概念

栈（stack）：是一种仅允许在表的一端进行插入或删除操作的线性表。这一端被称为栈顶（top），栈也称为后进先出（last in first out，LIFO）线性表。

小贴士

栈属于线性表的一种，是操作上有限制的线性表，只允许在栈顶进行操作。

参见 P59 知识链接“栈”

要实现撤消功能，首先要思考操作的顺序问题。例如，输入 abcde 的操作顺序是先输入 a，然后是 b，再是 c……而撤消的时候是按逆序，先撤消 e，再撤消 d……可以看出，后输入的先撤消，先输入的后撤消，如图 3-7 所示。

如果用某种后进先出的数据结构来存储 a，b，c，d，e 的话，可以方便地实现撤消。这种数据结构称为栈。

图 3-7　撤消菜单

思考与讨论

生活中是否还有其他类似的后进先出的例子？

活　动

5.1　请尝试写出栈的抽象数据类型定义。

5.2　网上搜索列车调度方法，根据图 3-8 画出用栈进行调度的过程示意图，并说明调度的原理。

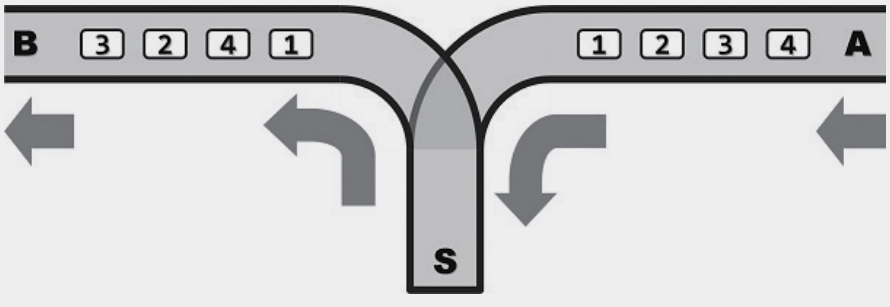


图 3-8　列车调度示意

2. 设计算法

在文档中依次输入 a，b，c，d，e 后，这些字母会依次进栈，每点击一下“撤消”按钮，栈内最后一个字母就会出栈。

栈可以用数组来存储元素，也可以用链表存储元素。假设用数组存储元素，即用数组存放输入的字母，用一个变量作为栈顶指针，如图 3-9 中的 top。栈顶指针始终指向栈内的最后的元素。

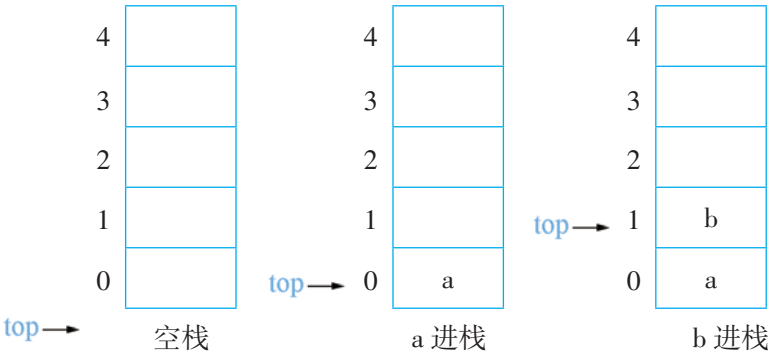


图 3-9　进栈示意

进栈和出栈的操作就可以用算法来描述了。

思考与讨论

栈与队列的操作有何不同？

小贴士

进栈（push）：也称入栈，是向栈插入新元素，即把新元素插入到栈顶元素的上面，使之成为新的栈顶元素。

出栈（pop）：也称退栈，是删除栈顶元素，使其相邻的后一个元素成为新的栈顶元素。

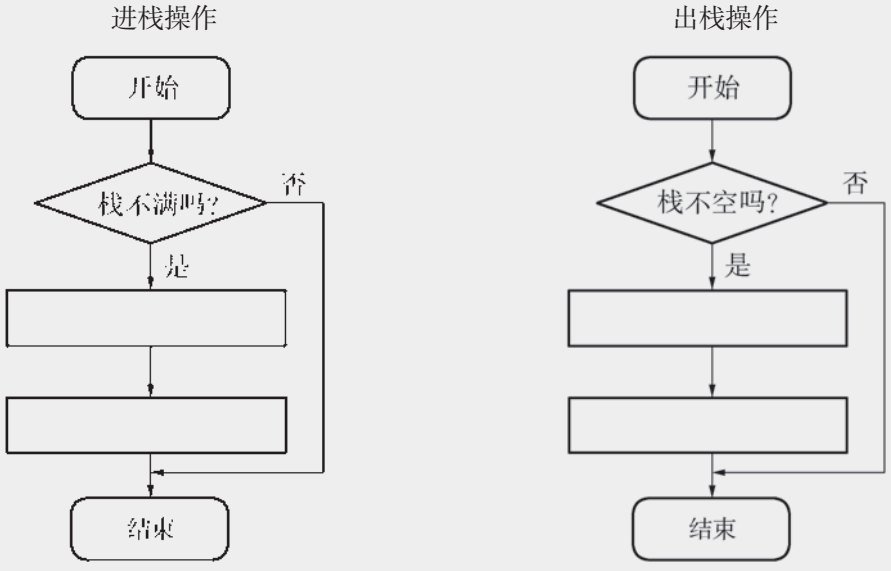
参见 P60 知识链接“栈的常用基本操作”

小贴士

用数组存储的栈称为顺序栈。为方便理解，本项目只存储字母。

活　动

5.3 在算法流程框图中完成进栈和出栈操作（数组名和变量名可以自取）。



3. 程序实现

根据上述的算法，可以编程实现撤消操作。首先要定义栈并进行初始化操作（即置空栈），指针变量要设定为初始值。

```
class Sqstack (object):
    def_init_ (self,size):
        self.size=size    # 定义栈的长度
        self.stack=[' ']*size    # 存储栈元素的列表
        self.top=-1    # 栈顶头指针
```

思考与讨论

你能编程实现恢复操作功能吗？

活　动

5.4 打开配套资源中的“顺序栈 .py”程序，补充完整以下代码，并运行测试，模拟实现撤消功能。

```
def Push(self,e):      # 进栈程序
    if self.top==self.size-1:
        print(" 栈满，不能进栈 ")
    else:
        _____
        _____

def Pop(self):     # 出栈程序
    if self.top==-1:
        print(" 栈空，没有元素出栈 ")
        return -1
    else:
        _____
        _____
        return e
```

知识链接

栈

要理解栈这个概念，首先要明白“栈”这个字的字面意思，如此才能把握本质。“栈”的意思是存储货物或供旅客住宿的地方，可引申为仓库、中转站，所以引入到计算机领域里，就是指数据暂时存储的地方，是一种只能在表的一端进行插入（进栈或入栈）和删除（出栈或是退栈）操作的线性表。它按照后进先出的原则存储数据，先进入的数据被压入栈底，最后进入的数据在栈顶，需要读数据的时候从栈顶开始弹出数据（最后进去的一个数据被第一个读出）。日常生活中有很多后进先出的例子，如碗柜里的一叠碗（图 3-10），先放进去的在下面，后放进去的在上面，取碗时只能从上面开始取，即先取到的是最后放上去的；又如冰糖葫芦串（图 3-11），先串进去的冰糖葫芦在下面，后串进去的冰糖葫芦在上面，吃的时候通常是后串进去的冰糖葫芦先吃到，而先串进去的冰糖葫芦后吃到。

图 3-10　碗柜里的碗

图 3-11　冰糖葫芦串

栈的抽象数据类型表示如下：

ADT Stack：

数据对象：$D=\{a_i \mid a_i \in ElemSet, i=1,2,\cdots,n,n>=0\}$

数据关系：$R=\{<a_{i-1}, a_i> \mid a_{i-1}, a_i \in D, i=2,\cdots,n\}$　#a_n 为栈顶

基本操作：

```
def __init__(self)   # 初始化一个空栈
def SEmpty(self)   # 若栈空，则返回 True，否则返回 False
def GetTop(self)   # 返回栈的栈顶元素
def Push(self,e)   #e 元素进栈
def Pop(self)   # 返回出栈元素
```

栈有两种存储方式，一种是用数组存储，这种方式存储的栈称为顺序栈；另一种用链表存储，这种方式存储的栈称为链式栈。一般根据进栈元素个数是否确定来选择存储结构，元素个数确定的用顺序栈，不确定的用链式栈。

栈的常用基本操作

1. 进栈

进栈就是从栈底开始，按顺序逐步向栈内添加数据。

算法思想：若栈不满，则先将栈顶指针增 1，然后送入进栈元素。栈满不能进栈。

```
    def Push ( self,  e):
            #e 为进栈元素，假设栈的容量为 size
        if (self.top==self.size-1):
            print(" 栈满，不能进栈 ")
        else:
            self.top=self.top+1
            self.stack[self.top]=e
    return
```

2. 出栈

出栈就是从栈顶开始，按逆序逐步将栈内数据取出。

算法思想：若栈不空，则将栈顶指针所指空间内容取出赋予变量，然后将栈顶指针减 1。栈空则没有元素可出。

```
def Pop (self):
    if self.top==-1:
        print(" 栈空，没有元素出栈 ")
    else:
        e=self.stack[self.top]
        self.top=self.top-1
        return e
```

项目六

探究文本字符的处理

——字符串的操作

电子表格等办公软件能方便地对文本字符进行如插入、删除、查找等编辑和处理（图 3-12）。这些文本字符的操作是如何实现的呢？文本字符如学生的姓名、性别等，在计算机世界中对应于字符串数据。字符串是非数值计算问题所要处理的主要对象之一，在文本编辑等方面使用非常广泛。因此，我们有必要了解字符串的概念及其基本操作。

D7 Audney

	A	B	C	D	E	F	G
1	学号	姓名	英文名	昵称	性别	年级	班级
2	170001	陈文昊	Chen, Wenhao	George	男	高一	1
3	170002	程飞	Cheng, Fei	Adam	男	高一	1
4	170003	顾小美	Gu, Xiao Mei	Mei	女	高一	1
5	170004	顾小文	Gu, Xiao Wen	Wen	女	高一	1
6	170005	姜宇晨	Jiang, Yuchen	Archie	男	高一	1
7	170006	李婷	Lee, Ting	Audney	女	高一	1
8	170007	林芝	Lim, Ji	Julia	女	高一	1
9	170008	沈伟	Shen, Wei	Matthew	男	高一	1
10	170009	王诗诗	Wang, Shi Shi	Sophia	女	高一	1
11	170010	吴宇	Wu, Yu	Bryan	男	高一	1
12	170011	徐弈杨	Xu, Yiyang	Bobo	男	高一	1
13	170012	于惠	Yu, Hui	Yvonne	女	高一	1
14	170013	余家成	Yu, Jiacheng	Jason	男	高一	1
15	170014	张友为	Chang, You-Wei	Peter	男	高一	1

图 3-12　学生信息表

项目学习目标

在本项目中，我们将一起模拟实现电子表格处理软件中文本字符的一些编辑处理功能，来学习字符串这一特殊的线性表数据结构。

完成本项目学习，须回答以下问题：

1. 文本字符可以进行哪些编辑处理？
2. 字符串是如何存储的？
3. 文本字符编辑处理对应的字符串操作有哪些？如何实现？

项目学习指引

1. 实现文本字符的编辑

用户在输入文本字符时，会发生漏输或多输，这时需要对输入的文本进行编辑修改，如插入或删除字符。使用文档处理软件进行编辑修改的操作很简单，只要定位光标，直接删除和插入即可，那么这些操作对于开发者来说是如何通过编程实现的呢？

核心概念

字符串（string）（简称串）：由零个或多个字符组成的有限序列。

字符串的长度：字符串中字符的个数。

在输入文本数据时，假设用数组存放一串文本字符，即字符串，实际上就是对数组进行赋值操作，如图 3-13 所示为存储学生信息表中学生李婷的昵称“Audney”的数组 s（以下都以存储单元加数组下标的形式表示）。

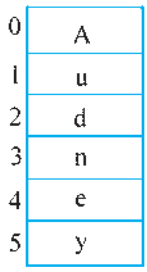


图 3-13　字符串的顺序存储

该字符串中有 6 个字符，称其长度为 6。

小贴士

字符串是特殊的线性表，即数据元素只有一个字符的线性表。字符串的插入、删除操作实现与一般的线性表相同。

若输入时，“Audney”错输成“Audneey”，多输了一个字母“e”，要在“Audneey”中删除这个“e”的过程如图 3-14、图 3-15 所示。

参见 P68 知识链接“字符串”；P70 知识链接“字符串的常用基本操作——删除操作”

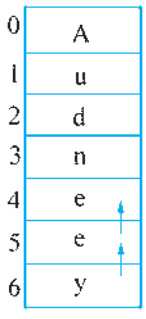


图 3-14　须删除字符后的字符依次前移覆盖被删字符

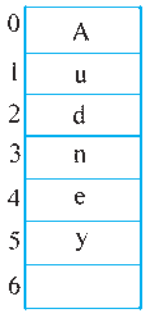


图 3-15　删除后

思考与讨论

1. 如果插入和删除的是多个字符的字符串，该如何处理？

2. 使用链表如何实现字符串的存储和删除？

假设漏输了字母 d，则在“Auney”中插入字符“d”的过程如图 3-16 所示。

参见 P70 知识链接“字符串的常用基本操作——插入操作”

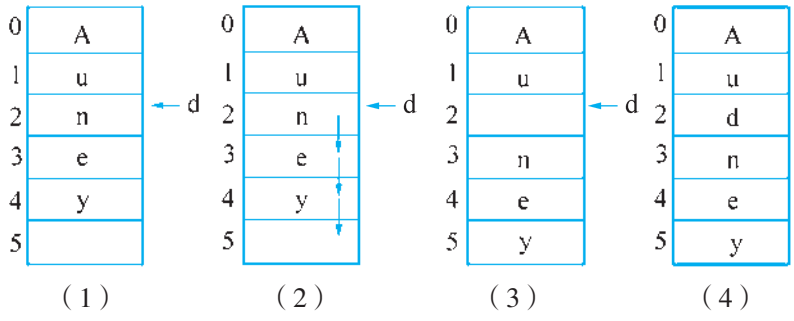


图 3-16　插入字符 d 的过程

活　动

6.1 尝试写出字符串抽象数据类型的定义。

6.2 画出字符串插入与删除的算法流程图，利用学过的数组知识，尝试完成下述代码，理解每一条语句的作用，并上机实践（为了便于理解算法思想，本项目所有活动采用 Python 列表模拟数组的方式，而不直接采用 Python 语言的字符串变量及相关操作）。

（1）插入程序：

```
s=['A', 'u', 'n', 'e', 'y']  # 定义 s 列表并赋值
n=len(s)  # n 为 s 列表长度
t= input(" 输入要插入的字符串 ")
m=len(t)
s.extend(t)   # 扩大 s 列表空间
j=n-1
i=3
while(j>=i-1):
    _________          # 元素后移，空出插入字符串的位置
    ______
for j in range(0,m):
    _________          # 依次送入元素
n=n+m
print(" 插入后的字符串为 ", s)
```

（2）删除程序：

```
s=['A', 'u', 'd', 'n', 'n', 'n', 'e', 'e', 'y']
print(s)
n=len(s)
i=5          #i 位置开始删除 m 个字符
m=3
j=i-1+m
while (j<n):
    _________          # 元素前移，覆盖删除部分
    _________
if (i-1+m>n-1):        # 释放多余的空间
    del s[i-1:n]
else:
    del s[n-m:n]
print(" 删除第 ",i," 位置，长度为 ",m," 的数据后：",s)
```

