

21世纪计算机科学与技术实践教学

丛书主编 陈明

陈明 编著

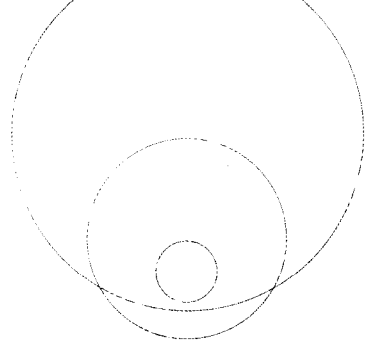
数据结构 (C语言版)

清华大学出版社



21世纪计算机科学与技术实践型教程

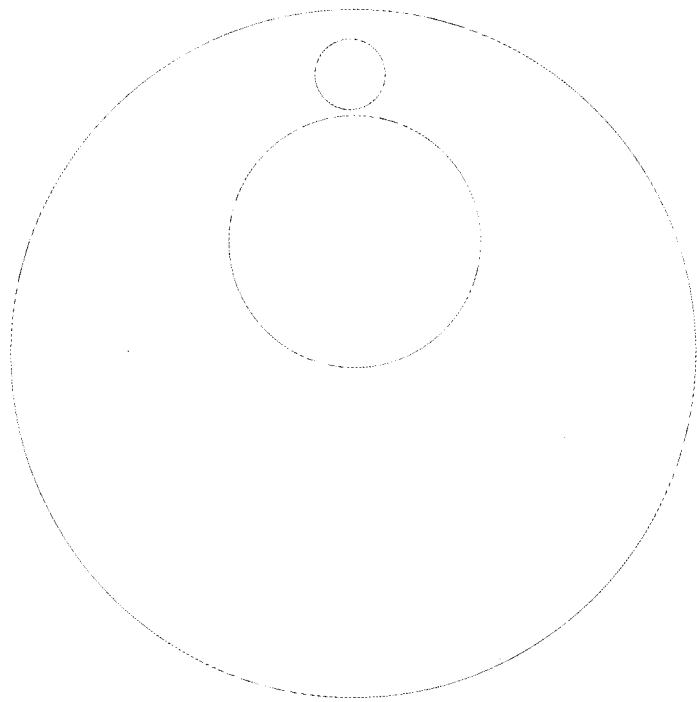
丛书主编 陈明



陈明 编著

数据结构

(C语言版)



清华大学出版社
北京

内 容 简 介

本书系统地介绍了各种典型的数据结构,主要包括线性表、栈和队列、串、数组和广义表、树、图、查找、排序、递归和文件等,为了加强对算法的理解,还介绍了算法分析方面的内容。

数据结构课程是计算机科学与技术专业最基础的课程之一,学好数据结构对于建立计算机科学技术基础和培养优秀的编程素质十分重要。本书是一部实践性很强的数据结构教学用书,书中不仅详细地介绍了数据结构的内容、方法与意义,而且通过大量的例子说明概念与算法,从而使学生能够更好地理解和运用所学知识。每章后的练习题能够加深对各章内容的认识,附录中给出的习题参考答案可以直接检查练习的结果和解题水平,让学生通过动手操作掌握知识。同时,各章还给出了上机练习的具体指导。

版权所有,翻印必究。举报电话:010 62782989 13501256678 13801310933

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

本书防伪标签采用特殊防伪技术,用户可通过在图案表面涂抹清水,图案消失,水干后图案复现;或将表面膜揭下,放在白纸上用彩笔涂抹,图案在白纸上再现的方法识别真伪。

图书在版编目(CIP)数据

数据结构(C语言版)/陈明编著. —北京:清华大学出版社,2005.9

(21世纪计算机科学与技术实践型教程)

