高等职业院校规划教材·软件技术系列

计算机软件技术基础

杨　平　主编
褚建立　主审

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

高等职业院校规划教材·软件技术系列

计算机软件技术基础

主编　杨　平

主审　褚建立

中国铁道出版社

CHINA RAILWAY PUBLISHING HOUSE

内 容 简 介

本书按照教育部提出的计算机基础课程三层次教学体系中的软件技术基础课程的要求，根据高职高专类学生的特点编写，实例贯穿其中，与现实生活相结合。全书共有数据结构、操作系统、软件工程三篇，分为18章。数据结构的主要内容包括算法、线性表、栈、队列、树、图、查找和排序；操作系统的主要内容包括操作系统引论、进程管理、处理机调度与死锁、存储器管理、设备管理及文件管理；软件工程的主要内容包括软件工程概述、传统软件工程设计、面向对象的软件工程及软件工程项目管理；最后附有软件项目开发计划文档供读者参考。本书在内容组织上由浅入深，循序渐进，语言通俗流畅，实例选用得当，与现实生活联系紧密，有利于读者理解和掌握。每章开头附有基本要求和重点难点，最后附有小结并配有相应的习题。

本书结构合理，内容丰富，通俗易懂，实用性强，适合作为高职高专院校的教材，也可作为计算机二级和三级等级考试的参考书。

图书在版编目（CIP）数据

计算机软件技术基础 / 杨平主编. —北京：中国铁道出版社，2009.4
（高等职业院校规划教材. 软件技术系列）
ISBN 978-7-113-09935-0

Ⅰ.计… Ⅱ.杨 Ⅲ.软件—高等学校：技术学校—教材 Ⅳ.TP31

中国版本图书馆CIP数据核字（2009）第057128号

书　　名：计算机软件技术基础
作　　者：杨　平　主编

策划编辑： 翟玉峰　沈　洁
责任编辑： 翟玉峰　　**编辑部电话：**（010）63583215
编辑助理： 张国成
封面设计： 付　巍　　**封面制作：** 白　雪
责任印制： 李　佳

出版发行： 中国铁道出版社（北京市宣武区右安门西街8号　　邮政编码：100054）
印　　刷： 河北省遵化市胶印厂
版　　次： 2009年5月第1版　2009年5月第1次印刷
开　　本： 787mm×1092mm　1/16　**印张：** 16　**字数：** 394千
印　　数： 4 000册
书　　号： ISBN 978-7-113-09935-0/TP・3228
定　　价： 25.00元

FOREWORD

前言

计算机应用技术的飞速发展，正在逐步揭开计算机神秘的面纱，现在已把它从高等学府和研究院的专业实验室中解放出来，成为人们工作、学习和生活中不可缺少的工具。时至今日，使用计算机进行文字处理、网上通信、休闲娱乐等已经成为一种时尚。然而，对于计算机工作原理不了解，必然大大束缚使用者的手脚，尤其对于那些希望使用计算机解决某些特殊问题或者希望尝试使用某种新方法去解决某些问题的用户更是如此。因此，了解计算机的工作原理，学习程序设计的基本方法十分重要。

计算机科学研究的重点是信息在计算机中的表示和问题处理方法。这些问题出现在许多不同的研究层次和不同的应用领域。从程序设计的观点看，信息在计算机中的表示就是“数据结构”研究的问题；信息在计算机中的处理就是“算法”研究的问题。因此，学习算法和数据结构的基本知识是了解计算机工作基本原理、掌握程序设计基本技术的必经之路。

用计算机解决实际问题，就是在计算机中建立一个解决问题的模型。在这个模型中，计算机的内部数据表示了需要被处理的实际对象，包括其内在的性质和关系；处理这些数据的程序则模拟对象领域中的求解过程；通过解释计算机程序的运行结果，便得到了实际问题的解。

