普通高等教育“十三五”创新型规划教材
理论+实践+数字资源一体化规划教材

紧扣教学大纲，突出重点
强化应用能力，迁移拓展
支持教学做考，立体资源

SHUJU JIEGOU

数据结构

主编 张珊靓 朱宗胜

电子科技大学出版社

普通高等教育“十三五”创新型规划教材
理论+实践+数字资源一体化规划教材

SHUJU JIEGOU

数据结构

主　审　余建国
主　编　张珊靓　朱宗胜
副主编　李　娜　王庆喜　王　勇　齐万华

电子科技大学出版社

图书在版编目(CIP)数据

数据结构 / 张珊靓，朱宗胜主编. — 成都 ：电子科技大学出版社，2016.9

ISBN 978-7-5647-3587-6

Ⅰ. ①数… Ⅱ. ①张… ②朱… Ⅲ. ①数据结构 Ⅳ. ①TP311.12

中国版本图书馆 CIP 数据核字(2016)第 089716 号

数据结构

主　编　张珊靓　朱宗胜

出　　版：电子科技大学出版社(成都市一环路东一段 159 号电子信息产业大厦　邮编：610051)

策划编辑：杜　倩

责任编辑：杨仪玮

主　　页：www.uestcp.com.cn

电子邮箱：uestcp@uestcp.com.cn

发　　行：新华书店经销

印　　刷：天津市蓟县宏图印务有限公司

成品尺寸：203mm×260mm　**印张** 16.75　**字数** 418 千字

版　　次：2016 年 9 月第一版

印　　次：2016 年 9 月第一次印刷

书　　号：ISBN 978-7-5647-3587-6

定　　价：35.00 元

前　言

数据结构对计算机学科起到承前启后的作用，因此它是计算机专业重要的专业基础课程之一，也是计算机及相关专业考研和水平等级考试的必考科目，而且正逐渐发展成为众多理工专业的热门选修课。

数据结构主要研究数据间的联系（数据的逻辑结构）、数据在计算机中的存储方法（数据的物理结构）以及处理不同结构数据的算法。计算机编程中加工处理的对象是数据，而数据具有一定的组织结构，所以学习编写计算机程序仅仅了解计算机语言是不够的，还必须掌握数据组织、存储和运算的一般方法，这便是数据结构课程中所学习和研究的内容，也是我们编写计算机程序的重要基础。

本书内容共分 9 章，第 1 章介绍数据结构的基本概念以及算法的表示和分析方法，是在前导课程“高级语言程序设计”基础上的进一步引申和总结；第 2 章到第 7 章以线性结构、树状结构、图状结构为主线，由简到繁地介绍了顺序表、链表以及栈和队、树和二叉树、图等几种基本数据结构及有关算法；第 8 章和第 9 章详细介绍了在实际应用中广泛使用的查找和排序的基本算法。

由于数据结构的原理和算法较抽象，数据结构课程知识丰富、内容抽象、学习量大，隐藏在各部分内容中的方法和技术多，而该课程一般在本科低年级开设，这对于只具有一些计算机程序设计知识的初学者来说，理解和掌握其中的原理就困难了。我们根据多年的教学经验，在分析国内多种同类教材的基础上，博采众长，编写了这本书，奉献给广大读者。

本书具有以下几个特点：

①用类 C 语言描述数据结构和算法；

②通俗易懂，由浅入深；

③注重理论与实践相结合。

本书由张珊靓、朱宗胜担任主编，李娜、王庆喜、王勇、齐万华担任副主编。由于作者水平有限，书中不妥与疏漏之处难免，敬请广大读者批评指正。

编　者

目　录

第 1 章　数据结构绪论 …… 1

1.1　数据结构的概述 …… 2

1.2　抽象数据类型的表示与实现 …… 8

1.3　算法 …… 12

第 2 章　线性表 …… 23

2.1　线性表的定义及基本操作 …… 24

2.2　线性表的顺序存储结构 …… 26

2.3　线性表的链式存储结构 …… 30

2.4　顺序存储和链式存储的区别 …… 37

2.5　一元多项式相加 …… 37

第 3 章　栈和队列 …… 46

3.1　栈 …… 47

3.2　队列 …… 55

第 4 章　串 …… 71

4.1　串的定义及操作 …… 72

4.2　串的存储结构 …… 73

4.3　串的模式匹配 …… 84

第 5 章　数组和广义表 …… 91

5.1　数组的概念 …… 92

5.2　广义表 …… 110

第 6 章　树与二叉树 …… 124

6.1　树的定义 …… 125

6.2　树的存储结构和基本操作 …… 126

6.3　二叉树的定义和基本性质 …… 130

6.4 二叉树的存储结构和基本操作 …… 132
6.5 二叉树的遍历 …… 135
6.6 线索二叉树 …… 139
6.7 树和森林 …… 147
6.8 哈夫曼树及其应用 …… 148