ISBN 7-302-11230-4

I. 数… II. 陈… III. ①数据结构—高等学校:技术学校—教材 ②C语言—程序设计—高等学校:技术学校—教材 IV. ①TP311.12 ②TP312

中国版本图书馆CIP数据核字(2005)第065081号

出版者:清华大学出版社

地 址:北京清华大学学研大厦

<http://www.tup.com.cn>

邮 编:100084

社总机:010-62770175

客户服务:010-62776969

责任编辑:谢 琛 Email: xiech@tup.tsinghua.edu.cn

印装者:北京国马印刷厂

发 行 者:新华书店总店北京发行所

开 本:185×260 印张:24.75 字数:566千字

版 次:2005年9月第1版 2005年9月第1次印刷

书 号:ISBN 7-302-11230-4/TP·7410

印 数:1~5000

定 价:29.00元

《21 世纪计算机科学与技术实践型教程》

编辑委员会

主任：

陈 明

中国石油大学教授

委员(按姓氏笔画排序)：

毛国君

北京工业大学教授

叶新铭

内蒙古大学教授

刘淑芬

吉林大学教授

刘书家

北京工商大学教授

白中英

北京邮电大学教授

汤 庸

中山大学教授

何炎祥

武汉大学教授

陈永义

北京气象学院教授

罗四维

北京交通大学教授

段友祥

中国石油大学教授

高维东

南开大学教授

郭 禾

大连理工大学副教授

姚 琳

北京科技大学副教授

崔武子

北京联合大学副教授

谢树煜

清华大学教授

焦金生

清华大学教授

曹元大

北京理工大学教授

韩江洪

合肥工业大学教授

策划编辑：谢 琛

《21 世纪计算机科学与技术实践型教程》

序

21 世纪影响世界的三大关键技术是：以计算机和网络为代表的信息技术；以基因工程为代表的生命科学和生物技术；以纳米技术为代表的新型材料技术。信息技术居三大关键技术之首。国民经济的发展采取信息化带动现代化的方针，要求在所有领域中迅速推广信息技术，导致需要大量的计算机科学与技术领域的优秀人才。

计算机科学与技术的广泛应用是计算机学科发展的原动力，计算机科学是一门应用科学。因此，计算机学科的优秀人才不仅应具有坚实的科学理论基础，而且更重要的是能将理论与实践相结合，并具有解决实际问题的能力。培养计算机科学与技术的优秀人才是社会的需要、国民经济发展的需要。

制定科学的教学计划对于培养计算机科学与技术人才十分重要，而教材的选择是实施教学计划的一个重要组成部分，《21 世纪计算机科学与技术实践型教程》主要考虑了下述两方面。

一方面，高等学校的计算机科学与技术专业的学生，在学习了基本的必修课和部分选修课程之后，立刻进行计算机应用系统的软件和硬件开发与应用尚存在一些困难，而《21 世纪计算机科学与技术实践型教程》就是为了填补这部分鸿沟。将理论与实际联系起来，结合起来，使学生不仅学会了计算机科学理论，而且也学会应用这些理论解决实际问题。

另一方面，计算机科学与技术专业的课程内容需要经过实践练习，才能深刻理解和掌握。因此，本套教材增强了实践性、应用性和可理解性，并在体例上做了改进——使用案例说明。

实践型教学占有重要的位置，不仅体现了理论和实践紧密结合的学科特征，而且对于提高学生的综合素质，培养学生的创新精神与实践能力有特殊的作用。因此，研究和撰写实践型教材是必须的，也是十分重要的任务。优秀的教材是保证高水平教学的重要因素，选择水平高、内容新、实践性强的教材可以促进课堂教学质量的快速提升。在教学中，应用实践型教材可以增强学生的认知能力、创新能力、实践能力以及团队协作和交流表达能力。

实践型教材应由教学经验丰富、实际应用经验丰富的教师撰写。此系列教材的作者不但从事多年的计算机教学，而且参加并完成了多项计算机类的科研项目，把他们积累的经验、知识、智慧、素质融合于教材中，奉献给计算机科学与技术的教学。

我们在组织本系列教材过程中，虽然经过了详细地思考和讨论，但毕竟是初步的尝试，不完善甚至缺点不可避免，敬请读者指正。

本系列教材主编 陈明

2005 年 1 月于北京

前 言

在非数值计算中,处理对象已从简单数值发展到具有一定结构的数据,这就需要讨论如何有效地组织计算机的存储,并在此基础上有效地实现对象间的运算,数据结构就是研究与解决这些问题的重要基础。

数据结构课程是计算机科学与技术专业的一门必修的、重要的专业基础课,是计算机程序设计的重要理论技术基础。通过对数据结构的学习,不仅可以使学生掌握数据结构的基本内容、典型算法和使用方法,而且能够训练学生应用数据结构和算法进行具体应用问题的程序设计能力。

本书共分 12 章,分别介绍了最常用的数据结构,各种数据结构的逻辑关系及其在计算机中的存储表示,以及在数据结构上的运算等。主要内容包括:算法分析、线性表、栈和队列、串、数组和广义表、树、图、查找、排序、递归和文件等内容。各章中所涉及的数据结构与算法都给出了类 C 语言描述。