2. 实现文本的查找

假设要在电子表格软件中，查找姓名为“Gu，Xiao Wen”的学生。可以使用软件自带的查找功能，如图 3-17 所示。

图 3-17　查找功能

看似简单的查找操作，编程实现时，需要对字符串进行匹配，即使用字符串的比较操作。例如，英文名“Gu,Xiao Wen”与“Gu,Xiao Mei”比较是否相等的比较过程如图 3-18 所示。

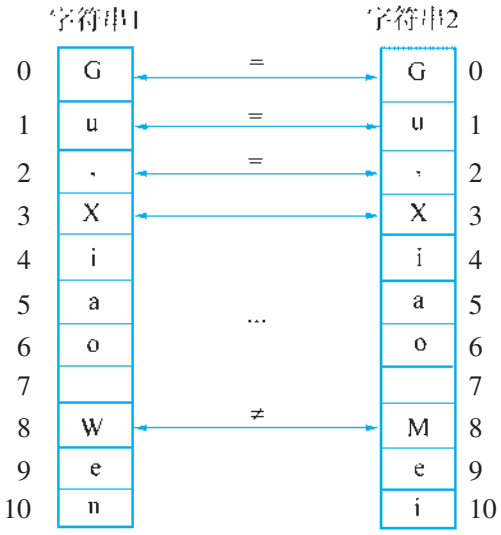


图 3-18　比较过程

即对两个字符串中的字符依次一一进行比较，直至遇到两个字符不相等或比较完所有字符，得出字符串不相等或相等的结论。

小贴士

比较操作：两个字符串的字符一一对应按字典序进行比较。可以进行相等、大于或小于的比较。

参见 P69 知识链接“字符串的常用基本操作——比较操作”

活　动

6.3　画出比较两个字符串 s 和 t 的算法流程图，尝试完成下述代码，理解每一条语句的作用，并上机实践。

```
s=[ 'G', 'u', ',', 'X', 'i', 'a', 'o', ' ', 'W', 'e', 'n'] # 定义字符串 s 列表
t= [ 'G', 'u', ',' , 'X', 'i', 'a', 'o', ' ', 'M', 'e', 'i'] # 定义字符串 t 列表
n=len(s)
m=len(t)
i=0
j=0
while (i<n and j<m):
    if(_____):
        print(" 字符串 s 小于字符串 t")
```

```
            break
        else:
            if(_____):
            print(" 字符串 s 大于字符串 t")
            break
        i=i+1
        j=j+1
if(_____):
    print (" 字符串 s 等于字符串 t")
```

3. 模拟实现文本函数的功能

电子表格软件还提供了许多函数，如图 3-19 所示，方便用户快速处理文本数据，如合并、截取字符串等。

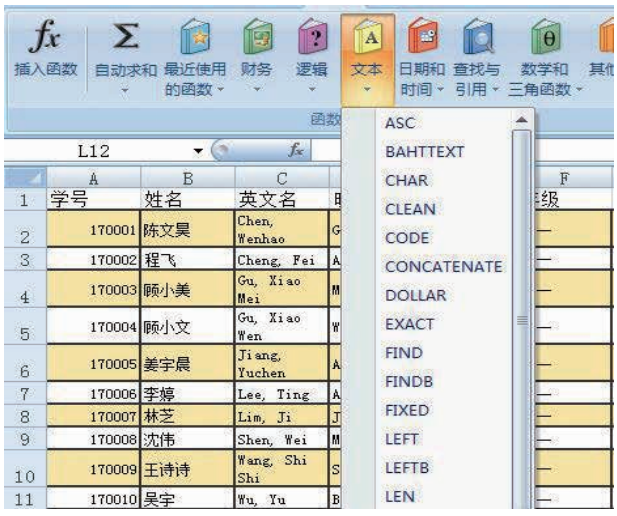


图 3-19　文本函数

思考与讨论

你还知道其他函数的功能和对应的字符串操作吗？

假设要把某学生的英文名和昵称合并在一起，可以使用软件提供的函数 TEXT，也可以使用“&”连接符；要截取英文名中的姓可以使用函数 MID。这些函数分别对应于字符串的连接操作和截取子串操作。

例如，英文名 s=“Cheng,Fei”，昵称 t=“Adam”，将英文名与昵称合并即 s+t 为“Cheng,Fei Adam”。实现过程如图 3-20 所示。

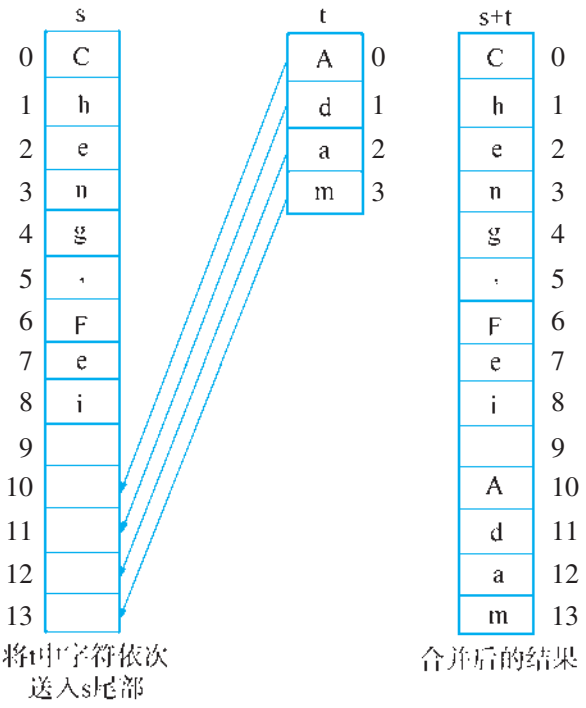


图 3-20　连接操作过程

若要在英文名“Cheng, Fei”中截取姓，即第 1 个位置开始截取 5 个字符，具体操作实现如图 3-21 所示。

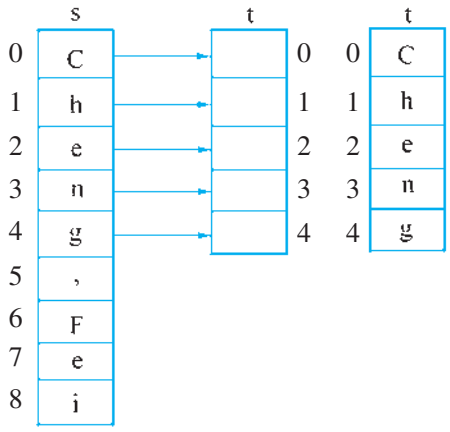


图 3-21　字符一一传送

小贴士

连接操作：将两个字符串连接成一个字符串。

子串：字符串中任意个连续的字符组成的子序列。

截取子串操作：在字符串中根据给定的起始、结束位置截取子串，或者根据起始位置和截取的长度来截取子串。

参见 P69 知识链接“字符串的常用基本操作——连接（合并）操作”

小贴士

注意第 1 个位置对应数组下标为 0，截取的子串在 t 数组中。

参见 P70 知识链接“字符串的常用基本操作——截取子串操作”“字符串的常用基本操作——复制操作”

活　动

6.4　画出将 t 字符串合并到 s 字符串的算法流程图，尝试完成下述代码，并理解每一条语句的作用。

```
s=[' ']*14          # 定义 s 列表空间
n=9
for j in range(0,n):          #s 列表赋值
    s[j]= input(" 输入一个字符 :")
t=['A', 'd', 'a', 'm']  # 定义 t 列表
m=len(t)
i=n
for j in range(0,m):
    _________
    _____
```

6.5　画出在 s 字符串中从第 i 位置开始截取长度为 m 的子串操作（结果保存在 t 中）的算法流程图，尝试完成下述代码，并理解每一条语句的作用。

```
s=['C', 'h', 'e', 'n', 'g', ',', 'F', 'e', 'i']          # 定义 s 列表
t=[' ']*5
n=len(s)
_______
_______
for j in range(0,m):
    if(i<n):
        _________
        i=i+1
```

知识链接

字符串

文档中的一个词组、一句话或选中的一段文字就是一个具体的字符串，在计算机中通常记为 s= “$a_1a_2a_3\cdots a_n$”，其中 s 是字符串的名，双引号括起来的字符序列称为字符串的值；a_i $(1 \leqslant i \leqslant n)$ 可以是字母、数字或其他字符，n 为字符串长度。长度为零的串称为空串。字符串是有限个字符组成的序列，它也是一种线性表，只是线性表中的每个数据元素只能是

字符类型，因此称字符串为特殊的线性表。字符串就是字符序列，是数据元素为字符型的线性表。因为字符串的数据元素类型是确定的字符类型，所以大部分程序设计语言都提供了字符串的存储和对字符串操作的函数，只要直接调用系统提供的字符串操作函数即可实现对字符串的操作。

字符串的抽象数据类型表示如下：

```
ADT String:
数据对象：D={a_i |a_i ∈ CharacterSet, i=1,2,…,n,n>=0}
数据关系：R={<a_i-1, a_i>| a_i-1, a_i ∈ D, i=2,…,n}
基本操作：
    def Assign(self,chars)  # 生成一个值等于 chars 的字符串
    def Copy(self,s)  # 复制字符串 s
    def Compare(self,s)
            # 比较操作，若大于 s 则返回 1；若等于 s 则返回 0；若小于 s 则返回 -1
    def Length(self)  # 返回字符串的元素个数，即求长度
    def Concat(self, s)  # 连接字符串 s
    def SubString(self, pos,len)  #pos 正确，返回第 pos 个字符起长度为 len 的子串
    def Index(self,s,pos)
            # 返回子串 s 在字符串中第 pos 个字符之后第一次出现的位置；若无则返回 0
    def Replace(self,s,t)  # 用 t 替换字符串中出现的所有的子串 s
    def Insert(self,pos,s)  # 在字符串的第 pos 个字符位置上插入 s
    def Delete(self,pos,len)  # 删除字符串中第 pos 位置开始长度为 len 的子串
```

字符串的常用基本操作

1. 比较操作

比较两个字符串 S 和 T 是否相等。

电子表格处理软件提供了对文本查找功能。字符串的比较是按字符的 ASCII 码值从左到右一一依次比较，直到出现不同的字符为止。

两个字符串除比较是否相等外，还有大于、小于的比较。例如，比较字符串“ABCDE”与字符串“ABRA”的大小，方法是从第一个字符开始，一一对应比较，因“C”>“R”为 false，所以“ABCDE”>“ABRA”为 false。而“ABCDE”<“ABRA”为 true。又如“bc”>“abcde”为 true，因为“b”>“a”为 true。

算法的基本思想是将存放两个字符串的数组对应字符元素一一比较，直至得出比较的结果。

2. 连接（合并）操作

若要将两个字符串合并，要进行连接操作，即将两个字符串连接成一个字符串，在

Python 语言中用 + 表示连接。例如，英文名 s=“Cheng,Fei”，昵称 t=“Adam”，s+t 为“Cheng,Fei Adam”，t+s 为“Adam Cheng,Fei”。C 语言用连接函数 strcat(s,t) 表示连接操作。

算法的基本思想是将第二个字符数组的元素插入到第一个字符数组的尾端。

3. 截取子串操作

文字处理软件中的选中文本一部分相当于截取子串操作。例如，若字符串 s=“Cheng，Fei”，用鼠标拖曳选中“Cheng”，“Cheng”就是“Cheng，Fei”的一个子串。当然“Cheng，Fei”可以有很多种子串。

在 Python 语言中截取子串操作是根据给定的开始位置和结束位置来截取子串的，例如，s=“Cheng，Fei”，那 s[0:5] 为 Cheng，s[6:9] 为 Fei。注意子串的尾字符是结束位置的前一个字符，若省略则截取到字符串尾。算法的基本思想是在字符数组中将要截取的子串复制到结果数组中。

4. 求长度操作

在 Python 语言中用 len(string) 可获得字符串 string 的长度。例如，s=“What is it?”，则 len(s) 为 11，注意字符串中的空格也是一个字符。

5. 复制操作

在 Python 语言中用赋值运算符 = 可实现复制操作。例如，str1=“student”，str2=str1，str1=“teacher”，结果是 str2 为“student”，str1 为“teacher”。

赋值操作是将字符串赋给一个字符串变量，Python 语言中用赋值运算符“=”实现赋值操作。算法的基本思想是将一个字符数组复制到另一个数组中。

6. 插入操作

字符串中任何位置上都可以插入字符串。例如，s=“Cheng，Fei”中第 11 个位置（下标为 10）上插入字符串 t=“Adam”，插入后 s=“Cheng，Fei Adam”。在 s=“Cheng，Fei”中第 7 个位置（下标为 6）上插入字符串 t=“Adam ”（末尾须有一个空格），插入后 s=“Cheng，Adam Fei”。在 Python 语言中没有直接使用的插入函数，可用截取子串和连接操作实现插入操作。使用数组存储时，算法的基本思想是先将插入位置上的元素依次后移空出空间，然后插入元素。

7. 删除操作

删除字符串中的子串操作。例如，s=“China Beijing Shanghai”中将第 14 个位置（下标为 13）开始的长度为 9 的子串删除，删除后 s=“China Beijing”。在 Python 语言中可以用 del 语句实现删除操作。使用数组存储时，算法的基本思想是将被删结点后面的数据元素依次往前移动覆盖前一结点（有些程序设计语言中字符串数组最后一个元素是结束符）。

单元挑战　按解密规则提取情报

一、项目任务

情报常常采用加密的方式传递，以防泄露，接收方在接收加密的情报（密文）后，按照事先订立的规则来提取情报。假设情报原文为中文，解密规则为：

（1）B → aA；

（2）A → ebh；

（3）加括号的字母序列变为逆序排列（去括号）；

（4）小写字母和汉字对应关系如表 3-1 所示：

表 3-1　字母—汉字对应表

a	b	c	d	e	f	g	h	i	j
不	相	他	间	要	是	陈	信	小	谍

请采用上述解密规则实现密文为“B（jdfcgi）”的情报提取。

二、项目指引

本项目要用到栈、队列和线性表。

1. 利用队列处理 B：aebh（由 B → aA，A → ebh 规则得）。
2. 利用栈处理：（jdfcgi）→ igcfdj。
3. 利用线性表（字母—汉字对应表）进行情报提取。