第 7 章 图 …… 160
7.1 图的定义与操作 …… 161
7.2 图的存储结构 …… 164
7.3 图的遍历 …… 172
7.5 最短路径 …… 182
7.6 拓扑排序 …… 187

第 8 章 查找 …… 194
8.1 查找的基本概念 …… 195
8.2 基于线性表的查找法 …… 197
8.3 基于树的查找 …… 202
8.4 计算式查找法——哈希法 …… 218

第 9 章 排序 …… 226
9.1 排序的基本概念 …… 227
9.2 插入排序 …… 230
9.3 交换排序 …… 236
9.4 选择排序 …… 242
9.5 归并排序 …… 252
9.6 基数排序 …… 254
9.7 排序方法的比较 …… 259

第 1 章　数据结构绪论

学习目标

知识目标

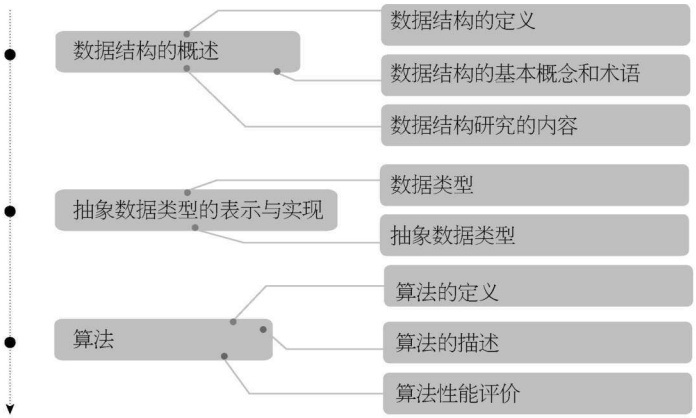
1.熟悉数据结构中各名词、术语的含义，掌握数据结构的基本概念。

2.理解数据类型和抽象数据类型的含义。

3.理解算法要素的确切含义，注意算法与程序的区别。

能力目标

掌握计算语句频度和估算算法时间复杂度的方法。



计算机科学是一门研究数据表示和数据处理的科学。计算机处理的数据不是杂乱无章的,而是有着某种内在的联系,它是计算机可以直接处理的最基本和最重要的对象。无论是进行科学计算或数据处理、过程控制还是对文件的存储和检索及数据库技术等计算机应用领域,都是对数据进行加工处理的过程。因此,要设计出一个结构好效率高的程序,必须研究数据的特性及数据间的相互关系及其对应的存储表示,并利用这些特性和关系设计出相应的算法和程序。

1.1 数据结构的概述

1.1.1 数据结构的定义

1. 什么是数据结构

数据结构在计算机科学界至今没有标准的定义。个人根据各自理解的不同而有不同的表述方法。

Clifford A.Shaffer 在《数据结构与算法分析》一书中的定义是:“数据结构是 ADT(抽象数据类型 Abstract Data Type) 的物理实现。”

Lobert L.Kruse 在《数据结构与程序设计》一书中,将一个数据结构的设计过程分成抽象层、数据结构层和实现层。其中,抽象层是指抽象数据类型层,它讨论数据的逻辑结构及其运算,数据结构层和实现层讨论一个数据结构的表示和在计算机内的存储细节以及运算的实现。

数据结构是指同一数据元素类中各数据元素之间存在的关系。数据结构分为逻辑结构、存储结构(物理结构)和数据的运算。数据的逻辑结构是对数据之间关系的描述,有时就把逻辑结构简称为数据结构。逻辑结构形式的定义为(K,R)或(D,S),其中,K 是数据元素的有限集,R 是 K 上的关系的有限集。