本书在结构上呈积木式,注重实践练习与应用。在教材的内容设置上,为了有助于学生加强动手实践环节,对于各种常用数据结构的介绍从实际出发,避免抽象的理论论述和复杂的公式推导,在典型的算法介绍中深入浅出、简洁明了。多处论述采用了提出问题、解决问题和归纳问题的步骤,注意从实际到理论、从具体到抽象、从个别到一般的原则。每章都设有小结、习题和上机实验。通过这些习题的练习,不仅能加深对基本概念和定义的理解,而且通过上机,能够提高编程能力和程序调试能力。在附录中给出了每章的习题答案。

在编写过程中,尽量多列举实例练习,注重了多种算法和方法的比较,在使用本教材时,可以根据具体的环境不同和对学生的要求不同而选择使用。

本书可以作为高等学校计算机专业和相近专业的教材,也可作为从事计算机应用的工程技术人员的参考书。

由于作者水平有限,书中不足之处在所难免,敬请读者批评指正。

陈 明

2005 年 6 月于北京

目 录

第 1 章 绪论	1
1.1 数据结构的重要性	1
1.2 基本术语	2
1.3 数据结构的概念	5
1.4 数据的逻辑结构	7
1.5 数据的存储结构	8
1.6 数据的运算	10
1.7 数据的逻辑结构、存储结构及运算的关系	11
1.8 算法的描述	11
1.9 本章小结	15
1.10 习题一	15
1.11 上机实验	16
第 2 章 算法分析	17
2.1 算法分析的概念	17
2.2 算法运行时间举例	19
2.3 最大连续子序列之和问题	20
2.3.1 简单易懂的 $O(N^3)$ 算法	20
2.3.2 一个改进的 $O(N^2)$ 算法	22
2.3.3 一个线性算法	23
2.4 静态搜索问题	24
2.4.1 顺序搜索	25
2.4.2 二分搜索	25
2.4.3 插值搜索	27
2.5 检验一个算法分析	27
2.6 Big-Oh 分析法的限制	28
2.7 本章小结	29
2.8 习题二	29

第 3 章 线性表	31
3.1 线性表及逻辑结构	31
3.2 线性表的顺序存储	35
3.2.1 顺序存储	35
3.2.2 顺序结构线性表的运算	36
3.2.3 顺序存储结构的特点	40
3.3 线性表的链式存储	41
3.3.1 线性链表	41
3.3.2 线性链表的运算	44
3.3.3 静态链表	49
3.3.4 静态链表的运算	50
3.3.5 循环链表	50
3.3.6 循环链表的运算	51
3.3.7 双向链表	52
3.3.8 双向链表的运算	53
3.3.9 链式存储结构的特点	55
3.4 链式存储结构的应用	56
3.4.1 约瑟夫问题	56
3.4.2 一元多项式求和	57
3.4.3 在集合方面的应用	61
3.5 本章小结	62
3.6 习题三	63
3.7 上机实验	64
第 4 章 栈和队列	65
4.1 栈	65
4.1.1 栈的定义	65
4.1.2 栈的顺序存储结构	66
4.1.3 栈的链式存储结构	70
4.1.4 顺序栈和链式栈的比较	72
4.2 栈的应用	73
4.2.1 迷宫问题	73
4.2.2 算术表达式求值	76
4.2.3 子程序的调用和返回	79
4.2.4 数制转换	79
4.2.5 行编辑	80
4.3 队列	81

4.3.1	队列的定义	81
4.3.2	队列的顺序存储	82
4.3.3	队列的链式存储	88
4.3.4	优先队列	92
4.4	队列的应用	93
4.4.1	解决设备速度不匹配问题	93
4.4.2	舞伴问题	93
4.5	本章小结	95
4.6	习题四	95
4.7	上机实验	96
第5章	串	97
5.1	串的基本概念	97
5.2	串的存储结构	98
5.2.1	串的静态存储结构	98
5.2.2	串的动态存储结构	100
5.3	串的基本运算和算法	102
5.3.1	串的基本运算	102
5.3.2	实现串的基本运算的算法	103
5.4	模式匹配	107
5.4.1	模式匹配的 BF 算法	107
5.4.2	模式匹配的 KMP 算法	110
5.5	串在文本编辑中的应用	114
5.6	本章小结	116
5.7	习题五	116
5.8	上机实验	117
第6章	数组与广义表	118
6.1	数组的定义及其基本操作	118
6.1.1	数组的定义	118
6.1.2	数组的基本操作	119
6.2	数组的顺序存储结构	120
6.3	矩阵的压缩存储	124
6.3.1	特殊矩阵的压缩存储	124
6.3.2	稀疏矩阵的压缩存储	126
6.4	广义表的概念	141
6.5	广义表的存储结构表示	142
6.6	广义表的运算	145