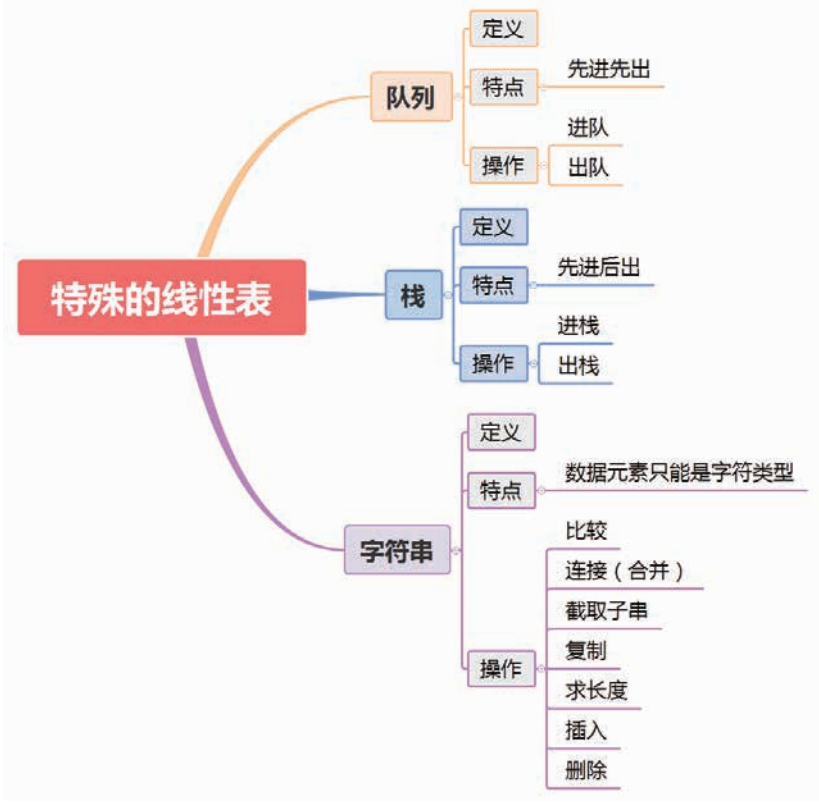
算法：

三、交流评价与反思

以自己熟悉的信息表达工具（如演示文稿等）制作电子作品，通过网络或课堂展示交流自己的算法和程序，并对他人的作品进行评价。

单元小结

一、主要内容梳理



二、单元练习

1. 假设有如下功能的演唱会门票预订系统：

（1）按先到者先订的规则进行预订。

（2）当门票都预订完后，允许在队伍中等待退票。

请画出算法流程图并编程实现排队预订及等候退票功能。

2. 四方向迷宫问题求解：假设有 8×8 的迷宫如下所示（迷宫用二维数组存放，可查阅资料了解二维数组），1 表示障碍不能通行，0 表示通道可以通行。求解方法为：从入口开始出发沿着某一个方向向前试探，若能够通行，则继续试探前行；如果遇到障碍，马上原

路返回，并在该路口做上记号，换另一个方向继续试探前行；重复上述步骤直到到达迷宫的出口后结束。由于迷宫错综复杂，靠人力来搜索迷宫的所有通路是很费时、费力的一件事，因此可以借助计算机编程来完成。

入口 0 0 1 0 0 0 1 0
0 0 1 0 0 0 1 0
0 0 0 0 1 1 0 0
0 1 1 1 0 0 0 0
0 0 0 1 0 0 0 0
0 1 0 0 0 1 0 0
0 1 1 1 0 1 1 0
0 0 0 0 1 0 0 0 出口

（1）画出求解迷宫问题中，栈的变化过程。

（2）设计解四方向迷宫问题的算法并编程实现。

提示：在编程时，为避免考虑第一行不能朝上走、第一列不能朝左走，最后一列不能朝右走，最下一行不能朝下走，可以在迷宫外围设置一圈“围墙”，这样所有的位置都可考虑四个方向，围墙等同于障碍（用 1 表示），如下所示。

1 1 1 1 1 1 1 1 1 1
1 0 0 1 0 0 0 1 0 1
1 0 0 1 0 0 0 1 0 1
1 0 0 0 0 1 1 0 0 1
1 0 1 1 1 0 0 0 0 1
1 0 0 0 1 0 0 0 0 1
1 0 1 0 0 0 1 0 0 1
1 0 1 1 1 0 1 1 0 1
1 0 0 0 0 1 0 0 0 1
1 1 1 1 1 1 1 1 1 1

3. 假设有一篇英文短文，请画出算法流程图并编程实现以下操作：

（1）统计英文短文中某单词出现的次数。

（2）在英文短文中查找并替换某单词。

三、单元评价

评 价 内 容	达 成 情 况
能列举队列的应用场合（A）	
能理解队列是先进先出受限制的线性表（T）	
能通过进队和出队操作模拟实现电子排队预订（T）	
能列举栈的应用场合（A）	
能理解栈是后进先出受限制的线性表（T）	
能通过进栈和出栈操作模拟实现撤消操作（T）	
知道电子表格等软件的文本处理功能是通过编程实现的；能列举字符串的实际应用场合（A、T）	
能理解什么是字符串（T）	
能通过字符串的比较操作实现查找功能；能通过字符串的连接操作实现文本的合并；能通过字符串的截取操作实现文本的删除等（T）	
能综合使用队列、栈和字符串，实现情报的提取（T、I、R）	

说明：A—信息意识，T—计算思维，I—数字化学习与创新，R—信息社会责任

第四单元

二叉树

大自然处处可见到的树，由树根、树干、树枝和树叶组成。在现实生活中，常用家族树（家谱）描述某一个家族，或用树状图描述某行政管理架构，它们的共同特点是图的形状上部窄小，下部宽大，类似于倒置的树。在计算机中，树是一种数据结构，在第二单元中已有初步介绍，它属于非线性的数据结构，数据元素间具有一对多的关系。其中有一种比较特殊的树，即二叉树，应用较广泛，本单元一起来学习二叉树这一数据结构，了解它的一些常用的基本操作。

学习目标

- 了解二叉树概念。
- 了解二叉树的基本操作。

单元挑战

使用二叉树解简单背包问题

项目七

探究计算机中算术表达式的计算
——了解二叉树及其基本操作

日常生活中，人们经常要进行四则运算，如计算 3×（4+5）-7，相信大家都知道该计算式正确的运算顺序，但是在计算机中输入这个计算式后（图 4-1），计算机是如何判断运算顺序并进行运算的呢？这样的计算式在计算机中称为算术表达式，书写为 3*（4+5）-7，须将其转换成方便计算机处理的表达式形式，并通过一定的算法来让计算机实现正确运算。

图 4-1 “计算器”应用程序界面

项目学习目标

在本项目中，我们将一起探究如何使用二叉树将算术表达式转换成计算机可以正确执行运算的表达式。

完成本项目学习，须回答以下问题：

1. 什么是二叉树？

2. 算术表达式如何用二叉树表示的？

3. 如何从二叉树得到后缀表达式？二叉树有哪些常用基本操作？

4. 采用后缀表达式是如何让计算机完成计算的？

项目学习指引

1. 探究计算机中算术表达式的计算原理

算术表达式的值是按运算符间优先级从高到低来计算的，这与四则运算的规则完全相同。由于算术表达式的多样性和复杂性，人们很难设计很好的算法编程来求解任意表达式的值。历史上，有一位逻辑学家提出一种表达式，这种表达式的每一运算符都置于其运算对象之后，故也称为后缀表达式，如表达式 1+2 和 3*（4+5）–7 的后缀表达式分别为 12+ 和 3 4 5 + * 7 –。可以看出，运算符是按原表达式优先级排列的，原本括号内的运算符排在了最前面，这样方便计算机按顺序来计算表达式，后缀表达式的计算过程如下所示：

3 4 5 + * 7 –

3　9*

27 7 –

20

使用后缀表达式的优势在于只用进栈和出栈操作可以完成任意算术表达式的运算。其运算方式为：从左到右读取后缀表达式，如果当前字符为变量或者为数字，则进栈；如果是运算符，则将栈顶两个元素（运算对象）弹出作相应运算，结果再进栈，重复上述过程，直至表达式读取完，栈里就剩下最终结果。

小贴士

每步运算中，参与的运算对象是运算符前的两个数字。

活　动

7.1　结合第三单元所学的栈的相关知识，按上述方法，画出通过进栈和出栈完成后缀表达式 3 4 5 + * 7 – 计算的过程。

2. 探究为何二叉树能将算术表达式转换为后缀表达式

如何能将算术表达式转换为后缀表达式呢？有一种方法是使用二叉树（binary tree）来转换。二叉树是有一个根结点（无前驱），每个树权都最多只有两个分支的树（可以是 0 个、1 个或 2 个），根结点左边的分支称为左子树、右边的分支称为右子树。结点的后继结点为左孩子和右孩子，没有分支的结点称为叶结点。树枝的方向都是往下的，箭头省略，如图 4-2 所示。

核心概念

二叉树是由 n 个结点的有限集合构成，n=0 称为空二叉树，n>0 时由一个根结点及两棵互不相交的左右子树组成，并且左右子树都是二叉树。

小贴士

树是一种重要的非线性数据结构，直观地看，它是数据元素（在树中称为结点）按分支关系组织起来的结构，与自然界中的树很像。

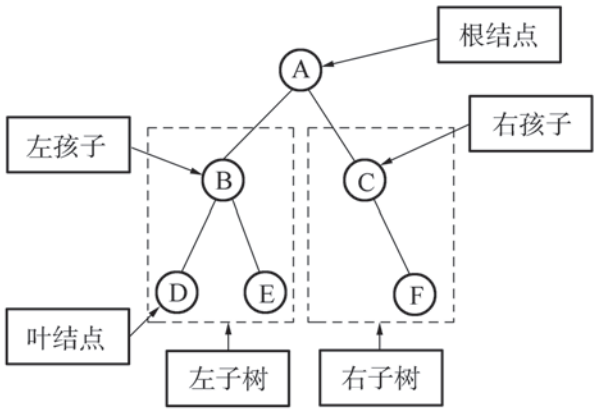


图 4-2　二叉树示例

图中所示的左子树和右子树是根结点 A 的左子树和右子树，在左子树中，结点 B 可以看成是该子树的根结点。而叶结点 D 和 E 可以看成是结点 B 的左子树和右子树（只有一个结点的子树），也是左孩子和右孩子。

参见 P82 知识链接“树”；P83 知识链接“二叉树”

思考与讨论

结点 C 有左子树和右子树吗？

了解了二叉树后，依据某一规则对二叉树的结点依次进行访问，就可以得到一个序列。假设有图 4-3 所示二叉树，我们采用先访问其左子树，然后访问根结点，再访问右子树的规则（所有子树也按此规则进行访问）。

小贴士

每个结点只访问一次。

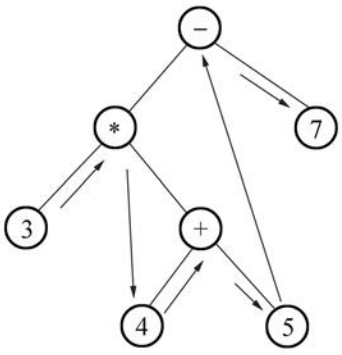


图 4-3　按序访问二叉树结点

具体就是先访问根结点“-”的左子树，由于该左子树的根结点“*”也有自己的左子树即结点 3，而结点 3 并无左子树，所以最先访问的是结点 3，继而是“*”……最后访问的是根结点“-”的右子树，即结点 7。

通过以上规则访问结点后，便可以得到 3*4+5-7 的序列。这样的序列称为中缀表达式，中缀表达式的运算符都位于运算对象的中间。中缀表达式的顺序是和算术表达式一致的，只是缺了括号。

上述访问规则称为二叉树的中序遍历，可以看出，中序遍历的序列对应了中缀表达式。

小贴士

遍历（traverse）是指沿着某条搜索路线，依次对每个结点均做一次且仅做一次访问。

思考与讨论

1. 如果没有结点 3，最先访问的是哪个结点？
2. 如果结点 7 有左孩子，最后访问的是哪个结点？
3. 如何可以在中序遍历后得到带括号的算术表达式？

与中序遍历相应的还有先序遍历和后序遍历。先序遍历是先访问根结点，再访问左子树，最后访问右子树（对于所有子树也采用此规则遍历），如图 4-4 所示。

参见 P84 知识链接“二叉树的常用基本操作”

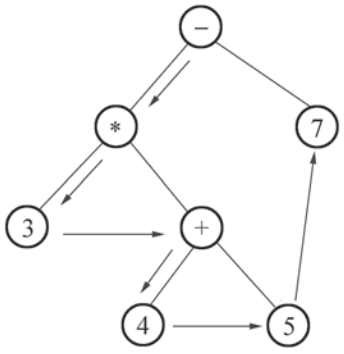


图 4-4　先序遍历

可以看出，先序遍历第一个访问的一定是根结点。

后序遍历是先访问左子树，再访问右子树，最后访问根结点（对于子树也采用此规则遍历）。而上图所示二叉树的后序遍历得到的序列恰好对应了后缀表达式 3 4 5 + * 7 −。

活　动

7.2 参照先序遍历和中序遍历的方式，画出上述表达式二叉树的后序遍历得出后缀表达式的过程。

3. 构建二叉树

如何能得到上述二叉树呢？算术表达式的运算符都是二元运算符，即每个运算符对应两个运算对象。先前说到按中序遍历得到的中缀表达式最贴近于算术表达式，因此，将运算符作为根结点，运算符前的运算对象作为左子树，运算符后的运算对象作为右子树（运算对象也可以是算术表达式）来构建二叉树。按中序遍历的方式访问该二叉树，正好能得到中缀表达式，如图 4-5 所示。

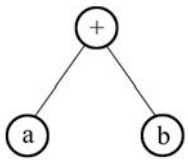


图 4-5　a+b 的表达式二叉树

按照此规则，对于算术表达式 3*（4+5）-7，我们把优先级最低的运算符“-”作为根结点，两边的运算对象作为左子树和右子树。对于左边的运算对象即表达式 3*（4+5），可以按同样规则来构建左子树，以此类推，最终建立如图 4-6 所示的表达式二叉树。对于这一表达式二叉树进行后序遍历可以得到后缀表达式。

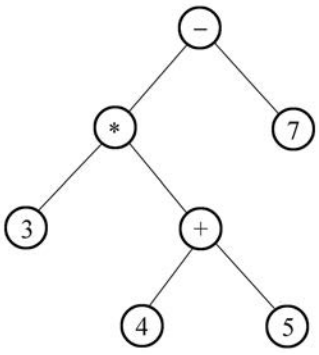


图 4-6　表达式二叉树

思考与讨论

先序遍历得到的表达式称为前缀表达式，前缀表达式是否和后缀表达式一样能用栈完成计算呢？

小贴士

此类二叉树称为表达式二叉树。

活　动

7.3　对于算术表达式 5*7+8*(4–2)，请为其构建表达式二叉树，并通过后序遍历的方式，将算术表达式转换成后缀表达式，然后用入栈和进栈操作来计算结果，画出表达式二叉树和栈操作示意图。

知识链接

树

日常生活中很多事物可以用树形图来表示，如家族图谱、行政管理架构等，如图 4–7、图 4–8 所示。

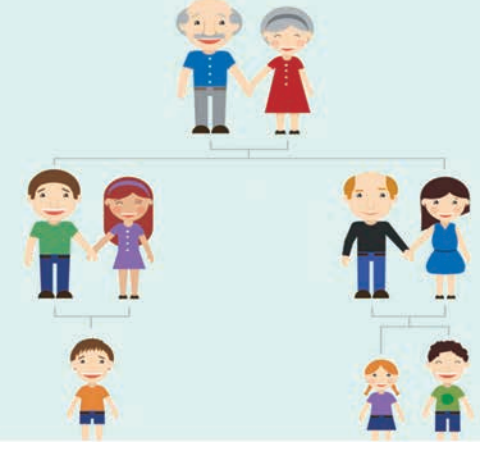


图 4–7　家族树

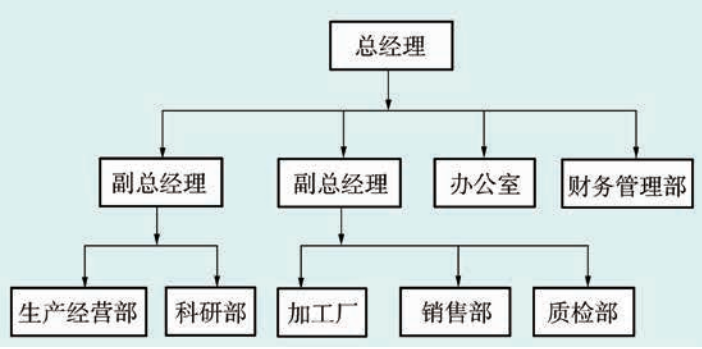


图 4–8　行政管理架构图

在数据结构中，树（tree）是 n（n>=0）个结点的有限集合 T，T 为空时称为空树，否则它满足如下两个条件：

（1）有且仅有一个特定的称为根(root) 的结点，如右图中的结点 A；

（2）其余的结点可分为 m（m>=0）个互不相交的子集 T_1，T_2，T_3，…，T_m，其中每个子集又是一棵树，并称其为子树（subtree），如图 4-9 中的子树 T_1，T_2，T_3。