自从美国唐·欧·克努特教授用汇编语言编写的《计算机程序设计技巧》第一卷《基本算法》问世以来,已经出现了用 Pascal、Java、C、C++、C# 等语言编写的数据结构方面的书。总体说来,这些语言基本上分为面向过程的语言和面向对象的语言两大类,也出现过采用两种语言描述数据结构的书籍,如 C 和 C++ 语言描述,但作者实际上是按照 C++ 语言描述。同时采用面向过程和面向对象语言描述数据结构的书籍,目前国内基本上是空白。对于同一种数据结构与算法,同时采用面向过程和面向对象语言进行描述,可以从中更深刻理解这两种思想的不同,这对于深刻理解计算机语言和思想有着重要的作用。C 语言是现在最流行的面向过程的语言,在业界使用非常广泛。而 C# 语言作为微软在新一代开发平台(.NET)上推出的、完全面向对象的语言,凭着其简洁、高效、模板、标准化的特性,就像程序设计语言中的一件艺术品,也吸引着越来越多的开发人员。当然,C# 语言也吸收了 C 语言的一些东西,如语法等,所以,在有些方面,C# 与 C 是相似的。

Niklaus Wirth 说过,Algorithm + Data Structures = Programs,其中,程序设计:为计算机处理问题编制一组指令集,算法:处理问题的策略,数据结构:问题的数学模型。

在计算机发展的早期,数据结构主要用于数值计算,例如:结构静力分析计算线性代数方程组,全球天气预报、环流模式方程,但是随着计算机应用领域的扩大和硬、软件技术的发展,非数值计算问题越来越多(占 90%)。

学生信息检索:计算机处理的对象之间存在着一种简单的线性关系,这类数学模型可以归结为线性数据结构。

计算机和人对弈(八皇后问题等):为了得到合理的布局,计算机要存储当前的布局,从最初的布局开始,每个布局都会衍生出许多新的布局,这时计算机所要处理的是一棵对弈树,也是一种典型的数据结构。

教学计划的编排(交通灯):一个教学计划包含许多课程,课程间有些必须按规定的先后顺序进行,有些则没有次序要求,各个课程间的次序关系可用图来表示。这也是一种典型的数据结构。

由此可见,描述这类非数值计算问题的数学模型已不再是数学方程,而是诸如表、树、图之类的数据结构,因此说,数据结构主要是研究非数值计算的程序设计中所出现的计算机操作对象以及它们之间关系和操作的学科。

2. 数据结构的发展阶段

1968 年克努思教授开创了数据结构的最初体系,他所著的《计算机程序设计艺术》第一卷《基本算法》是第一本较系统地阐述数据的逻辑结构和存储结构及其操作的著作。20 世纪 70 年代初,数据结构作为一门独立的课程开始进入大学课堂。数据结构随着程序设计的发展而发展。程序设计经历了三个阶段:无结构阶段、结构化阶段和面向对象阶段。相应地,数据结构的发展也经历了三个阶段:

(1)无结构阶段。20 世纪 40～60 年代,计算机的应用主要针对科学计算,程序设计技术以机器语言 / 汇编语言为主,程序处理的数据是纯粹的数值,数据之间的关系主要是数学公式或数学模型。在这一阶段,人类的自然语言与计算机编程语言之间存在着巨大的鸿沟,程序设计属于面向计算机的程序设计,设计人员关注的重心是使程序尽可能地被计算机接受并按指令正确执行,至于程序能否让人理解并不重要。

(2)结构化阶段。20 世纪 60～80 年代,计算机开始广泛应用于非数值处理领域,数据表示成为程序设计的重要问题,人们认识到程序设计规范化的重要性,提出了程序结构模块化,并开始注意数据表示与操作的结构化。数据结构及抽象数据类型就是在这种情况下形成的。数据结构概念的引入,对程序设计的规范化起到了重大作用。图灵奖获得者沃思给出了一个著名的公式:数据结构 + 算法 = 程序。从这个公式可以看到,数据结构和算法是构成程序的两个重要的组成部分,一个软件系统通常是以一个或几个关键数据结构为核心而组织的。随着软件系统的规模越来越大、复杂性不断增加,人们不得不对结构化技术重新评价。由于软件系统的实现依赖于关键数据结构,如果这些关键数据结构的一个或几个有所改变,则涉及整个系统,甚至导致整个系统彻底崩溃。