计算机软件技术基础是一门计算机专业基础课，是在学生了解计算机基础知识的基础上，为提高学生对软件本质的理解开设的一门课程，通过对计算机软件基础领域的基本原理、方法和思想的学习，来提高学生对软件开发工具的使用能力和对环境的适应能力；学习者除了要掌握现有计算机软件的使用方法外，还必须掌握软件设计与开发的基本知识和有关技术，如数据的组织、程序的组织、计算机资源的利用、数据处理技术等，以便得心应手地进行应用软件的设计与开发。

本书针对高职高专学生的特点、结合作者的实际教学经验，在知识上力求全面，在写法上力求精练、简明。全书共有三篇，分18章，每章后面都附有一定数量的习题。

第一篇为数据结构，主要介绍了算法、线性表、栈、队列、树、图以及查找、排序等，其中的算法是基于C语言实现的。

第二篇为操作系统，概括地介绍了操作系统的基本概念、地位、作用以及操作系统的五大管理功能（进程管理、处理机调度与死锁、存储器管理、设备管理、文件管理）。

第三篇为软件工程，主要包括软件工程概述、传统软件工程设计、面向对象的软件工程、软件工程项目管理等内容。

本书配有电子教案、教学计划与课程标准，可到网站 http://edu.tqbooks.net 下载。

本书由邢台职业技术学院杨平担任主编，褚建立担任主审。其中，杨平编写第3、5章并负责全书统稿，褚建立编写第2、4章并负责全书审定，吴丽丽编写第6章并负责全书订正、修改，第8章由赵美枝编写，第9、10、11、12、14、15章由顾爱琴编写，第13章由顾爱琴、丁莉编写，第16、18章由曾凡晋编写，第17章由王月青编写，第1章由霍艳玲编写，第7章由王党利编写。

本书结构合理，层次清晰，讲解详细透彻，适合作为高职高专院校的教材，也可作为计算机二级和三级等级考试的参考书。

由于编者水平有限，加之时间仓促，书中难免有疏漏之处，欢迎大家批评指正。

编 者

2008年10月

CONTENTS

目 录

第一篇 数 据 结 构

第二篇 操 作 系 统

第三篇 软 件 工 程

第一篇　数据结构

数据结构是计算机软件技术中最重要的一部分。计算机科学研究的重点是信息在计算机中的表示和问题的处理方法。这些问题出现在许多不同的研究层次和不同的应用领域。从程序设计的观点看，信息在计算机中的表示就是“数据结构”研究的问题；信息在计算机中的处理问题就是“算法”研究的问题。因此，学习算法和数据结构的基本知识是了解计算机工作基本原理、掌握程序设计基本技术的必经之路。

本篇以C语言作为算法的描述工具，系统介绍了算法和数据结构的基本概念、线性数据结构（线性表、栈、队列）、非线性数据结构（树、图）、数据查找和排序技术等。

数据结构是一门实践性较强的软件基础课程，为了学好这门课程，必须在掌握理论知识的同时，加强上机实践，达到理论与实际应用相结合的目的。通过本篇的学习，读者可以掌握计算机中数据的组织方式和基本算法思想，结合实际应用加深对基本概念的理解，提高在实际中运用数据结构知识的能力。数据结构学习的效果不仅关系到后续课程的学习，而且直接关系到软件设计水平的提高和专业素质的培养，因而成为提高软件水平的关键性课程。

第1章 算 法

基本要求：

- 理解数据结构的基本概念和术语
- 了解算法的概念和基本特征
- 理解算法的设计方法
- 掌握评价算法好坏的方法

教学重点和难点：

- 数据结构的概念
- 数据结构术语
- 算法的概念
- 算法的分析

众所周知，早期电子计算机的应用范围仅局限于科学计算，其处理的对象是纯数值性的信息，通常人们把这类问题称为数值计算。近30年来，电子计算机的发展异常迅猛，它不仅表现在计算机本身运算速度不断提高，信息存储量日益扩大，价格逐步下降，更重要的是计算机广泛应用于情报检索、企业管理、系统工程等方面，已远远超出了数值计算的范围，而渗透到人类社会活动的一切领域。相应地，计算机的处理对象也从简单的纯数值性信息发展到非数值性的和具有一定结构的信息。现代计算机科学的观点，是把计算机程序处理的一切数值的、非数值的信息，乃至程序统称为数据（data），而电子计算机则是加工处理数据（信息）的工具。