6.7	本章小结	145
6.8	习题六	145
6.9	上机实验	147
第 7 章	树	148
7.1	树的定义及基本操作	148
7.1.1	树的定义	148
7.1.2	树的表示形式	149
7.1.3	树的常用概念	150
7.1.4	树的基本操作	150
7.2	二叉树	151
7.2.1	二叉树的定义	151
7.2.2	二叉树的性质	152
7.2.3	二叉树的存储结构	155
7.2.4	二叉树的遍历	157
7.2.5	二叉树遍历的应用	160
7.3	线索二叉树	162
7.3.1	线索二叉树的定义	162
7.3.2	线索二叉树的遍历	163
7.3.3	线索二叉树的运算	167
7.4	树、森林和二叉树的关系	168
7.4.1	树的存储结构	168
7.4.2	森林与二叉树的转换	172
7.4.3	树和森林的遍历	175
7.5	哈夫曼树及其应用	176
7.5.1	哈夫曼树的定义	176
7.5.2	哈夫曼树的构造	177
7.5.3	哈夫曼树在编码问题中的应用	179
7.6	本章小结	183
7.7	习题七	184
7.8	上机实验	187
第 8 章	图	188
8.1	图的基本概念	188
8.2	图的存储结构	192
8.2.1	邻接矩阵	192
8.2.2	邻接表	195

8.2.3	十字链表	198
8.2.4	邻接多重表	200
8.3	图的遍历	201
8.3.1	深度优先搜索	202
8.3.2	广度优先搜索	204
8.4	生成树	206
8.4.1	普里姆算法	207
8.4.2	克鲁斯卡尔算法	210
8.5	最短路径	213
8.5.1	单源最短路径	213
8.5.2	每一对顶点之间的最短路径	216
8.6	拓扑排序	217
8.7	关键路径	221
8.8	本章小结	228
8.9	习题八	229
8.10	上机实验	232
第9章	查找	233
9.1	基本概念	233
9.2	线性表的查找	234
9.2.1	顺序查找	234
9.2.2	折半查找	236
9.2.3	分块查找	238
9.3	树表的查找	241
9.3.1	二叉查找树	242
9.3.2	平衡二叉树	248
9.3.3	B ⁺ 树	254
9.4	哈希表的查找	256
9.4.1	哈希表	256
9.4.2	构造哈希表的基本方法	257
9.4.3	解决冲突的方法	259
9.4.4	哈希表的查找方法	262
9.5	各种查找方法的比较	263
9.6	本章小结	264
9.7	习题九	264
9.8	上机实验	266

第 10 章 排序	267
10.1 基本概念	267
10.1.1 排序方法的分类	269
10.1.2 排序算法分析	269
10.2 内部排序	270
10.2.1 插入排序	270
10.2.2 冒泡排序	274
10.2.3 快速排序	276
10.2.4 选择排序	278
10.2.5 归并排序	284
10.2.6 基数排序	287
10.3 内部排序方法比较	291
10.4 外部排序简介	293
10.5 本章小结	293
10.6 习题十	294
10.7 上机实验	296
第 11 章 递归	297
11.1 递归的定义	297
11.2 常见递归问题	298
11.2.1 汉诺塔问题	298
11.2.2 八皇后问题	300
11.2.3 组合公式的计算	302
11.3 递归的实现	303
11.4 递归转化为非递归的一般过程	306
11.5 递归的时间和空间复杂度	310
11.6 本章小结	311
11.7 习题十一	311
11.8 上机实验	312
第 12 章 文件	313
12.1 外存储器的介绍	313
12.2 文件的概念	314
12.2.1 文件及其类别	315
12.2.2 文件的操作	316
12.3 文件的组织	317

目录 数据结构(C语言版)

12.3.1 顺序文件.....	318
12.3.2 索引文件.....	319
12.3.3 散列文件.....	324
12.3.4 多关键字文件.....	326
12.4 本章小结.....	328
12.5 习题十二.....	328
附录 A 各章习题答案与提示.....	330
参考文献.....	381

第 1 章 绪 论

在开始学习数据结构(data structure)之前,首先了解学习数据结构的意义、数据结构的含义及数据结构的一些相关基本概念等。这对于深刻理解书中各章节的内容将会有很大的帮助。