(3)面向对象阶段。面向对象技术(首先是面向对象程序设计)始于 20 世纪 80 年代初,是目前最流行的程序设计技术。在面向对象技术中,问题世界的相关实体被视为一个对象,对象由属性和方法构成,属性用以描述实体的状态或特征,方法用以改变实体的状态或描述实体的行为。一组具有相同属性和方法的对象的集合抽象为类,每个具体的对象都是类的一个实例。例如,“教授”是一个类,“ 赵教授”“ 李教授”等对象都是“教授”类的实例。

由于对象(类)将密切相关的属性(数据)和方法(操作)定义为一个整体,从而实现了封装和信息隐藏。使用类时,无须了解其内部的实现细节,一旦数据(结构)修改了,只需修改类内部的局部代码,软件系统的其余部分无须修改。

数据结构主要强调两个方面的内容:① 数据之间的关系;② 针对这些关系的基本操作。这两个方面实际上蕴涵着面向对象的思想:类重点描述实体的状态与行为,而数据结构重点描述数据之间的关系及其基本操作,数据及其相互关系构成了对实体状态的描述,针对数据元素之间关系的操作构成了对实体行为的描述。由此可见,类与数据结构之间具有对应关系,如图 1-1 所示。

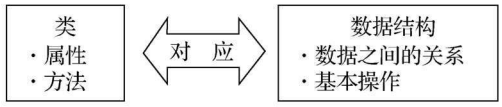


图 1-1 类和数据结构之间的对应关系

数据结构的发展并未终结。一方面,数据结构将继续随着程序设计的发展而发展;另一方面,面向各专门领域的数据结构得到研究和发展,各种实用的高级数据结构被研究出来,各种空间数据结构也在探索中。

1.1.2 数据结构的基本概念和术语

1. 基本概念和术语

现实世界各领域中的大量信息都必须转换成数据才能在计算机中存储、处理。数据是信息的载体。笼统地说,所谓数据,就是计算机加工处理的对象。数据一般分两大类:数值数据(numerical data)和非数值数据(non-numerical data)。

数值数据是一些整数、实数或复数,主要用于工程计算、科学计算和商务处理等。

非数值数据包括字符、文字、图形、图像、语音、表格等。

数据元素(data element)是数据的基本单位。在不同的条件下,数据元素又可称为元素、结点、顶点、记录等。例如,学校学生信息系统中的张军、李强等数据元素。

一个数据元素可由若干个数据项(data item)组成。例如,学籍管理系统中学生信息表的每一个数据元素就是一个学生记录 ,它包括学生的学号、姓名、性别、籍贯、出生年月、成绩等数据项。通常,在解决实际应用问题时是把每个学生记录当作一个基本单位进行访问和处理。

数据对象(data object)是具有相同性质的数据元素的集合。在某个具体问题中,数据元素都具有相同的性质(元素值不一定相等),属于同一数据对象,数据元素是数据对象的一个实例。例如,在交通咨询系统的交通网中,所有顶点的集合是一个数据对象,顶点 A 和顶点 B 各自代表城市,是该数据对象中的两个实例,其数据元素的值分别为 A 和 B。

数据结构主要是为研究和解决如何使用计算机处理非数值问题而产生的理论、技术和方法。它表达数据的构造形式,即一个数据由哪些成分数据构成,以什么方式构成,具有什么结构。在这里,我们称组成数据的成分数据为数据元素。一般地,数据元素可以是简单类型的,如整数、实数、字符等,也可以是结构类型,如记录。若把每个学生的记录看成一个数

据元素，它包括学号、姓名、性别等数据项，一个班的学生记录组成了表 1-1 所示的学生情况表。表是一个数据结构。

表 1-1　学生情况表

学号	姓名	性别	出生年月	联系电话
1001	张三	男	1980-1	2561231
1002	李四	男	1983-9	2561233
2010	王五	男	1986-6	2561231
2032	陈力	女	1978-10	2561237
3040	刘强	男	1979-6	2561221
3069	景山	女	1979-5	2561239
…	…	…	…	…

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。它是由数据元素依据某种逻辑联系组织起来的。

数据结构一般包括以下 3 个方面的内容。

(1)数据元素之间的逻辑关系。所有元素的逻辑关系构成了逻辑结构，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。