处理对象的转变导致系统程序和应用程序的规模越来越大，结构也相当复杂，单凭程序设计人员的经验和技巧已难以设计出效率高、可靠性强的程序，数据的表示方法和组织形式已成为影响数据处理效率的关键。因此，就要求人们对计算机程序所加工的对象进行系统的研究，即研究数据的特性以及数据之间存在的关系——数据结构（data structure）。

数据结构是随着电子计算机的产生和发展而发展起来的一门较新的计算机学科。1968年，美国唐·欧·克努特教授开创了数据结构的最初体系，他所著的《计算机程序设计技巧》第一卷《基本算法》是第一本较系统阐述数据逻辑结构和存储结构及其操作的著作。从20世纪60年代末到70年代初，出现了大型程序，软件也相对独立，结构程序设计成为程序设计方法学的主要内容，人们越来越重视数据结构，认为程序设计的实质是对确定的问题选择一种好的结构，再设计一种好的算法。从70年代中期到80年代初，各种版本的数据结构著作相继出现。

“数据结构”在计算机科学中是一门综合性的专业基础课。数据结构的研究不仅涉及计算机硬件（特别是编码理论、存储装置和存取方法等）的研究范围，而且和计算机软件的研究有着密切的关系，无论是编译程序还是操作系统，都涉及数据元素在存储器中的分配问题。在研究信息检索时也必须考虑如何组织数据，以便查找和存取数据元素。

值得注意的是，数据结构的发展并未终结，一方面，面向各专门领域中特殊问题的数据结构得到研究和发展，如多维图形数据结构等；另一方面，从抽象数据类型的观点来讨论数据结构，已成为一种新的趋势，而且越来越被人们所重视。由此可见，数据结构技术的产生时间并不长，它正处于迅速发展阶段。同时，随着电子计算机的发展和更新，新的数据结构将会不断出现。

1.1 数据结构的概念

什么是数据结构？这是一个难于直接回答的问题。一般来说，用计算机解决一个具体问题时，大致需要经过下列几个步骤：首先要从具体问题中抽象出一个适当的数学模型，然后设计一个解此数学模型的算法（algorithm），最后编出程序、进行测试、调整直至得到最终解答。寻求数学模型的实质是分析问题，从中提取操作的对象，并找出这些操作对象之间含有的关系，然后用数学的语言加以描述。为了说明这个问题，下面首先看一个例子，然后再给出明确的含义。

假定为某个学校的全体学生建立一个通讯录，其中记录全体学生的姓名和相应的住址，现在要写一个算法，要求当给定任何一个学生的姓名时，该算法能够查出该学生的住址。这样一个算法的设计，将完全依赖于通讯录中的学生姓名及相应的住址是如何架构的，以及计算机是怎样存储通讯录中的信息。

如果通讯录中的学生姓名是随意排列的，其次序没有任何规律，那么当给定一个姓名时，则只能对通讯录从头开始逐个与给定的姓名进行比较，顺序查找，直至找到所给定的姓名为止。这种方法相当费时，而且效率很低。

然而，若对学生通讯录进行适当的组织，按学生所在班级来排列，并且再建立一个索引表，这个表用来登记每个班级学生姓名在通讯录中起始处的位置。这样一来，情况将大为改善。这时，当要查找某学生的住址时，则可先从索引表中查到该学生所在班级的学生姓名是从何处开始，而后就从此处开始查找，而不必去查看其他部分的姓名。由于采用了新的结构，于是，就可写出一个完全不相同的算法。