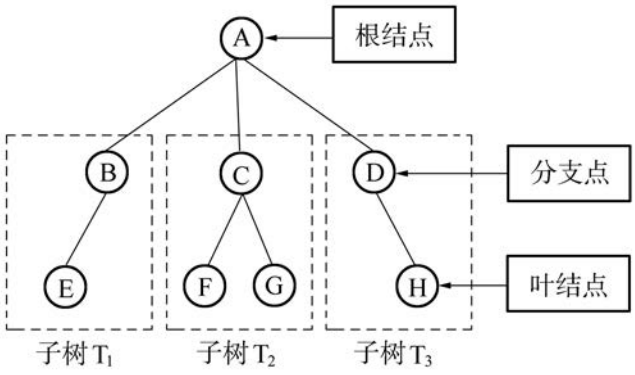


图 4-9　树示例

二叉树

二叉树是由 n（n>=0）个结点的有限集合构成，n=0 称为空二叉树，n>0 时由一个根结点及两棵互不相交的左右子树组成，并且左右子树都是二叉树，如图 4-10 所示。

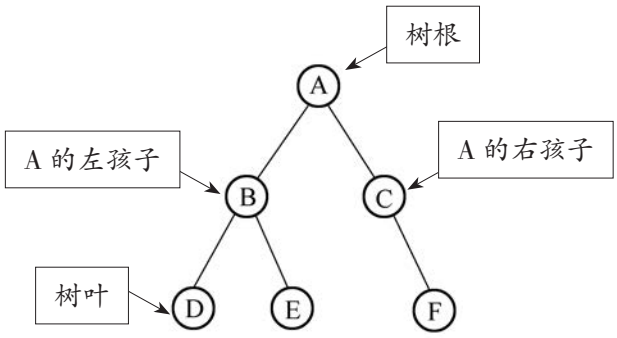


图 4-10　二叉树示例

（1）树根（或称根结点）：没有前驱的结点称为树根，一棵非空二叉树有且仅有一个树根，如图中的结点 A。

（2）结点的度：结点的后继数。如图中结点 A 的度为 2，结点 C 的度为 1，结点 F 的度为 0。

（3）分支点：有后继的结点，也称内结点，即结点的度不为零，如结点 B 和结点 C。

（4）树叶（或称叶结点）：无后继的结点，也称终端结点，即结点的度为零，如结点 D、E、F。

（5）树的深度：树的层次数。上图中二叉树为三层，根结点为第一层，结点 B、C 为第二层，结点 D、E、F 为第三层。

（6）左孩子：直接左后继结点，如结点 D 是结点 B 的左孩子；右孩子：直接右后继结点，如结点 E 是结点 B 的右孩子。

（7）双亲结点：直接前驱结点，如结点 C 是结点 F 的父结点。

二叉树的抽象数据类型表示如下：

ADT BinaryTree：

数据对象：具有相同特性的数据元素集合。

数据关系：数据元素集合中仅有一个结点无前驱元素（根）；其余结点有且仅有一个前驱结点，每个结点最多有两个后继结点且有左右之分；每个结点到根都存在一个线性序列。

基本操作：

```
def __init__(self)    # 初始化一棵空二叉树
```

```
def BTEmpty(self)   # 若二叉树为空，则返回 True，否则返回 False
def Root(self)   # 返回树根
def Value(self,e)   # 返回 e 结点的值
def Depth(self)   # 返回树的深度
def Parent(self,e)   # 返回 e 的双亲
def Leftchild((self,e)   # 返回 e 的左孩子
def Rightchild((self,e)   # 返回 e 的右孩子
def PreOrderTraverse(self)   # 先序遍历二叉树
def inOrderTraverse(self)   # 中序遍历二叉树
def PostOrderTraverse(self)   # 后序遍历二叉树
```

二叉树的常用基本操作

二叉树的基本操作主要有二叉树的创建、二叉树的遍历、求二叉树的深度等。其中二叉树的遍历方法主要有以下几种。

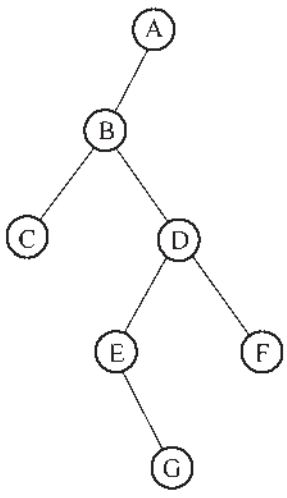
（1）先序遍历：先访问二叉树根结点，然后先序遍历左子树，再先序遍历右子树。

（2）中序遍历：先中序遍历左子树，然后访问二叉树根结点，再中序遍历右子树。

（3）后序遍历：先后序遍历左子树，再后序遍历右子树，最后访问二叉树根结点。

（4）层次遍历：对二叉树从上到下，逐层从左到右访问每个结点。

前三种属于深度优先遍历，而层次遍历属于广度优先遍历，具体示例如图 4-11 所示。



先序遍历序列为：ABCDEGF
中序遍历序列为：CBEGDFA
后序遍历序列为：CGEFDBA
层次遍历序列为：ABCDEFG

图 4-11　遍历示例

单元挑战　使用二叉树解简单背包问题

一、项目任务

假设现有一个载重量最多为 10 千克的背包，以及 5 件物品。其中，第一件物品的重量和价值分别为 3 千克和 9 元；第二件物品的重量和价值分别为 2 千克和 8 元；第三件物品的重量和价值分别为 4 千克和 7 元；第四件物品的重量和价值分别为 7 千克和 16 元；第五件物品的重量和价值分别为 6 千克和 15 元。那么，此背包中可以放哪些物品，使得重量总和不超过 10 千克，而价值总和最大呢?

请构建二叉树，列出所有可能的物品放置策略，并选出其中的最优策略（背包里所有物品的价值总和最大），比比哪一组的速度快。

二、项目指引

本项目求解的关键是，判断如果当前背包里的重量与下一件要放入物品的重量的和未超过重量限制，则放入下一件物品；超过重量限制，则不放入下一件物品，而后在两种情况下分别继续同样的判断，如此反复，直至考虑完所有物品。过程中的每一次判断相当于产生了两条策略路径，这与二叉树的树形很像，如图 4-12 所示。每一个分支结点都表示背包的一种状态，而叶结点表示得出的一种策略。每个结点的左孩子表示放入下一件物品的背包状态（一旦无法放下一件物品则无左孩子），每个结点的右孩子表示不放入下一件物品的背包状态。

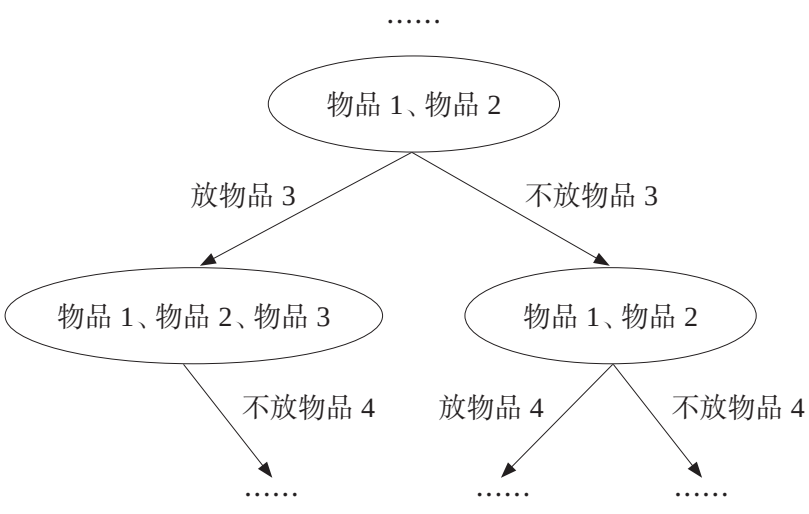


图 4-12　示例

（1）假设根结点为背包空的状态，画出二叉树。为方便计算和加快速度，将物品以（重量、价值）的形式，按物品顺序从上到下排列在二叉树对应的每一层旁，思考物品的排列顺序对构建二叉树有何影响。

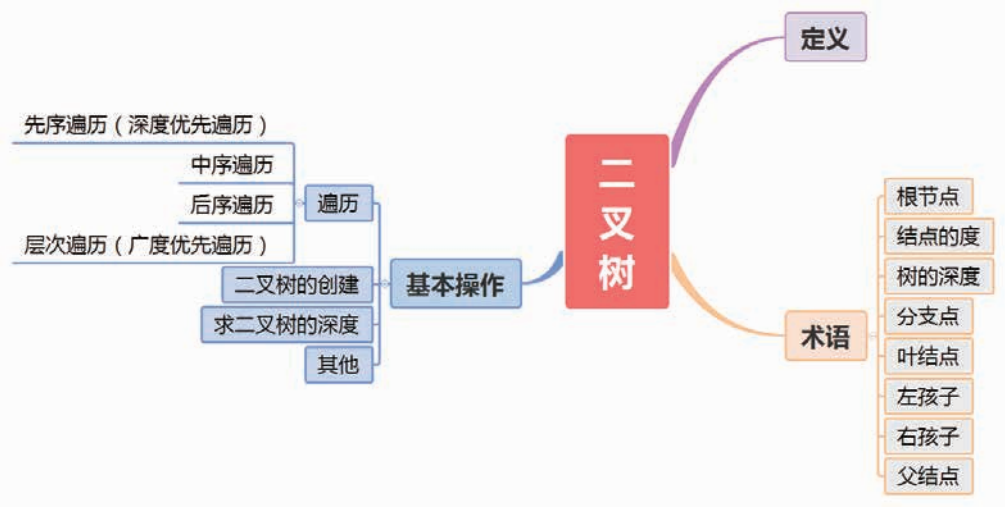
（2）写出最优策略。

三、交流评价与反思

以自己熟悉的信息表达工具（如演示文稿等）制作电子作品，通过网络或课堂展示交流构建的二叉树和求得的最优策略。并对他人的作品进行评价。

单元小结

一、主要内容梳理



二、单元练习

1. 假设有一个八位二进制编码，通过图 4-13 所示二叉树的后序遍历可以得到该二进制编码，请写出该编码。

2. 已知算术表达式 8-（6-4）*2+5，构建并画出表达式二叉树，将该算术表达式转换成前缀表达式，尝试利用前缀表达式完成计算。

3. 在先序遍历序列中添加空子树∮标记后可以唯一确定二叉树。已知某二叉树带标记的先序序列为：ABD ∮∮ EF ∮∮ C ∮ G ∮∮∮，画出该二叉树，并写出它的中序遍历和后序遍历序列。

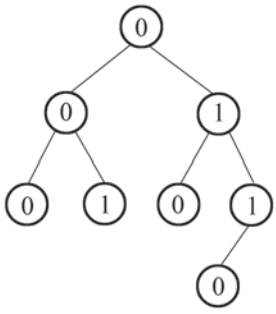


图 4-13　二叉树

三、单元评价

评 价 内 容	达 成 情 况
理解计算机处理算术表达式与人计算算式的不同（A）	
了解什么是树与二叉树（T）	
了解二叉树生成后缀表达式、中缀表达式、前缀表达式的原理（T）	
了解二叉树的先序遍历、中序遍历和后序遍历（T）	
了解二叉树有哪些基本操作（T）	

说明：A—信息意识，T—计算思维，I—数字化学习与创新，R—信息社会责任

第五单元

排序与查找

算法对于程序设计来说是非常重要的，一个好的算法可以提高程序的运行效率。无论是人工智能机下棋程序、智能搜索引擎，还是自动解魔方机器人，都离不开好的算法。例如，战胜世界围棋冠军的 AlphaGo 就使用了蒙特卡罗树搜索。同时，算法又离不开数据结构。通常说，算法 + 数据结构 = 程序，那么数据结构和算法又有着什么样的关系呢？其实同学们在前面单元已经经历过设计算法编写程序的过程，可以从中感受到一点数据结构与算法的关系。本单元将通过具体的排序和查找算法来带领大家进一步体会数据结构与算法的关系。

学习目标

- 掌握常用的数据排序方法。
- 掌握常用的数据查找方法。
- 理解迭代概念。
- 理解递归概念。
- 进一步理解算法与数据结构的关系。

单元挑战

使用二叉查找树查找学生成绩信息

项目八

模拟实现商品排序

——常用排序算法及其比较

在日常生活中经常可以看到各种排行榜，如关注度排行榜、流行歌曲排行榜、球队实力排行榜、销量排行榜（图 5-1）等。许多网络购物平台提供当月最畅销的商品排行。假如某店铺中某一种类商品只有 6 件不同价格的商品，若采用人工排序，由于数据量小，简单目测一下就可以得出排序结果。但实际上大部分情况下的排序数据可能会是成千上万条，因此，采用人工的方式会耗时多，且容易出错。若采用合理的数据结构和算法，编程实现后，系统便能对大量数据进行自动排序，省时省力，且只要程序正确，基本不会出错。

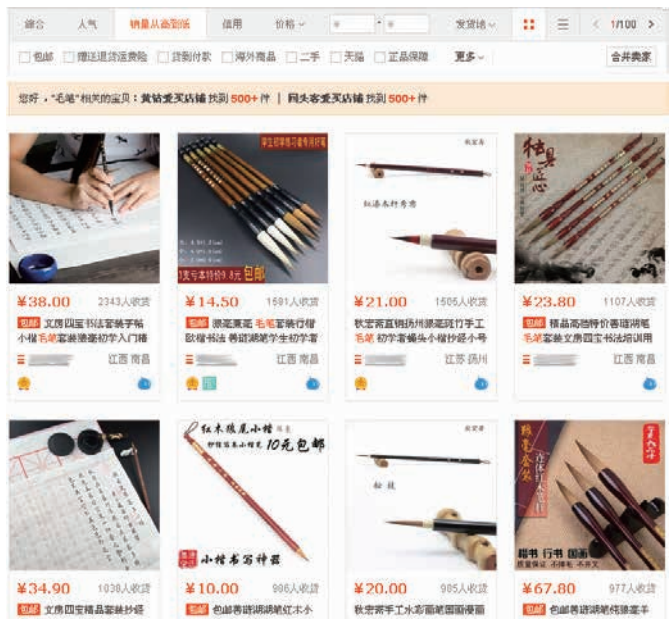


图 5-1　按销量排序页面

项目学习目标

在本项目中，我们将一起使用不同的排序算法模拟实现商品排序的功能，从中了解排序算法与数据结构之间的关系。

完成本项目学习，须回答以下问题：

1. 商品排序是如何实现的？
2. 什么是排序？计算机中有哪些常用的排序算法？
3. 排序算法各有何特点？
4. 排序算法与数据结构之间的关系是什么？

项目学习指引

1. 尝试使用插入排序法实现商品销量排序

若某家店铺某一类商品 6 个品种的月销量分别是 21，25，49，25，16，8，要对这些商品按销量进行排序，可采用顺序存储结构下的插入排序法，部分过程如图 5-2 所示。