1.1 数据结构的重要性

在计算机发展的初期,计算机主要用于处理数值计算问题,程序设计人员也主要把精力集中在程序设计的方法与技巧上,但随着计算机应用领域的扩大和软硬件的发展,计算机对信息的处理已从单一的数值计算发展到大量地解决非数值计算问题,其处理的信息也由简单的数值发展到字符、图像、声音等具有复杂结构的数据。而数据结构就是随着计算机的发展而出现的一门较新的计算机课程。

非数值计算问题的数据之间的相互关系一般无法完全用数学方程式加以描述,并且系统程序和许多应用程序的规模庞大,结构复杂,这时考虑问题的关键已不再是数学分析和计算方法,而是能否设计出合适的数据结构,从而有效地解决问题。

计算机科学是一门研究用计算机进行信息表示和处理的科学。这里面涉及两个问题:信息的表示和信息的处理。信息的表示和组成又直接影响到处理信息程序的效率。随着计算机的普及,信息量的增加,信息范围的拓宽,需要人们对计算机程序加工的对象进行系统地研究,即研究数据的特性以及数据之间存在的关系,而数据结构正是描述数据特性以及数据之间存在的关系的一门课程。

例 1-1 电话号码查询问题。

编写一个查询某个城市或单位的私人电话号码的程序。要求对任意给出的一个姓名,若该人有电话号码,则迅速找到其电话号码;否则指出该人没有电话号码。

要解决此问题首先需要构造一张电话号码登记表。表中每个结点存放两个数据项:姓名和电话号码。

要写出好的查找算法,取决于这张表的结构及存储方式。最简单的方式是将表中结点顺序地存储在计算机中。查找时从头开始依次查找姓名,直到找出正确的姓名或是找遍整个表均没有找到为止。这种查找算法对于一个不大的单位或许是可行的,但对一个

有成千上万私人电话的城市就不实用了。若这张表是按姓氏排列的,则可另造一张姓氏索引表,采用如图 1-1 所示的存储结构,地址指明电话号码登记表中某姓氏的首地址。那么查找过程是先在索引表中查找姓氏,然后根据索引表中的地址到电话号码登记表中核查姓名,这样查找登记时就无须查找其他姓氏的名字了。因此,在这种新的结构上产生的查找算法就更为有效和高效。

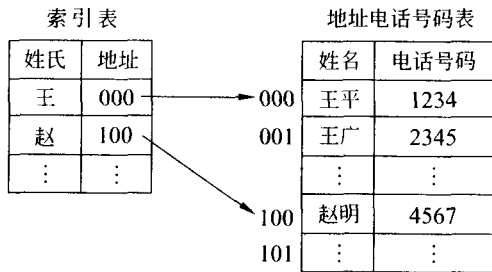


图 1-1 电话号码查询

数据结构是计算机专业的核心课程之一,在众多的计算机系统软件和应用软件中都要用到各种数据结构。可以这样说,数据结构不仅是一般程序设计的基础,而且是实现编译程序、操作系统、数据库系统及其他系统程序和大型应用程序的基础。因此,仅掌握几种计算机程序语言是难以应付众多复杂的研究课题的,要想有效地使用计算机,还必须学习数据结构的知识。

瑞士计算机科学家 Niklaus Wirth 教授曾提出这样一个等式:算法+数据结构=程序,这个等式形象地描述了算法、数据结构和程序之间的关系,这里的数据结构指的是数据的逻辑结构和存储结构,而算法就是对数据运算的描述。由此可见,程序设计的实质就是对实际问题选取一种好的数据结构,加之设计一个好的算法,而且好的算法很大程度上取决于描述实际问题的数据结构。对于一个程序设计人员来说,学好数据结构是十分重要的,下面一节就是对数据结构和一些相关概念的解释。

1.2 基本术语

为了更好地理解数据结构这个概念,先给出数据结构中一些常用名词和术语的解释。

数据(data) 信息的载体,它是描述客观事物的数、字符以及所有能输入到计算机中被计算机程序识别、加工处理的信息的集合。数据不只是通常意义下的数字等,随着计算机的广泛应用,数据的范畴也随之拓宽,计算机可以处理的字符串、图像、声音等都可以称为数据。所以不能只是泛泛地理解数据这个概念。如表 1-1 所示,列出了学号为“S01012”的学生姓名是“张风”,“张风”就是该同学的姓名数据。