上述的学生通讯录就是一个数据结构问题。可以看到，计算机算法与数据的结构密切相关，算法无不依附于具体的数据结构，数据结构直接关系到算法的选择和效率。

下面再对学生通讯录作进一步讨论。我们知道，当有新学生进校时，通讯录需要添加新学生的姓名和相应的住址；在学生毕业离校时，应从通讯录中删除毕业学生的姓名和住址。这就要求在已安排好的结构中进行插入和删除。对于一种具体的结构，如何实现插入和删除？是把要添加的学生姓名和住址插入到前面、还是末尾、或是中间某个合适的位置上？插入后，对原有的数据是否有影响？有什么样的影响？删除某学生的姓名和住址后，其他的数据（学生的姓名和住址）是否要移动？若需要移动，则应如何移动？这一系列的问题说明，为增加和减少数据，还必须对数据结构定义一些运算。上面只涉及两种运算，即插入和删除运算。当然，还会提出一些其他可能的运算，如学生搬家后，住址变了，为适应这种需要，就应该定义、修改、运算等。

这些运算显然要由计算机来完成，这就要设计相应的插入、删除和修改的算法。也就是说，数据结构还须要给出每种结构类型所定义的各种运算的算法。

通过以上讨论，我们可以直观地认为：数据结构是研究程序设计中计算机操作的对象以及它们之间的关系和运算的一门学科。

1.2 数据结构的基本概念和术语

1. 数据

数据（data）是信息的载体，它能够被计算机识别、存储和加工处理，它是计算机程序加工的“原料”。例如，一个代数方程求解程序中所用的数据是整数和实数，而编译程序中使用的数据是字符串。随着计算机软、硬件的发展，计算机应用领域的扩大，数据的含义也随之拓广了。例如，计算机处理的图像、声音等，它们也属于数据的范畴。

2. 数据元素

数据元素（data element）是数据的基本单位。有些情况下，数据元素也称为元素、结点、顶点、记录。有时一个数据元素可以由若干个数据项（也可称为字段、域）组成，数据项是具有独立含义的最小标识单位。

3. 数据类型

数据类型（data type）是具有相同性质的计算机数据的集合及在这个数据集合上的一组操作。如整数，它是[-maxint，maxint]区间上的整数（maxint 是依赖于所使用的计算机及语言的最大整数），在这个整数集上可以进行加、减、乘、整除、取模等操作。

数据类型可以分为原子数据类型和结构数据类型。原子数据类型是由计算机语言提供的，如 C 语言的整型、字符型；结构数据类型是借用计算机语言提供的一种描述数据元素之间逻辑关系的机制，由用户自己定义的，如 C 语言的数组、结构体类型等。

4. 数据结构

简单地说，数据结构（data structure）是指相互有关联的数据元素的集合。例如，矩阵就是数据结构，在这个数据结构中，数据元素之间有着位置上的关系。又如，图书馆中的图书卡片目录则是一个较为复杂的数据结构，对于列在各卡片上的各种书之间，可能在主题、作者等问题上相互关联，甚至一本书本身也有不同的相关成分。

数据结构也就是数据的组织形式，它一般包括三个方面的内容。

① 数据的逻辑结构。数据的逻辑结构是从逻辑关系上描述数据，它与数据的存储无关，是独立于计算机的，因此数据的逻辑结构可以看做是从具体问题抽象出来的数学模型。

② 数据的存储结构。在实际进行数据处理时，被处理的各数据元素总是被存放在计算机的存储空间中，并且各数据元素在计算机存储空间中的位置关系与它们的逻辑关系不一定是相同的，而且一般也不可能相同。

数据的存储结构是逻辑结构用计算机语言的实现，它是依赖于计算机语言的，对机器语言而言，存储结构是具体的，但我们只在高级语言的层次上来讨论存储结构。

③ 数据的运算。数据的运算就是对数据施加的操作，它定义在数据的逻辑结构上，每种逻辑结构都有一个运算的集合。例如，最常用的运算有检索、插入、删除、更新、排序等。这些运算实际上只是在抽象的数据上所施加的一系列抽象的操作。所谓抽象的操作，是指我们只知道这些操作是“做什么”，而无须考虑“如何做”。只有确定了存储结构之后，才考虑如何具体实现这些运算。