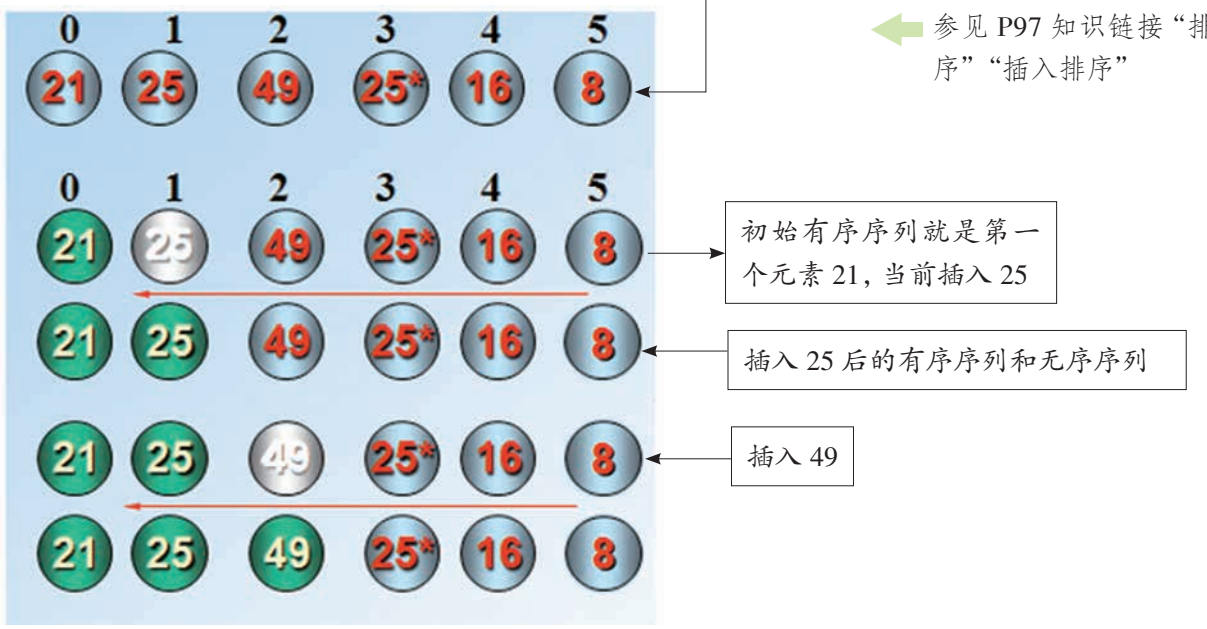


图 5-2　插入排序部分过程

先将初始无序序列的 6 件商品销量数据存放在下标 0 ~ 5 某一数组中，然后将第一个商品销量数据 21 视作初始有序序列。将剩余的 5 件商品销量数据视作无序序列。接下来每次从无序序列中取第一个商品销量数据插入到有序序列的适当位置上。为了操作方便，插入的商品销量数据与有序序列中商品销量数据从后往前依次比较，一边比较一边将大的商品销量数据往后移动，直至找到插入位置。

思考与讨论

1. 商品除了可以按销量排序外，还可以按什么排序？
2. 生活中还有哪些排序的例子？

核心概念

排序（sort）：将无序序列调整为有序序列。

参见 P97 知识链接“排序”“插入排序”

小贴士

此处的插入排序称为直接插入排序。

序列中有两个商品销量数据都为 25，为区分这两个 25，将后一个 25 加星号以示区别。

活 动

8.1 参照上述步骤画出剩下的排序过程图。

8.2 若采用链式存储结构，请根据图 5-3 所示，用文字描述排序过程，并画出每一个过程图。谈谈使用顺序存储结构和链式存储结构实现插入排序，各有何特点，以及具体的算法是否相同；在数据量小和大的情况下分别用哪一种比较好。

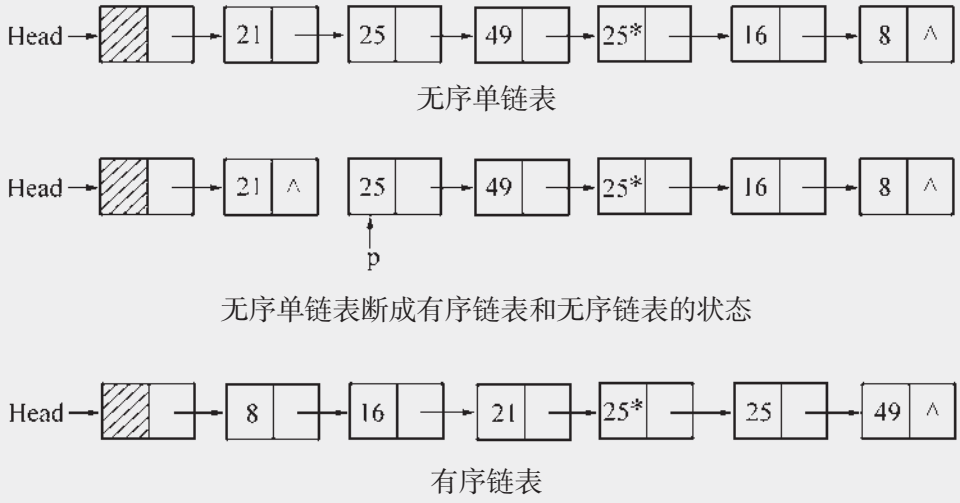


图 5-3 采用链式存储结构实现插入排序

插入 25：

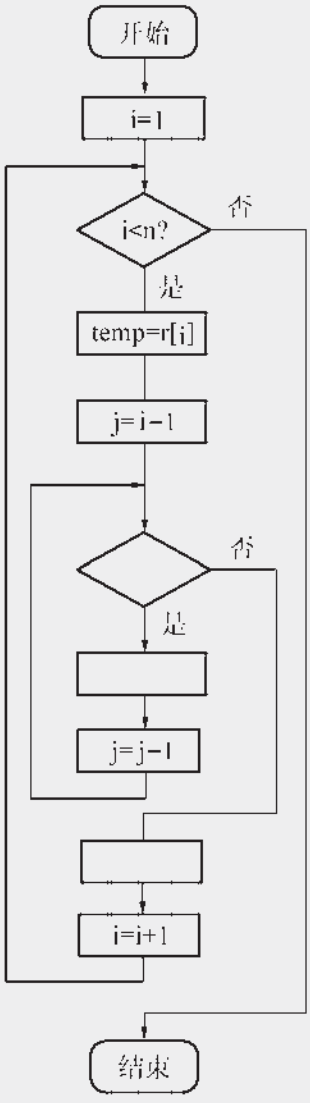
插入 49：

插入 25*：

插入 16：

插入 8：

8.3 在以下流程框图（局部）中完成插入排序算法（设数据在 r 数组中，n 为元素个数）；编程实现对大量商品数据进行排序，并运行测试。



程序：

小贴士

冒泡排序法是交换排序方法中的一种，因其排序过程如同气泡上冒的过程一般而形象地被称为冒泡排序法。

参见 P98 知识链接“交换排序”

2. 尝试使用冒泡排序法实现商品销量排序

若采用冒泡排序法，则将这些销量数据看成水里的气泡，数字大小表示气泡的轻重，最先冒出最轻的（最小数），第二次冒出第二轻的，以此类推，如图 5-4 所示，图中斜线下是无序的，斜线上是已冒出的有序的。实现冒泡的方法：从序列最后两个元素开始比较，只要后面元素小，则交换；交换上去的再与最后第三个元素比较，以此类推，这样两两相邻元素比较就冒出当前最小元素。

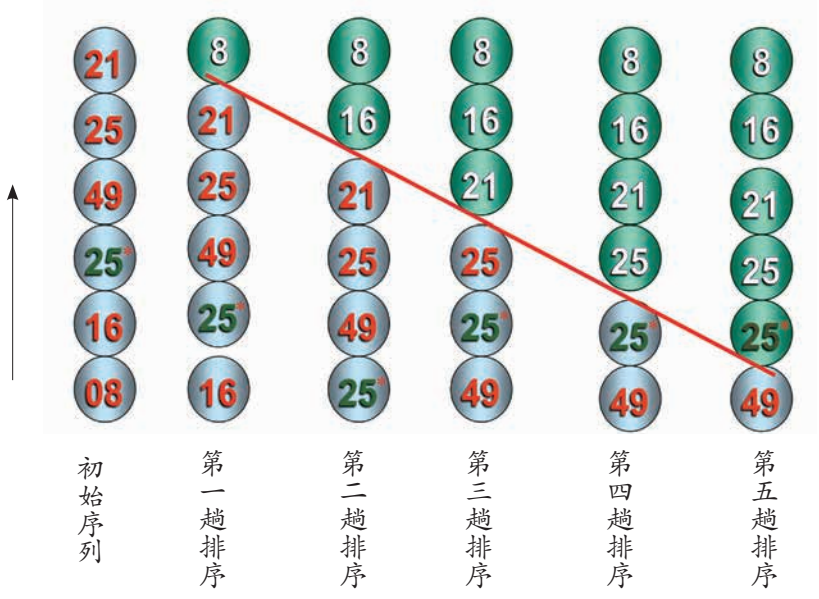


图 5-4　采用数组的冒泡排序过程

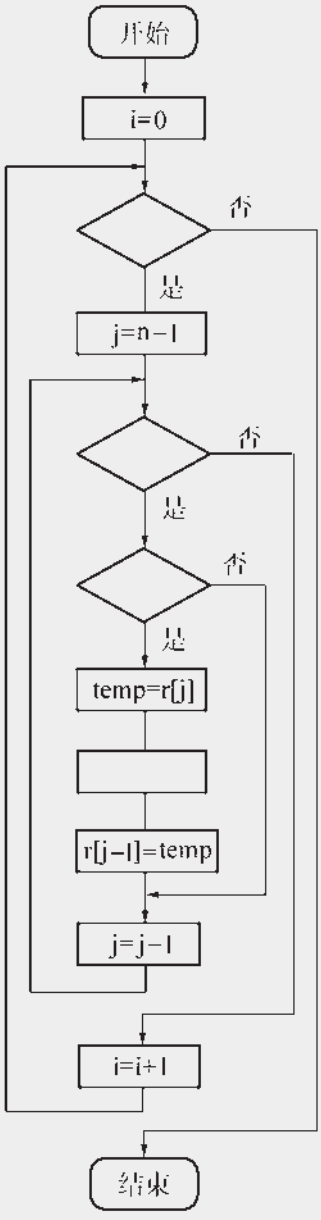
思考与讨论

1. 用小的数上冒可以完成排序，那用大的数下沉能完成排序吗？

2. 冒泡排序在链式存储结构下的排序过程有何不同？

活　动

8.4 在以下流程图（局部）框中完成冒泡排序算法（设数据在 r 数组中，n 为元素个数），并编程实现，上机运行测试。



程序：

3. 尝试使用选择排序法实现商品销量排序

若采用选择排序，每趟排序将未完成排序序列中的最小数依次和未完成排序序列中第一个数交换，过程如图 5-5 所示。

参见 P98 知识链接“选择排序”

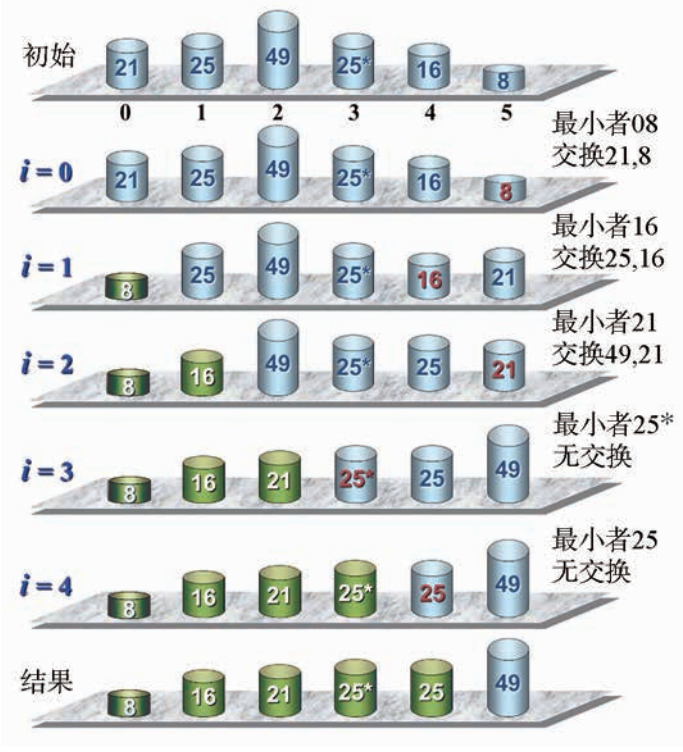


图 5-5　选择排序各趟排序后的结果

小贴士

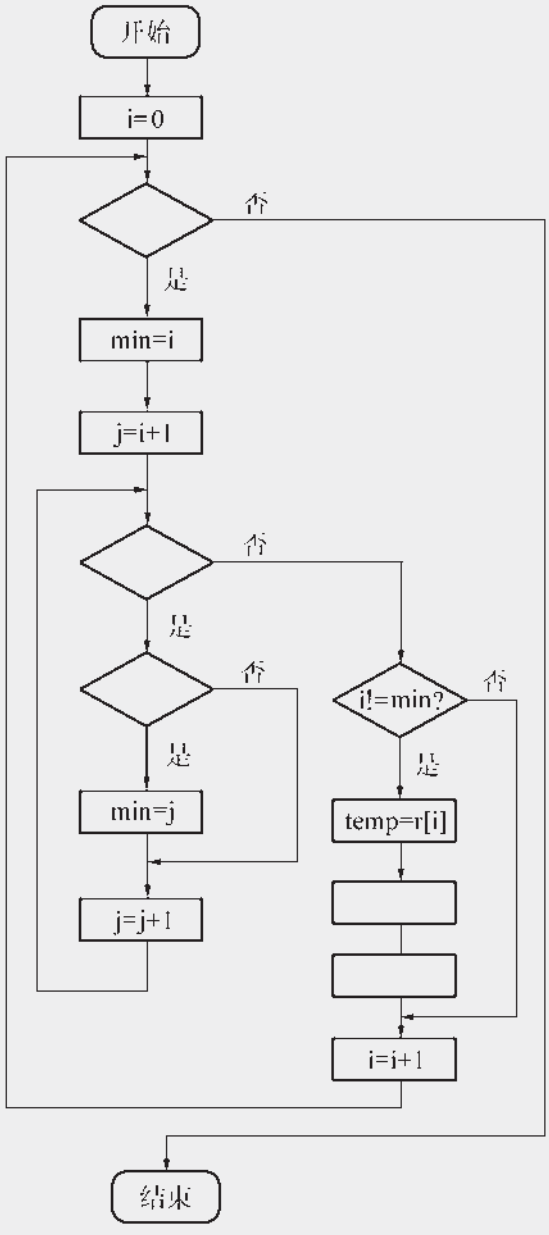
此种选择排序称为直接选择排序。

思考与讨论

选择排序结点的移动次数与序列的初始排列状态有关，那结点的比较次数与序列的初始排列状态有关吗？

活　动

8.5　在以下流程框图（局部）中完成选择排序算法（设数据在 r 数组中，n 为元素个数），编程实现，并运行测试。



程序：

4. 比较三种排序方法

上述三种排序算法都能实现商品销量排行榜，但每种排序算法都有各自的特点，如表 5-1 所示。

小贴士

稳定性：待排序列中相同关键字（如 25、25*）排序后相对位置不发生变化的排序称为稳定的排序。

参见 P99 知识链接“稳定性”