(2)数据元素及其逻辑关系在计算机内的存储结构。它是依赖于计算机语言的。一般只在高级语言的层次上讨论存储结构。

(3)数据的运算，是对数据施加的操作。数据运算的定义(功能)是基于逻辑结构的，每一种逻辑结构都有一组相应的操作。例如，插入、删除、修改、查找和排序等。而数据运算的实现是基于存储结构的，是采用某种计算机语言编写的算法。

同一种逻辑结构可以设计多种存储结构，在不同的存储结构中，实现同一种运算的算法可能不同。

2. 逻辑结构

对数据元素之间的逻辑关系的描述称为数据的逻辑结构(logical structure)。根据数据结构中数据元素之间的结构关系的不同特征，可以分为 4 种基本逻辑结构：集合结构(set)、线性结构(linear)、树形结构(tree)和图形结构(graph)。

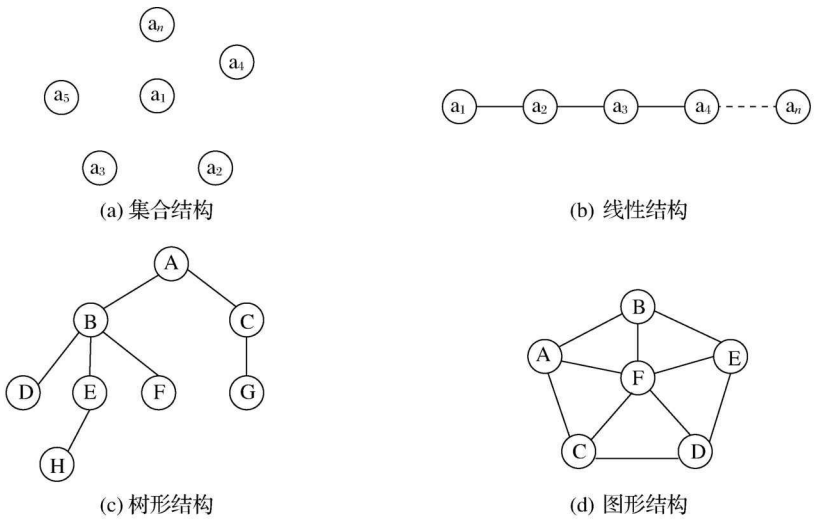
(1)集合结构。结构中的数据元素的有限集合。数据元素之间除了“属于同一个集合”的关系之外没有其他关系。元素顺序是随意、松散的。

(2)线性结构。结构中的数据元素是有序的。数据元素之间存在一对一的关系。其特点是开始结点和终端结点都是唯一的，除了开始结点和终端结点以外，其余结点都有且仅有一个直接前驱和一个直接后继。

(3)树形结构。结构中的数据元素是有层次的，数据元素之间存在一对多的关系。特点是每个结点最多只有一个直接前驱，但可以有多个直接后继，只有一个开始结点，但可以有多个终端结点。

(4)图形结构。结构中的数据元素之间存在多对多的关系,也称为网状结构。其特点是每个结点的直接前驱和直接后继的个数可以有多个。可能没有开始结点和终端结点,也有可能有多个开始结点和多个终端结点。

图 1-2 为上述 4 种基本结构的关系图。由于“集合”是元素之间关系极为松散的一种结构,因此也可用其他结构来表示它。



(a) 集合结构　(b) 线性结构　(c) 树形结构　(d) 图形结构

图 1-2　四种基本数据结构

上述 4 种基本的结构关系可分为两大类:线性结构(linear structure)和非线性结构(non-linear structure)。我们把除了线性结构以外的几种结构关系(树、图和集合)都归入非线性结构一类。

从上面所介绍的数据结构的概念中可以知道,一个数据结构有两个要素:一个是数据元素的集合,另一个是关系的集合。在形式上,数据结构通常可以采用一个二元组来表示。

数据结构的形式定义为:数据结构是一个二元组

Data_Structure=(D,S)