例如，学生通讯录，如表 1-1 所示。

表 1-1　学生通讯录

学　号	姓　名	性　别	年　龄	联系方式	家庭住址
0203001	张　三	女	17	0319-2278888	河北邢台
0203002	李四	男	18	0315-3247651	河北邯郸
0203003	王五	女	18	021-85623492	上海
0203004	马六	男	17	022-12369872	天津
0203005	赵七	男	18	010-65987412	北京
…	…	…	…	…	…

表 1-1 称为一个数据结构，表中的每一行是一个结点（或记录），它是由学号、姓名、性别、年龄、联系方式、家庭住址等数据项组成。该表中数据元素之间的逻辑关系是：对表中任意一个结点，与它相邻且在它前面的结点（或称为直接前驱，immediate predecessor）最多只有一个；与表中任意一个结点相邻且在其后的结点（或称为直接后继，immediate successor）也最多只有一个。表中只有第一个结点没有直接前驱，故称为开始结点；只有最后一个结点没有直接后继，故称之为终端结点。例如，表中“王五”所在结点的直接前驱和直接后继分别是“李四”和“马六”所在的结点，上述结点间的关系构成了学生通讯录的逻辑结构。

该表的存储结构则是指用计算机语言如何表示结点之间的这种关系，即表中的结点是顺序邻接地存储在一片连续的单元之中，还是用指针将这些结点链接在一起。在这个表中，可能要经常查看某一学生的学号，当学生退学时要删除相应的结点，进来新学生时要增加结点。怎样进行查找、删除、插入，这就是数据的运算问题。搞清楚上述三个问题，也就弄清了学生通讯录这个数据结构。

综上所述，可以将数据结构定义为：按某种逻辑关系组织起来的一批数据，应用计算机语言，按一定的存储表示方式把它们存储在计算机的存储器中，并在这些数据上定义了一个运算的集合，就叫做一个数据结构。

在不产生混淆的前提下，我们常常将数据的逻辑结构简称为数据结构。数据的逻辑结构有两大类：

① 线性结构。线性结构的逻辑特征是有且仅有一个开始结点和一个终端结点，并且所有结点都最多只有一个直接前驱和一个直接后继。例如，让表 1-1 中的几个同学按照学号站成一排，第 2 章中所要介绍的线性表就是一个典型的线性结构。

② 非线性结构。非线性结构的逻辑特征是一个结点可能有多个直接前驱和直接后继。例如，让表 1-1 中的几个同学手拉手站成一排。第 6 章介绍的图就是一个典型的非线性结构。

存储结构是数据结构不可缺少的一个方面，因此常常将同一逻辑结构的不同存储结构，冠以不同的数据结构名称来标识它们。例如，线性表是一种逻辑结构，若是采用顺序方法的存储表示，则称该结构为顺序表；若是采用链接方法的存储表示，则称该结构为链表。

同理，数据的运算也是数据结构不可分割的一个方面，在给定了数据的逻辑结构和存储结构之后，按定义的运算集合及其运算的性质不同，也可能导致完全不同的数据结构。例如，若对线性表的插入、删除运算限制在表的一端进行，则该线性表称为栈；若插入限制在表的一端进行，而删除限制在表的另一端进行，则该线性表称为队列。更进一步，若线性表采用顺序表或链表作为存储结构，则对插入和删除运算做了上述限制之后，可分别得到顺序栈或链栈、顺序队列或链队列。

数据的存储结构可用以下四种基本存储方法得到：

① 顺序存储方法。顺序存储方法是把逻辑上相邻的结点存储在物理位置相邻的存储单元中，结点间的逻辑关系由存储单元的邻接关系来体现。由此得到的存储表示称为顺序存储结构（sequential storage structure），通常顺序存储结构是借助于程序语言中的数组来描述的。该方法主要应用于线性的数据结构，非线性的数据结构也可以通过某种线性化的方法来实现顺序存储。