表 5-1　排序方法基本特点比较

排序方法	基本特点	稳定性
插入排序	1. 算法相对简洁。 2. 待排序列基本有序情况下排序速度快。	稳定
冒泡排序	1. 不需要完成所有数据排序就可得排行榜的前几名。 2. 待排序列基本有序情况下排序速度快。 3. 算法较容易理解。	稳定
选择排序	1. 排序过程中数据的移动次数少。 2. 不需要完成所有数据排序就可得排行榜的前几名。 3. 算法较容易理解。	不稳定

活　动

8.6 请针对上述表格中的各排序方法特点和稳定性给出具体的理由。

8.7 某商品品种繁多，假设有 1000 种，已知商品是按商品编号排列的，现要求尽快地找出价格最低的前 10 种商品，若有价格相同就按商品编号顺序排列，那么选用哪种排序方法才能较好地完成此项工作呢？请给出理由，并编程实现。

知识链接

排序

将一组次序任意的数据元素转变为按其关键字（可以标识数据元素的某个数据项的值）递增（或递减）次序排列的过程，称为排序。例如，表 5-2 所示商品销售表可以按序号（主关键字，可以唯一标识一个数据元素）递增进行排序，也可以按单价或销量（次关键字，可识别若干数据元素）进行排序。排序后，每个数据元素的顺序会按关键字的顺序排列。

排序方法有插入排序（直接插入排序、折半插入排序、希尔排序等）、交换排序（冒泡排序、快速排序）、选择排序（直接选择排序、堆排序等）、归并排序等。

表 5-2　商品销售表

序号	商品名称	单价（元）	销量（件）
0	文具盒 1	10	190
1	文具盒 2	15	170
……	……	……	……
n-1	文具盒 3	13	205

1. 插入排序

常用的插入排序有直接插入排序、折半插入排序和希尔排序。本单元项目采用的是直接插入排序。直接插入排序的思想是将数据元素序列（初始无序序列）分成两部分，一部分为有序序列，另一部分为无序序列，将无序序列中的每一个元素依次插入到有序序列中。即将待排序数据元素按其关键字的大小插入到有序序列的适当位置上。

插入排序的比较次数取决于各元素的初始排列情况。

直接插入排序过程示例如下：

5，3，7，2，4，6（原始序列）

3，5，7，2，4，6（将无序序列 3，7，2，4，6 中的 3 插入到有序序列 5 前）

3，5，7，2，4，6（将无序序列 7，2，4，6 中的 7 插入到有序序列 3，5 后）

2，3，5，7，4，6（将无序序列 2，4，6 中的 2 插入到有序序列 3，5，7 最前面）

2，3，4，5，7，6（将无序序列 4，6 中的 4 插入到有序序列 2，3，5，7 的 3 后）

2，3，4，5，6，7（将 6 插入到 5 后，排序完成）

2. 交换排序

交换排序是针对排序数据元素，两两比较关键字，若发现存在逆序（两个关键字按非递增次序排列），则交换这两个数据元素，一直到待排序数据元素序列中没有逆序为止。常用的排序方法有冒泡排序和快速排序。本单元项目采用的是冒泡排序。

冒泡排序的思想是：设 n 为序列的元素个数，用相邻的两两比较方法第一次在 n 个数中冒出最小数，第二次在 n-1 个数中冒出最小数（序列的第二小数），第三次在 n-2 个数中冒出最小数（序列的第三小数）……共冒 n-1 次。

冒泡排序过程示例如下：

5，3，7，2，4，6（原始序列）

2，5，3，7，4，6（从右往左两两比较，原始序列最小元素 2 冒泡）

2，3，5，4，7，6（3，5，4，7，6 序列最小元素 3 冒泡）

2，3，4，5，6，7（5，4，7，6 序列最小元素 4 冒泡）

2，3，4，5，6，7（5，6，7 序列最小元素 5 冒泡）

2，3，4，5，6，7（6，7 序列最小元素 6 冒泡）

2，3，4，5，6，7（排序完成）

3. 选择排序

选择排序是将数据元素序列分成有序序列和无序序列两部分，初始有序序列为零。每

趟排序都从无序序列中选取出关键字最小的数据元素放在有序序列的最后，直到全部数据元素排序完毕。常用的选择排序方法有直接选择排序和堆排序。

直接选择排序的思想是：每一趟 (如第 i 趟 , i = 0, 1, …, n–2) 在后面 n–i 个待排序列中选出最小数 , 与原第 i 位置上的元素进行交换，作为有序序列中的第 i 个元素。待到第 n–2 趟完成，待排序序列只剩下 1 个数时 , 就不用再选了。

直接选择排序结点的移动次数比较少，结点的移动次数与结点序列的初始排列有关。

直接选择排序过程示例如下：

5，3，7，2，4，6（原始序列）

2，3，7，5，4，6（将最小元素 2 作为有序序列的第一个元素，与 5 交换）

2，3，7，5，4，6（将 3，7，5，4，6 序列的最小元素 3 放到有序序列后，位置不变）

2，3，4，5，7，6（将 7，5，4，6 序列的最小元素 4 与 7 交换）

2，3，4，5，7，6（将 5，7，6 序列的最小元素 5 放到有序序列后，位置不变）

2，3，4，5，6，7（将 7，6 序列的最小元素 6 与 7 交换，完成排序）

稳定性

待排序列中相同关键字排序后相对位置是否发生变化称为排序的稳定性。相对位置不发生变化的排序称为稳定的排序，相反，相对位置发生变化的排序称为不稳定的排序。例如，四位学生获奖等级情况：

学号	1	2	3	4
等级	2	3	2	1

按等级关键字排序后：

学号	4	3	1	2
等级	1	2	2	3

有两个 2 等奖，排序后 3 号的 2 等奖排到了 1 号的 2 等奖前面了，即相同关键字相对位置发生了变化，因此，称此排序是不稳定的排序。

排序方法的稳定性是衡量排序方法的一个标准。

排序算法与数据存储结构的关系

排序的数据可以用数组或链表存储，是用数组存储好还是用链表存储好呢？这取决于数据本身，若排序数据的数量增减频繁的话，采用链表存储排序数据比较好，否则还是用数组存储较好。数组上排序算法的实现比链表上的容易些，另外链表所需的存储空间也大些，有指针开销。

项目九

实现查找指定商品

——查找算法的应用及数据结构的选择

通常网络购物平台中出售的商品非常多，要在众多商品信息中查找某一商品，若依次翻页比对，可能要花相当长的时间。因此，平台一般都会提供查找功能，如图 5-6 所示。程序员在编程实现查找功能时，要依靠好的算法和合适的数据结构以提高查找效率。

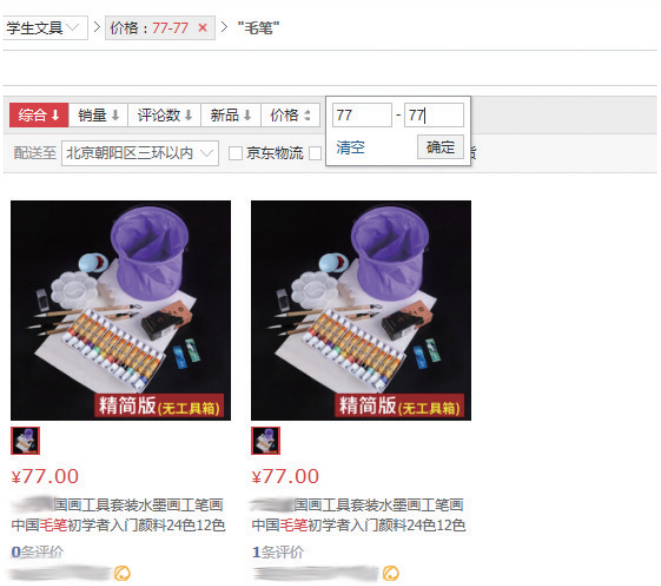


图 5-6　查找价格为 77 元的商品结果

项目学习目标

在本项目中，我们将一起通过探索各种查找方法，了解迭代和递归，选择一些合适的数据结构来加快查找速度，从而进一步理解算法与数据结构之间的关系。

完成本项目学习，须回答以下问题：

1. 查找的算法有哪些？

2. 用迭代法如何实现有序商品序列的二分查找？用递归法如何实现有序商品序列的二分查找？两者有何区别？

4. 有序或无序商品序列可以分别使用什么算法进行查找？

5. 查找的速度与数据结构的选择是否相关？为什么？

项目学习指引

1. 采用顺序查找法查找商品

假设某店铺出售的某品种商品有 11 种不同的价格：15,83,19,88,37,96,64,5,80,18,92，要在其中查找价格为 64 的商品，最简单、最基本的方法是将商品价格序列存放在数组中，将待查元素 64 依次与数组中的每个元素一一比较，这种方法称为顺序查找法，如图 5-7 所示。

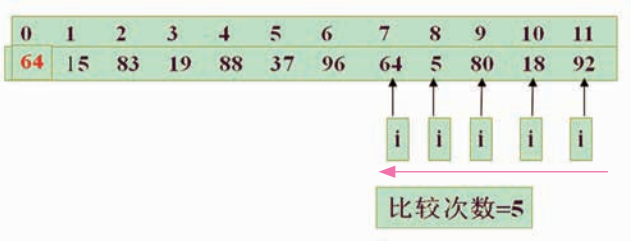


图 5-7　顺序查找过程示意

核心概念

查找（search）是根据给定的某个值，在查找表中找到一个其关键字等于给定值（也称待查关键字）的数据元素。

图中所示的是从数组最后往前依次比较，当 i 为 7 时查找成功，此时共比较了 5 次。为编程方便，将待查元素 64 放在数组的 0 号空间，若序列中没有 64，那 i 会移到 0，查找失败。把 0 号空间称为监视哨，即 i 不可能越过 0 为负数。

参见 P108 知识链接“查找”“顺序查找法”

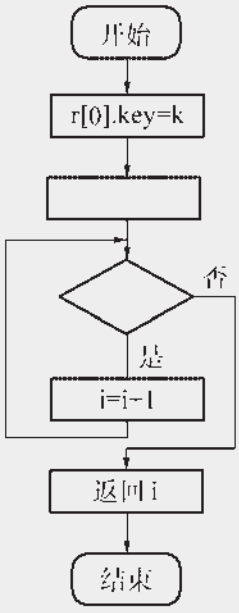
思考与讨论

顺序查找法的特点是什么？

活　动

9.1　在右部的流程框图（局部）中，完成顺序查找算法（假设 n 个数据在数组 r 中，待查关键字 k）；编程实现，并运行测试。

程序：



2. 体验使用二分查找法查找商品

假设该店铺出售的另一品种商品有 11 种价格，按从小到大排序，价格为 5，13，19，21，37，56，64，75，80，88，92。现要查找价格为 21 的商品在商品序列中的位置，可采用以下方法。

（1）使用迭代法。

首先确定待查价格所在区间，先将整个区间一分为二，如图 5-8 所示。

图 5-8　有序的商品价格序列

用 low（序列的下界）指向第一个商品，high（上界）指向最后一个商品，mid（中间位置）指向中间价格对应的商品，待查价格为 21，首先将 21 与 mid 指的中间位置的价格 56 比较，因 21<56，则 21 必然在 56 之前的区间中，那么就在｛5，13，19，21，37｝中用同样的方法查找，如图 5-9 所示。

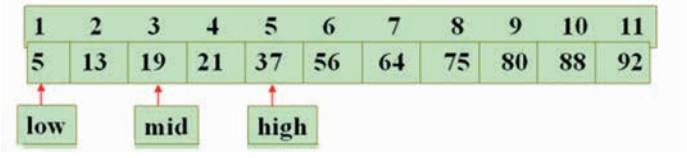


图 5-9　序列按价格折半取前半段

继续将 21 与前半段｛5，13，19，21，37｝的中间元素比较，因 21>19，则在 19 的后半段｛21，37｝中查找，如图 5-10 所示。

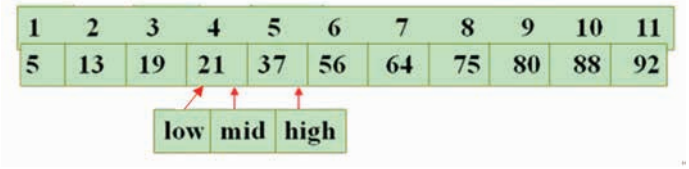


图 5-10　序列再折半

待查元素 21 与中间元素 21 比较相等，即找到了，查找结束，得到 21 在序列中的位置为 4。

核心概念

迭代法：用计算机解决问题的一种基本方法。是一种不断用变量的旧值递推出新值的过程，常用于重复性操作。

参见 P109 知识链接“迭代”

小贴士

这种查找方法称为折半查找，又称二分查找，该算法采用了迭代（iteration）的方法。

二分查找法查找时待查范围缩小得很快，这样查找速度就很快，但要求序列是有序的（从小到大排列，或从大到小排列）。

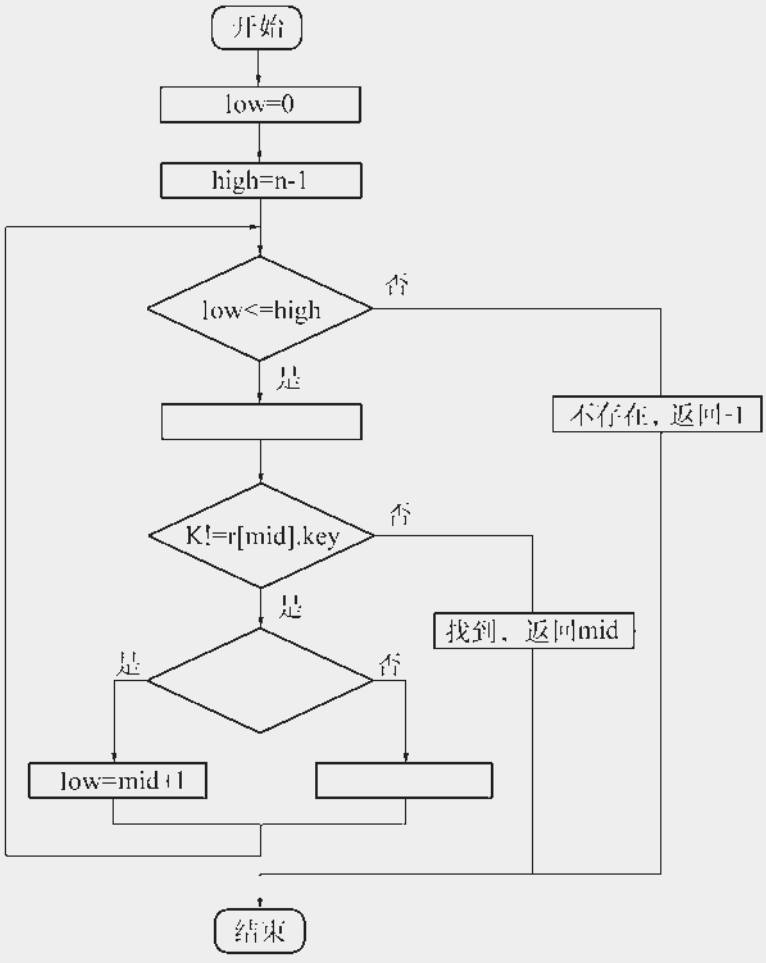
参见 P108 知识链接“二分查找法”