表 1-1 学生成绩表

学号	姓名	语文	数学	英语
S01012	张风	85	69	92
S01022	李强	87	73	74
S02013	王海	92	64	84

数据项(data item) 是数据具有独立意义的不可分的最小单位,它是对数据的数据元素属性的描述。数据项也被称为字段或域。

数据元素(data element) 是数据的基本单位,是对一个客观实体的数据描述。一个数据元素可以由一个或若干个数据项组成,如图 1-2 所示。数据元素也被称为结点或记录。

利用上面表 1-1 来说明数据项和数据元素,整个表记录的是学生的基本信息和成绩数据,一个学生的信息和成绩就是其中的一个数据元素。

学号	姓名	语文	数学	英语
S01012	张风	85	69	92
S01022	李强	87	73	74
S02013	王海	92	64	84

图 1-2 数据元素和数据项

一个数据元素由学号、姓名、语文成绩、数学成绩和英语成绩五个数据项组成。

数据对象(data object) 是具有相同性质的数据元素的集合,它是数据的一个子集。如上例所示,一个班级的成绩表可以看作一个数据对象。例如,集合 $\{1,2,3,4,5,\dots\}$ 是自然数的数据对象,而集合 $\{a,b,c,d,\dots,z\}$ 是英文字母表的数据对象。可以看出,数据对象可以是无限的,也可以是有限的。

数据类型(data type) 是具有相同性质的计算机数据的集合及定义在这个数据集合上的一组操作的总称。例如,在 C 语言中的整数数据类型是集合 $C=\{0,\pm 1,\pm 2,\pm 3,\pm 4,\dots\}$ 及定义在该集合上的加、减、乘、整除和取余等一组操作。数据类型封装了数据存储与操作的具体细节。

每个数据项属于某个确定的基本数据类型,数据类型可分为两类:原子类型和结构类型。

原子类型 如果一个数据元素由一个数据项组成,这个数据元素的类型就是这个数据项的原子类型,值在逻辑上不可分解(如 `int i, float j`)。

结构类型 如果由多个不同类型的数据项组成,则这个数据元素的类型就是由各数据项类型构造而成的结构类型,值由若干成分按某种结构组成,如 `struct stu`。上面提到的学生成绩表中,数据项姓名的数据类型为字符型,而成绩的数据类型是整型的,所以这个数据元素是一个结构类型。上述成绩表数据用 C 语言的结构体数组 `Class1stu[50]` 来存储。


```
struct stu
{ /* 数据项 */
int stuID;
char name[20];
int maths_score;
int chinese_score;
int english_score;
} Class1stu[50];
```

不同的高级语言提供的基本数据类型有所不同,在C语言中,提供了实型、整型、字符型和指针型等基本数据类型。

抽象数据类型(abstract data type) 就是对象的数学模型,简称为 ADT,是用户在水数据类型基础上新定义的数据类型。抽象数据类型定义包括数据组成和对数据的处理操作。可见抽象数据类型是数据和数据使用者的一个接口。使用者通过 ADT 中定义的接口来创建、撤销和操纵 ADT 中的对象。

抽象数据类型的定义仅取决于它的一组逻辑特性,而与它在计算机中如何表示和实现无关,只要其数学特性不变,都不影响其外部的使用。从抽象数据类型的角度入手设计软件系统,可大大提高软件构件的复用率。抽象数据类型的定义包括数据对象定义、数据关系定义和基本操作定义三部分,其书写格式为

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。}

抽象数据类型可以看作是描述问题的模型,它独立于具体实现。它的优点是将数据和操作封装在一起,使得用户程序只能通过在 ADT 里定义的某些操作来访问其中的数据,从而实现了信息隐藏。在 C++ 中,可以用类(包括模板类)的说明来表示 ADT,用类的实现来实现 ADT。因此,C++ 中实现的类相当于数据的存储结构及其在存储结构上实现的对数据的操作。

ADT 和类的概念实际上反映了程序或软件设计的两层抽象: ADT 相当于是在概念层(或称为抽象层)上描述问题,而类相当于在实现层上描述问题。此外,C++ 中的类只是一个由用户定义的普通类型,可用它来定义变量(称为对象或类的实例)。因此,在 C++ 中,最终是通过操作对象来解决实际问题的,所以可将该层次看作是应用层。例如,main 程序就可看作是用户的应用程序。

由于 C 语言中没有提供“类”这一数据类型,因此无法实现 ADT,所以不采用 ADT 的形式来描述数据结构。但它实际上等价于定义的数据的逻辑结构以及在逻辑结构上定义的抽象操作。

数据结构 数据之间的相互关系(即数据的组织形式)及在这些数据上定义的数据运算方法的集合。