② 链接存储方法。链接存储方法不要求逻辑上相邻的结点在物理位置上也相邻，结点间的逻辑关系是由附加的指针字段表示的。由此得到的存储表示称为链式存储结构（linked storage structure），通常链接存储结构要借助于程序语言的指针类型来描述它。

③ 索引存储方法。索引存储方法通常是在存储结点信息的同时，建立附加的索引表。索引表中的每一项称为索引项，索引项的一般形式是关键字、地址，关键字是能唯一标识一个结点的那些数据项。若每个结点在索引表中都有一个索引项，则该索引表称为稠密索引（dense index）。若一组结点在索引表中只对应一个索引项，则该索引表称为稀疏索引（sparse index）。稠密索引中索引项地址指出结点所在的存储位置，而稀疏索引中索引项的地址则指示一组结点的起始存储位置。

④ 散列存储方法。散列存储方法的基本思想是根据结点的关键字直接计算出该结点的存储地址。

上述四种基本存储方法，既可以单独使用，也可以组合起来对数据结构进行存储映像。同一种逻辑结构采用不同的存储方法，可以得到不同的存储结构。选择何种存储结构来表示相应的逻辑结构，视具体要求而定，主要考虑运算是否方便及算法的时空要求。

有的教科书上将数据的逻辑结构和数据的存储结构定义为数据结构，而将数据的运算定义为数据结构的操作。但是，无论怎样定义数据结构，都应该将数据的逻辑结构、数据的存储结构及数据的运算这三方面看成一个整体。希望读者学习时，不要孤立地去理解某一个方面，而要注意它们之间的联系。

1.3 算法的基本概念

通俗地讲，一个算法就是一种解题方法。严格地讲，就是指解题方案的准确和对此完整的描述。

对于一个问题，如果可以通过一个计算机程序，在有限的存储空间内运行有限长的时间而得

到正确的结果，则称这个问题是算法可解的。但算法不等于程序，也不等于计算方法。当然，程序也可以作为算法的一种描述，但程序通常还需考虑很多与方法和分析无关的细节问题，这是因为在编写程序时要受到计算机系统运行环境的限制。通常，程序的编制不可能优于算法的设计。

1.3.1 算法的基本特征

算法是对特定问题求解步骤的一种描述，它是指令的有限序列，其中每一条指令表示一个或多个操作。此外，一个算法还具有以下几个基本特征：

1. 可行性（effectiveness）

算法的可行性包括以下两个方面：

① 算法中的每一个步骤必须能够实现。例如，在算法中不允许执行分母为 0 的操作，在实数范围内不能求一个负数的平方根等。

② 算法执行的结果要能够达到预期的目的。

针对实际问题设计的算法，人们总是希望能够得到满意的结果。但一个算法又总是在某个特定的计算工具上执行。因此，算法在执行过程中往往受到计算工具的限制，使执行结果产生偏差。例如，在进行数值计算时，如果某计算工具具有 7 位有效数字（如程序设计语言中的单精度运算），则在计算下列三个量 $X=10^{20}$，$Y=1$，$Z=-10^{20}$ 的和时，如果采用不同的运算顺序，就会得到不同的结果，即：

$X+Y+Z=10^{20}+1+(-10^{20})=0$

$X+Z+Y=10^{20}+(-10^{20})+1=1$

而在数学上，$X+Y+Z$ 与 $X+Z+Y$ 的结果是完全等价的。因此，算法与计算公式是有差别的。在设计一个算法时，必须考虑它的可行性，否则可能不会得到满意的结果。

2. 确定性（definiteness）

算法的确定性是指算法中的每一个步骤都必须是有明确定义的，不允许有模棱两可的解释，也不允许有多义性。在解决实际问题时，可能会出现这样或那样的情况。例如，针对某种特殊问题，数学公式是正确的，但按此数学公式设计的计算过程可能会使计算机系统无所适从。这是因为根据数学公式设计的计算过程只考虑了正常使用的情况，而当出现异常情况时此计算过程就不适用了。