其中,D 是数据元素的有限集,S 是 D 上关系的有限集。

$$D=\{D_i \quad I \quad 1\leqslant i\leqslant n, n\geqslant 0\}$$

$$S-\{S_j\ I \quad l\leqslant j\leqslant m, m\geqslant O\}$$

其中,D_i 表示集合 D 中的第 i 个结点或数据元素。n 为 D 中结点的个数,特别地,若 $n=0$,则 K 是一个空集,因而 B 也就无结构可言,有时也可以认为它具有任一结构。S_j 表示集合 S 中的第 j 个二元关系(后面均简称关系)。m 为 S 中关系的个数,特别地,若 $m=0$,则 S 是一个空集,表明集合 D 中的元结点间不存在任何关系,彼此是独立的。

3. 存储结构

数据的逻辑结构是面向应用问题的,是从用户角度看到的数据的结构。数据必须在计算机内存储,数据的存储结构(storage structure)是数据在计算机内的组织方式,是逻辑数据的存储映像,它是面向计算机的。

我们知道,计算机存储器(主存)是由有限个存储单元组成的一个连续的存储空间,这些

存储单元或者是字节编址，或者是字编址的。从存储器角度看，存储器中存储的数据都是二进制的数据，但它们可以被机器指令解释成为指令、整数、实数、字符、逻辑值等，也可以被数据结构的算法解释成为具有某种结构的数据。

在以后的讨论中，在不会引起混淆的场合下，我们可以混合使用结点和元素这两个术语。但在必要时，我们将包括位置信息在内的存储块整体称为结点，而将其中的元素信息部分称为该结点的元素。

最常用的存储结构有顺序(sequential)存储结构、链式(linked)存储结构、索引(index)存储结构和散列(hash)存储结构。

(1)顺序存储结构。顺序存储结构是把逻辑上相邻的元素存储在物理位置相邻的存储单元中，由此得到的存储表示称为顺序存储结构。顺序存储结构是一种最基本的存储表示方法，通常借助于程序设计语言中的数组来实现。

顺序存储结构的主要优点是节省存储空间，因为分配给数据的存储单元全用于存放结点的数据值，数据元素之间的逻辑关系没有占用额外的存储空间。主要缺点是不便于修改，因为在对元素进行插入、删除等运算时，可能要移动一系列的元素。

(2)链式存储结构。链式存储结构对逻辑上相邻的元素不要求其物理位置相邻，元素间的逻辑关系通过附设的指针字段来表示，由此得到的存储表示称为链式存储结构。链式存储结构通常借助于程序设计语言中的指针类型来实现。

链式存储结构的主要优点是便于修改。与顺序存储结构相比，链式存储结构的主要缺点是存储空间的利用率较低。另外，由于逻辑上相邻的结点在存储器中不一定相邻，因此，在用这种方法存储的线性结构中不能对结点进行随机访问。

(3)索引存储结构。索引存储结构是在存储结点信息的同时，还建立附加的索引表。

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)

关键字唯一标识一个结点，地址作为指向结点的指针。对于线性结构来说，各结点的地址在索引表中是按结点的序号依次排列的。

在进行关键字查找时，可以先在索引表中快速查找到相应的关键字，然后通过地址找到结点表中对应的结点。

索引存储结构的优点是线性结构采用索引存储后，可以对结点进行随机访问。索引存储结构的缺点是为建立索引表而增加了时间和空间的开销。

(4)散列存储结构。散列存储结构是根据结点的值确定结点的存储地址。具体做法是：以结点中某个字段的值为自变量，通过某个函数(称为散列函数)计算出对应的函数值 i，再把 i 当作结点的存储地址。例如：

h(key) =key mod p

散列存储的优点是查找速度快，只要给出待查找结点的数值，就有可能立即算出结点的存储地址。但是，散列存储方法只存储结点的数值，不存储结点与结点之间的逻辑关系。散列存储方法一般只用于要求对数据能够进行快速查找、插入的场合。

这些基本的存储表示方法以及它们的组合，可用于实现数据的多种存储结构。

1.1.3 数据结构研究的内容

研究数据结构是为了解决应用问题,所以讨论数据结构必须同时讨论在数据结构上执行的相关运算及其算法才有意义。通过对运算及其算法的性能分析和讨论,使得我们在求解应用问题时,能选择和设计适当的数据结构,编写出高效的程序。

数据和操纵数据的运算是研究数据结构不可分割的两个方面,所以我们在讨论数据结构时,不但要讨论数据的逻辑结构、存储结构,还要讨论在数据结构上执行的运算(operation),以及实现这些运算的算法(algorithm)。