思考与讨论

1. 若序列是无序序列能用这种方法查找吗？为什么？
2. 中间位置如何求得？
3. 二分查找只能在顺序存储结构即数组存储序列下实现，为什么？
4. 迭代还能运用在其他什么场合？

活　动

9.2 在以下流程框图（局部）中完成二分查找算法（迭代法，low 指示查找区间的下界，high 指示查找区间的上界，待查元素 k，n 为元素个数，序列在数组 r 中）；编程实现，并运行测试（可参考配套资源中的“二分查找 .py”程序）。



（2）使用递归法。

二分查找中每次查找的区间是不同的，在不断缩小，但查找方法是一样的，所以也可以用递归（recursion）来实现。

假设有这样一种函数 BinSearch（r, low, high, k），其中 r 代表存储数据元素的数组，low 代表区间最左边的数据元素的数组下标，high 代表区间最右边的数据元素的数组下标。k 代表待查关键字。

r	5	13	19	21	37	56	64	75	80	88	92
（下标）	1	2	3	4	5	6	7	8	9	10	11

图 5-11　存放商品序列的数组 r

针对图 5-11 所示价格有序商品序列，可以使用 BinSearch（r, 1, 11, 21）调用上述函数，因为 21 小于当前区间中点第 6 位置上的价格 56，所以取前半段，即要去调用 BinSearch（r, 1, 5, 21），因为 21 大于当前区间中点第 3 位置上的价格 19，所以取后半段，即要去调用 BinSearch（r, 4, 5, 21）。此时中点第 4 位置上的价格等于待查商品价格 21，查找完成，但函数执行并未完成，位于最上层的 BinSearch（r, 1, 11, 21）并未获得结果。所以要一层一层返回 21 的位置（数组下标）4，直至函数 BinSearch（r, 1, 11, 21）的值为 4 结束，如图 5-12 所示。

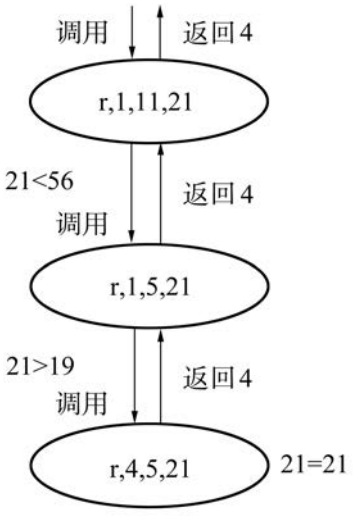


图 5-12　递归调用与返回

思考与讨论

1. 若要查找 80，BinSearch 函数如何调用？
2. 递归还能运用在其他什么场合？

核心概念

递归法：是把问题转化为规模缩小了的同类问题的子问题。然后通过一个过程或函数在其定义中直接或间接调用自身求解的方法。

小贴士

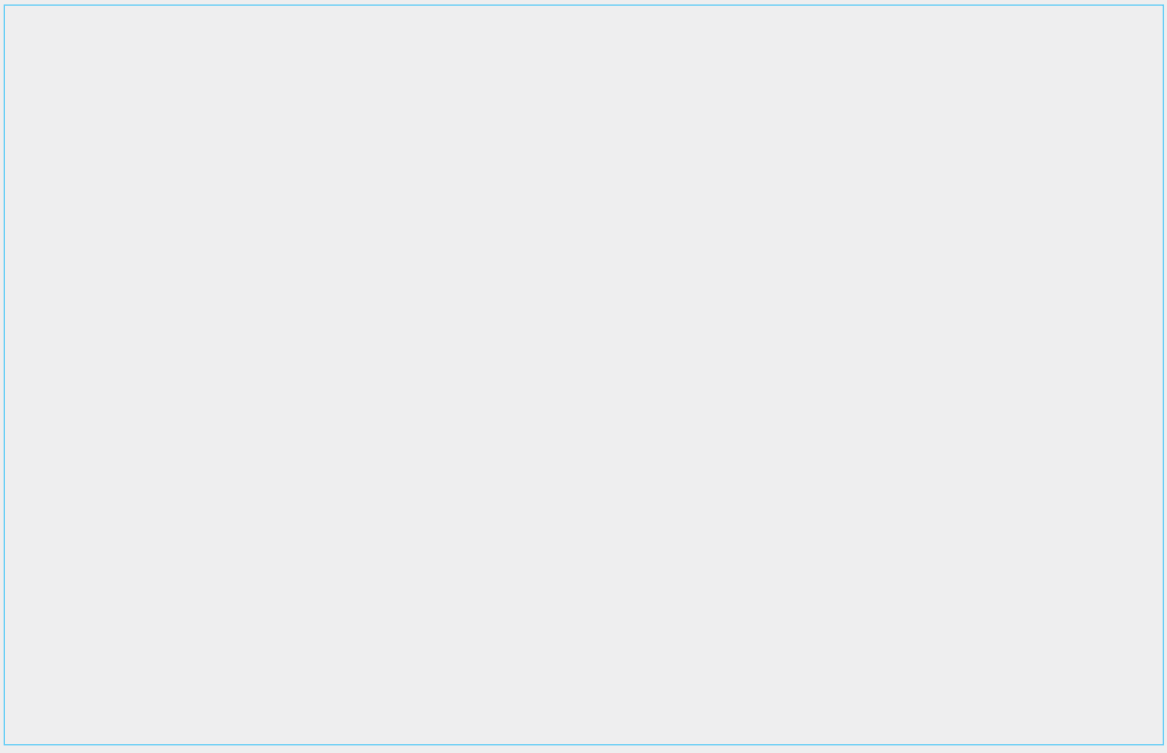
BinSearch 函数通过不断调用自身实现查找，这类函数称为递归函数。

参见 P110 知识链接“递归”

活　动

9.3　画出二分查找的递归算法流程图，（low 指示查找区间的下界，high 指示查找区间的上界，待查元素 k，序列在数组 r 中）；完成程序，编程实现，并运行测试（可参考配套资源中的“二分查找递归.py”程序。

流程图：



程序：

```
def  BinSearch( r, low, high, k):
    if(low<=high):
        mid=_________
        if(k<r[mid].key):
            return BinSearch(_________);
        else:
            if(k>r[mid].key):
                return BinSearch(_________);
            else:
                return  mid
    else:
        return –9999
```

3. 采用索引查找法查找商品

假设要在大量价格无序的商品中，查找价格为 22 的商品。可以通过建立索引表的方式查找，如图 5-13 所示。

参见 P108 知识链接“索引查找法”

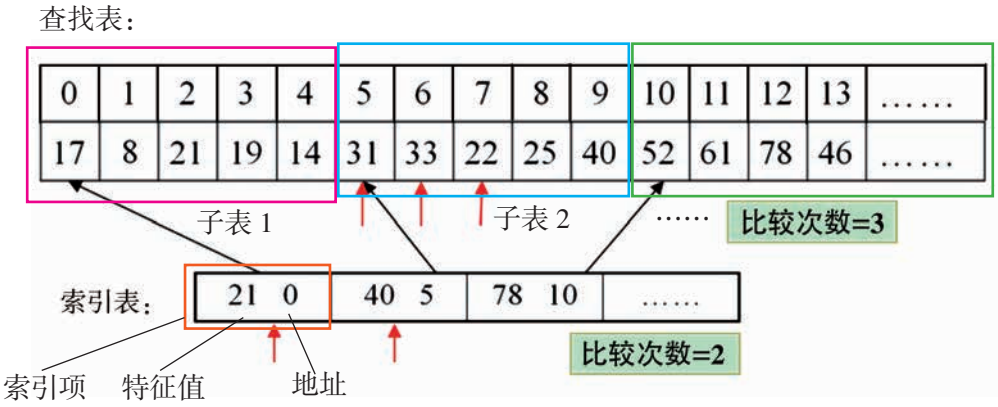


图 5-13　索引表

其中，索引表由索引项组成。索引表中的每一个索引项都包含两项内容，一项是子表的特征值（索引值），另一项是地址（子表在主表中的开始位置）。例如，索引项（21　0）表示小于等于 21 的序列（子表 1）在数据表中的开始位置是 0，索引项（40　5）表示小于等于 40 大于 21 的序列（子表 2）在查找表中的开始位置是 5，以此类推。查找 22 时，先在索引表中查找到索引项（40　5），再在查找表中的子表 2 中从第一个位置开始查找，直到找到 22。查找时，总共比较 5 次。

小贴士

子表（也称块）一般根据特征值划分。图中子表的划分规则是，特征值所在的子表里的所有数据都小于下一个子表里的所有数据。特征值是该子表里的最大数。

思考与讨论

1. 上述第二个索引项的特征值是否可以选 33？为什么？如果选 33，那么子表 2 该如何划分？
2. 索引查找法对主表有什么要求？

活　动

9.4　对于以下价格序列的商品，若采用索引查找法，请为其建立索引表，查找 38 并计算比较次数，谈谈索引查找的好处。

1,6,8,9,7,27,13,15,64,55,44,42,38,78,90,66

索引表：

4. 分析查找算法与数据结构的关系

使用不同查找法查找一个数据元素的比较次数是不同的，我们可以用平均查找次数来衡量查找的速度。例如，顺序查找法的平均查找次数是（1+2+……+n）/n，其中，n 代表元素个数，括号内的每一项代表查找第 i 个元素的比较次数，利用求和公式可以得平均查找次数为（1+n）/2，可见该平均查找次数与查找序列元素的个数成正比。而采用索引存储结构的索引查找法查找次数与主表的数据元素的个数几乎是无关的。因此，数据量大的情况下，采用索引查找法查找的算法效率会高很多。但是采用索引存储结构来存储须附加存储索引表，因此会占用更多的存储空间。

小贴士

一个算法是由控制结构（顺序、分支和循环）和固有数据类型的操作构成的。算法执行时间取决于两者的综合效果，是衡量算法效率的一个方面。算法执行时间与基本操作次数成正比。

思考与讨论

1. 若使用二分查找法，查找每个数据元素的比较次数分别是多少？

2. 二分查找法和顺序查找法的查找速度哪个快？为什么？

活　动

9.5 查阅资料，根据所学，归纳表中各查找算法分别适合何种存储结构，并给出理由，谈谈数据结构和算法之间的关系。

算法	顺序存储结构	链式存储结构	索引存储结构
二分查找			
顺序查找			
索引查找			

知识链接

查找

查找是根据给定的某个值，在查找表中找到一个其关键字等于给定值（也称待查关键字）的数据元素。

查找表是由同一类型的数据元素（或记录）构成的集合，是由所要查找的元素区间构成的表。查找表可以是线性表，也可以是树表（二叉查找树），如商品信息表就是一种线性的查找表。

常用的查找方法有顺序查找法、二分查找法、索引查找法等。

1. 二分查找法

二分查找法是在查找表中每次取中间元素和待查关键字进行比较，若小则在元素区间的前半段中查找；若大则在元素区间的后半段中查找。重复这样的过程，直到找到满足条件的数据元素，即查找成功；或直到子序列不存在为止，即查找不成功。二分查找法的优点是比较次数少，查找速度快，但是要求查找表为有序表，且只能是顺序存储结构，即用数组存储。

例如，在计算机猜数的游戏中，给出一个 1~100 之间的数字，让计算机猜，如果小于该数字，提示计算机“小了”；大于该数字，则提示计算机“大了”。此游戏中，计算机可以采用二分查找法来完成竞猜，即先将 50（100 的一半）作为初始答案，在得到“大了”的信息时，取 1~49 区间的中点作为答案，如果得到的是“小了”的信息，取 51~100 区间的中点作为答案，若仍未猜中，则重复上述过程，不断以二分法缩小区间范围直至猜出正确答案。

2. 顺序查找法

顺序查找法是从查找表的一端开始依次将表中的数据元素与待查关键字进行逐一比较。顺序查找法既适用于有序表，也适用于无序表。

假定有 n 个记录存放在数组 r[1]，r[2]，...，r[n] 中，其中第 i 个记录的关键字值为 r[i].key。待查关键字为 k，将 k 依次与 r[n].key，r[n−1].key，...，r[1].key 进行比较，一旦某个 r[i].key 等于 k，查找成功，返回下标 i，若所有的数据元素都与 k 值不相等，则给出查找失败（i=0）的信息。

顺序查找法的特点是算法比较简单，但查找一个数据元素的比较次数多，算法效率较低，一旦数据量 n 很大，查找时间就相对较长。

3. 索引查找法

索引查找法是基于索引存储结构的一种查找方法。索引存储结构是主表（查找表）按照一定的条件划分成若干个逻辑上的子表（或称块），并为每个子表分别建立一个索引项，这些索引项的集合构成一个索引表（也存储在计算机中）。其中，子表满足两个条件：（1）子表内的数据元素不要求有序；（2）子表间有序，即某一子表的所有数据元素要大于前一个子表的所有数据元素（第一个子表除外），小于后一个子表的所有数据元素（最后一个子

表除外）。索引项至少包含两项内容：特征值（也称索引值）和地址（也称块首指针）。其中，特征值是子表中最大的数据元素，地址用于指向子表在主表中的开始位置。

使用索引查找法查找的时候，先在索引表中按待查关键字的大小查找索引项，找到后，根据地址找到主表中的子表开始位置，开始查找。简单来说，就是先找子表，再在子表中查找。

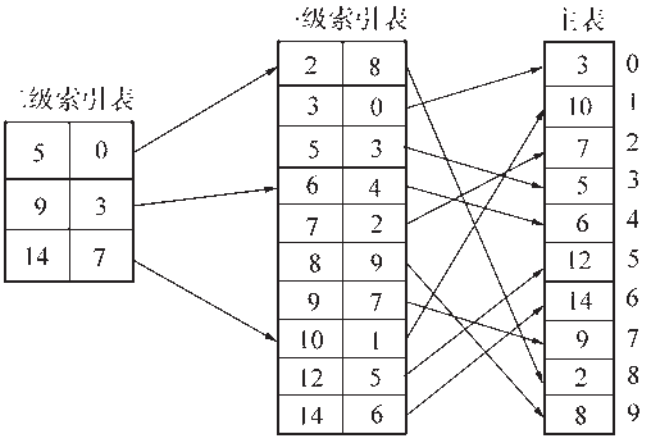


图 5-14　二级索引表

索引表都是有序的，可以顺序查找，也可以二分查找，若索引表数据量大则用二分查找。

若主表无法按条件划分成若干个子表，则可以对每个数据元素建立一个索引项，所建的索引表是有序的，因此可以用二分查找，也可以在索引表上再建一层索引，从而加快查找速度，如图 5-14 所示。

查找时从高级索引表开始，一级一级往下直至找到。

索引技术在不断发展，这里所述的是最基本的方法。

总之，在数据量大的情况下，索引存储结构一旦建立后，采用索引存储可大幅提高查找速度。

迭代与递归

1. 迭代

迭代是重复反馈过程的活动，其目的通常是为了逼近所需目标或结果。每一次对“过程”的重复称为迭代，而每一次迭代得到的结果会作为下一次迭代的初值。

迭代法则是一种不断用变量的旧值递推出新值的过程，适合完成重复性操作。

例如，求解如下问题：一个饲养场引进一只刚出生的新品种兔子，这种兔子从出生的下一个月开始，每月新生一只兔子，新生的兔子也以此规律繁殖。如果所有的兔子都不死去，问到第 12 个月时，该饲养场共有兔子多少只？