3. 有穷性（finiteness）

算法的有穷性是指算法必须能在有限的时间内做完，即算法必须能够在执行有限个步骤之后终止。数学中的无穷级数在计算时只能取有限项，即计算无穷级数值的过程只能是有穷的。因此，一个数的无穷级数表示只是一个计算公式，而根据精度要求确定的计算过程才是有穷的算法。

算法的有穷性还应包括合理的执行时间的含义。因为如果一个算法需要执行千万年，显然失去了实用价值。

4. 拥有足够的信息

一个算法是否有效还取决于为算法所提供的信息是否足够。通常，算法中的各种运算总是要施加到各个运算对象上，而这些运算对象又可能具有某种初始状态，这是算法执行的起点或依据。因此，一个算法执行的结果总是与输入的初始数据有关，不同的输入将会有不同的输出结果。当输入不够或输入错误时，算法本身也就无法执行或导致执行出错。一般来说，当算法

拥有足够的信息时，此算法才是有效的，而当提供的信息不够时，算法并不有效。

综上所述，所谓算法，是一组严谨地定义运算顺序的规则，并且每一个规则都是有效的，且是明确的，此顺序将在有限的次数下终止。

1.3.2 算法设计基本方法

下面介绍几种常用的算法设计方法。

1. 列举法

列举法的基本思想是：根据提出的问题，列举所有可能的情况，并用问题中结合的条件检验哪些是需要的，哪些是不需要的。因此，列举法常用于解决“是否存在”或“有多少种可能”等类型的问题，例如求解不定方程的问题。

列举法的特点是算法比较简单，但当列举的可能情况较多时，执行列举算法的工作量将很大。因此，在用列举法设计算法时，使方案优化，尽量减少运算工作量，是应该重点注意的。通常，在设计列举算法时，只要对实际问题进行详细的分析，将与问题有关的知识条理化、完备化、系统化，从中找出规律；或对所有可能的情况进行分类，引出一些有用的信息，是可以大大减少列举量的。

例如，从A市到B市有3条路，从B市到C市有两条路。从A市经过B市到C市有几种走法？作图1-1，然后把每一种走法一一列举出来：

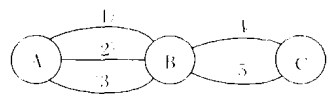

图1-1 走法示意图

第一种走法：A ① B ④ C　　第二种走法：A ① B ⑤ C

第三种走法：A ② B ④ C　　第四种走法：A ② B ⑤ C

第五种走法：A ③ B ④ C　　第六种走法：A ③ B ⑤ C

从A市经过B市到C市共有6种走法。

2. 归纳法

归纳法的基本思想是：通过列举少量的特殊情况，经过分析，最后找出一般的关系。显然，归纳法要比列举法更能反映问题的本质，并且可以解决列举量为无限的问题。但是，从一个实际问题中总结归纳出一般的关系并不是一件容易的事情，要归纳出一个数学模型则更为困难。从本质上讲，归纳就是通过观察一些简单而特殊的情况，最后总结出有用的结论或解决问题的有效途径。

归纳是一种抽象，即从特殊现象中找出一般关系。但由于在归纳过程中不可能对所有的情况进行列举，因此最后由归纳得到的结论还只是一种猜测，还需要对这种猜测加以必要的证明。实际上，通过精心观察而得到的猜测得不到证实或最后证明猜测是错的，也是常有的事。

例如，买葡萄的时候就用到了归纳法，我们往往先尝一尝，如果尝的几个很甜，就归纳出所有的葡萄都很甜的，然后可放心地买上一大串。

3. 递推

所谓递推，是指从已知的初始条件出发，逐次推出所要求的各中间结果和最后结果。其中，初始条件或者问题本身已经给定，或是通过对问题的分析与化简而得到确定。递推本质上也属于归纳，工程上许多递推关系式实际上是通过对实际问题的分析与归纳而得到的，因此递推关系式往往是归纳的结果。