这是一个典型的递推问题。不妨假设第 1 个月时兔子的只数为 R_1，第 2 个月时兔子的只数为 R_2，第 3 个月时兔子的只数为 R_3……根据题意，“这种兔子从出生的下一个月开始，每月新生一只兔子”，则有

$R_1=1$，$R_2=R_1+R_1\times 1=2$，$R_3=R_2+R_2\times 1=4$ ……

根据这个规律，可以归纳出下面的递推公式：

$R_n=(R_{n-1})\times 2\ (n\geqslant 2)$

对应 R_n 和 R_{n-1}，可定义两个迭代变量 y 和 x，将上面的递推公式转换成如下迭代关系：

$y=x\times 2$

$x=y$

让计算机重复执行这个迭代关系 11 次，就可以算出第 12 个月时的兔子数。可以看出，计算机每执行一次，新得到的 y 和 x 都作为下一次迭代的初值。

2. 递归

递归是把问题转化为规模缩小了的同类问题的子问题，然后通过一个过程或函数在其定义中直接或间接调用自身求解的方法。一个直接调用自己或通过一系列的调用语句间接地调用自己的函数称为递归函数。

例如，阶乘函数 f(n)=n*f(n−1)，从函数看出，要求出 n 阶乘就要先求出 (n−1) 阶乘，要求出 n−1 阶乘就要先求出 (n−2) 阶乘……以此类推，直至 n=0 的阶乘为 1，这个过程称为递推；然后从 0 的阶乘可得 1 的阶乘，从 1 的阶乘可得 2 的阶乘……以此类推，直至得到 n 的阶乘，这个过程称为返回（也称回归）。所有的递归算法的执行过程都分成递推和回归这样两个阶段，f(n) 的运行轨迹如图 5−15 所示。

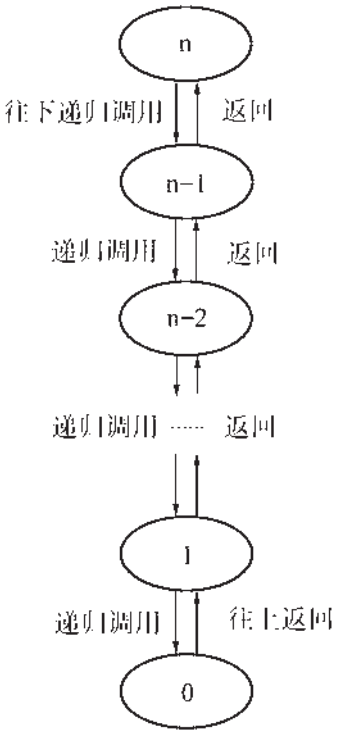


图 5−15　递归状态图

图中椭圆圈表示调用函数，椭圆圈中的符号表示当前调用的参数，从状态图看出先调用的函数后返回，而后调用的函数先返回。例如，上述求阶乘的递归函数：

```
def func(n):
    if n == 0:
        return 1
    else:
        return n * func(n−1)
print(func(5))
```

假设要求 5 的阶乘，则程序从调用 func(5) 开始运行，遇到调用自身函数的语句 return n * func(n−1)，计算 5* func(4)，此时要调用 func(4)，以此类推，一直到调用 func(0)，遇到语句 return 1（即 func(0)=1），返回到 func(1) 的 return 1 * func(0) 语句，计算 1*func(0) 等于 1，继续返回，以此类推，直至返回到 func(5) 的 return 5* func(4) 语句，计算 5*func(4) 等于 120。然后执行 print 语句，输出结果。

因为递归函数具有先调用的后返回的特点，所以要用先进后出的数据结构栈来实现。

可以看出，递归只需少量的程序语句就可以描述出解题过程所需要的多次重复计算。迭代和递归在人工智能领域也被大量应用。

拓展阅读

算法度量及分析

对于一个问题可以有多种算法，那么如何来衡量哪种算法最有效？或者优于目前已知的算法呢？人们一般从两个方面来衡量。一个是时间效率，即算法处理数据时所花费的时间，用时间复杂度来表示；一个是空间效率，即算法所需求的存储量的大小，用空间复杂度来表示。但二者往往有冲突，不能同时兼顾，一般取时间效率，时间效率被认为更重要一些。

1. 时间复杂度分析

对于解决同一个问题的算法，执行时间短的显然比执行时间长的时间效率高，即执行时间短的算法比执行时间长的算法时间复杂度要低。那么算法执行时间的长短如何度量呢？一种方法是编制一个程序实现这个算法，然后输入不同的数据运行这个程序，测定该程序运行的时间，这称为事后统计法。这种方法的缺陷非常明显：一是必须编制程序和运行程序，非常耗费时间，也比较麻烦；二是受到的约束条件比较多，比如运行程序的计算机软硬件条件、使用的编程语言等，有时这些会掩盖算法本身的优劣。

另一种方法是分析算法运行的时间，称为事前分析法。它不用上机运行依算法编制的程序，而是分析影响算法执行时间的各种因素，从而估算出算法执行的时间。其中，一个最重要的因素是输入算法的数据量（称为问题规模）。例如，一个查找单词的算法。在 10 个单词中查找某个单词与在 10 万个单词中查找某个单词所花费的时间不可同日而语。因此，一个算法的执行时间 T 可被表示为问题规模 n 的一个函数 T(n)。

除了问题规模以外，实现算法的程序设计语言、源程序编译后产生的机器代码的质量、机器执行指令的速度等都会影响算法的执行时间。因此，不可能将 T(n) 表达为算法实际执行的时间。一般用算法中语句被执行的次数来表示算法的时间效率（算法的时间复杂度）。

当算法的时间复杂度为常量时，表示它不随问题规模 n 的大小而改变，记为 T(n)= 0(1)。当算法的时间复杂度与问题规模 n 成线性关系时，则记为 T(n) = 0(n)。

2. 空间复杂度分析

算法的空间复杂度就是算法或程序运行时所占用的存储空间。一般只统计数据部分所占用的存储空间，而不统计代码部分所占的存储空间。

——摘自清华大学出版社《数据结构实例教程》（第 2 版）

单元挑战　使用二叉查找树查找学生成绩信息

一、项目任务

二叉查找树（也称二叉排序树）：对于树中的每个结点 k，若 k 的左子树不空，则左子树上所有结点的值均小于 k 结点的值；若 k 的右子树不空，则右子树上所有结点的值均大于 k 结点的值。

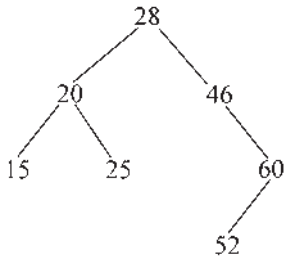


图 5-16　二叉树

二叉查找树是适用于查找的二叉树，查找方法是：

（1）若待查值等于根结点的关键字，则查找成功；

（2）若待查值小于根结点的关键字，则继续在左子树上进行查找；

（3）若待查值大于根结点的关键字，则继续在右子树上进行查找。

查找总是从树根开始，比如在图 5-16 所示二叉树中查找关键字 25，先与树根 28 比较，25<28，往左走继续与 20 比较，因 25>20，往右走与 25 比较，此时两数相等，查找结束。

请以二叉查找树为索引对学生成绩进行查找，二叉树存储成绩，单链表存储学生信息，数据结构形式为：

树结点结构为：（分数，左孩子指针，右孩子指针，本分数学生链首指针）

表结构形式为：（学号，姓名，指向下一结点指针）

二、项目指引

1. 根据本小组同学的成绩，画出二叉查找树的结构图（参考图 5-17）。

2. 设计二叉查找树根据成绩查找学生信息的算法流程。

*3. 编程实现。

*4. 讨论使用二叉查找树的优缺点。

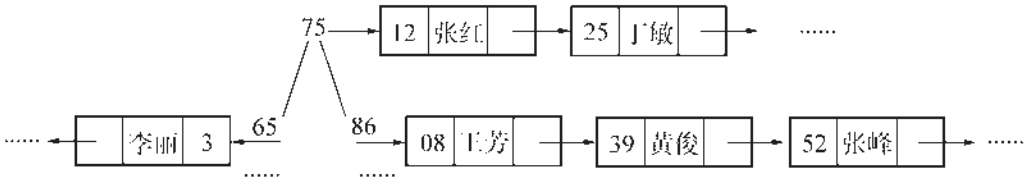


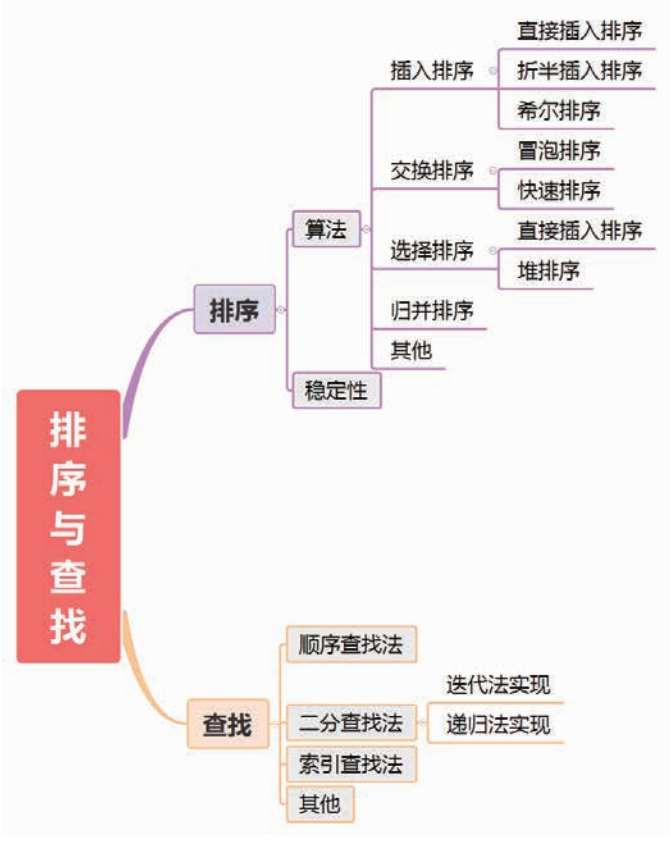
图 5-17　示例

三、交流评价与反思

以自己熟悉的信息表达工具（如演示文稿等）制作电子作品，通过网络或课堂展示交流自己的算法和程序，并对他人的作品进行评价。

单元小结

一、主要内容梳理



二、单元练习

1. 请采用不同的排序方法对序列（Q，H，C，Y，P，A，M，S，R，D，F，X）中的数据元素按字母序的升序进行排序，编程并上机调试，说说你采用的数据结构和理由。

2. 已知某企业生产部门员工的工龄记录为（5，3，10，8，7，14，15，12，18，16，25，17）（按工号的顺序排列），现要找出工龄为 25 年的员工。

（1）使用顺序查找法完成。

（2）使用索引查找法完成。

（3）对记录进行排序后用二分查找法完成。

3. 假设有 3 个分别命名为 A、B 和 C 的柱子，在柱 A 上插有 n 个直径大小各不相同，从上到下，以从小到大顺序排列的编号分别为 1，2，…，n 的圆盘，（如图 5-18A，图中 n=4）。现要求将 A 柱上的 n 个圆盘移至 C 柱上并仍按同样的顺序叠排，且圆盘移动时必须遵循下列规则：

（1）每次只能移动一个圆盘。

（2）圆盘可以插在 A、B 和 C 中的任一柱上。

（3）任何时刻都不能将一个大的圆盘压在小的圆盘之上。

可参照以下解法：

① 用 C 柱做过渡，将 A 柱上的 (n−1) 个盘子移到 B 柱上；

② 将 A 柱上最后一个盘子直接移到 C 柱上；

③ 用 A 柱做过渡，将 B 柱上的 (n−1) 个盘子移到 C 柱上。

请使用递归法编写程序。

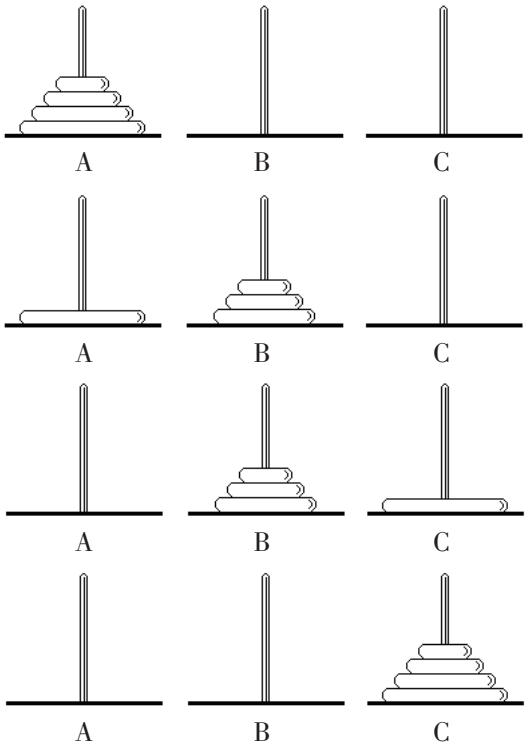


图 5-18　移动步骤示意

三、单元评价

评 价 内 容	达 成 情 况
知道商品排序可以采用不同的算法和数据结构（A、T）	
能说出计算机常用排序算法（T）	
能理解直接插入排序、冒泡排序、直接选择排序的思想和特点；能选择合适的算法对商品销量进行排序（T）	
知道商品查找可以采用不同的算法和数据结构；能简单描述数据结构和算法的关系（T）	
能理解顺序查找的思想和特点（T）	
能理解二分查找法的思想和特点，能使用迭代和递归的方法，完成二分查找法（I、T）	
能描述索引查找的思想和特点（T）	
能初步了解算法的时间复杂度和空间复杂度（T）	

说明：A—信息意识，T—计算思维，I—数字化学习与创新，R—信息社会责任

附 录

部分名词术语中英文对照

（以汉字拼音字母次序为序）

遍历　traverse
插入　insert
查找　search
抽象数据类型　abstract data type，ADT
出栈　pop
串　string
递归　recursion
迭代　iteration
队列　queue
队头　front
队尾　rear
二叉树　binary tree
后进先出　last in first out，LIFO
进栈　push
排序　sort
删除　delete
树　tree
数据　data
数据对象　data object
数据结构　data structure
数据类型　data type
数据项　data item
数据元素　data element
数组　array
先进先出　first in first out，FIFO
线性表　linear list
栈　stack
栈顶　top

PUTONG GAOZHONG JIAOKESHU
XINXIJISHU

普通高中教科书
信息技术　选择性必修1
数据与数据结构
上海科技教育出版社有限公司出版发行
（上海市闵行区号景路159弄A座8楼　邮政编码201101）
湖南省新华书店经销　湖南长沙鸿发印务实业有限公司印刷
开本890×1240　1/16　印张7.5
2021年8月第1版　2021年12月第2次印刷
ISBN 978-7-5428-7469-6/G·4468
定价：9.54元
批准文号：湘发改价费〔2017〕343号　举报电话：12315

此书如有印、装质量问题，请向印厂调换
印厂地址：长沙黄花印刷工业园三号　电话：0731-82755298