数据运算就是施加于数据的操作。数据运算包括运算的定义和运算的实现,前者确定运算的功能,后者是在存储结构上确定对应运算算法。

在数据结构中,运算不仅仅是加减乘除这些算术运算,还常常涉及算法问题(如常见的有插入、删除、更改、查找和排序等)。算法的实现与数据的存储结构密切相关。

1.2 抽象数据类型的表示与实现

1.2.1 数据类型

在用高级程序语言编写的程序中,必须对程序中出现的每个变量、常量或表达式,明确说明它们所属的数据类型。因为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,以及在这些之上允许进行的操作。

定义:一组性质相同的值的集合,以及定义于这个值集合上的一组操作的总称。

C 语言中的数据类型

char	int	float	double	void
字符型	整型	浮点型	双精度型	无值

构造数据类型由基本数据类型或构造数据类型组成。

基本数据类型可以看作是计算机中已实现的数据结构。

数据类型就是数据结构,不过它是从编程者的角度来使用的。

数据类型是模板,必须定义属于某种数据类型的变量,才能参加运算。

1.2.2 抽象数据类型

1. 数据抽象

抽象可以被理解为一种机制,其实质是抽取共同的和实质的东西,忽略非本质的细节。抽象可以使我们的求解问题过程以自顶向下的方式分步进行:首先考虑问题的最主要方面,然后再逐步细化,进一步考虑问题的某些细节,并最终实现之。

在程序设计中,抽象机制被用于两个方面:数据和过程。数据抽象使程序设计者可以将数据元素间的逻辑关系与数据在计算机内的具体表示分别考虑。过程抽象可使程序设计者将在数据上定义的运算与实现这些运算的具体方法分开考虑。抽象的好处在于降低了问题

求解的难度。

封装通常是指把数据和操纵数据的运算组合在一起的机制。使用者只能通过一组允许的运算访问其中的数据。从某种意义上说,数据的使用者只需知道这些运算的规范(定义)便可访问数据,而无须了解数据是如何组织和存储的,以及这些运算的具体算法如何等与实现有关的细节。也就是说对使用者隐藏了实现的细节。这种程序设计的策略称为信息隐蔽。

我们通常将数据和操纵数据的运算组成模块,每个模块有一个明确定义的接口,模块内部信息只能经过这一接口被外部访问。一个模块的接口是实现运算的一组函数。这样的模块被称为黑盒子。一个程序使用一个采用信息隐蔽原则设计的模块被称为该模块的客户。

封装和信息隐蔽有助于降低问题求解的复杂性,提高程序的可靠性。

2. 抽象数据类型

我们已熟悉的一类 C/C++语言常用的基本数据类型,包括字符型、整型、实型、指针类型等。这些数据类型的值是不可分割的。现实世界最终可以用这些基本数据类型来表示。另一类是结构类型。结构类型的数据的值是由若干成分按某种结构组成的,因此是可以分解的。

数组和结构体是 C/C++语言提供的两种非常有效的组织数据的机制。

【例 1-1】 int data[50] 定义了 50 个整数的数组,其下标范围为 0～49。结构体需要显式定义。

```
struct student{
int student—id;
char name[20];
char sex;
int age;
};
```

数据类型是数据抽象的一种方式。一个数据类型定义了一个值的集合以及作用于该值集的运算的集合。程序设计语言中,数据类型不仅规定了该类型的变量(或常量)的取值范围,还定义了该类型允许的运算。

例如:C++类型为 int(32 位)的变量的取值范围是 -2^{31}～$(2^{31}-1)$,在整型数上的运算有“+”“一”“－”“/”“%”,关系运算“<”“>”“<一”“>一”“一一”“! 一”,赋值运算“一”,等等。

抽象数据类型复数的定义:

```
ADT Complex {
数据对象:D={e1,e2|e1,e2∈RealSet }
数据关系:R1={<e1,e2> | e1 是复数的实数部分,| e2 是复数的虚数部分 }
基本操作:
InitComplex( &Z, v1, v2 )
```

操作结果:构造复数 Z,其实部和虚部分别被赋予参数 v1 和 v2 的值。

DestroyComplex(&Z)

操作结果:复数 Z 被销毁。

GetReal(Z, &realPart)

初始条件:复数已存在。

操作结果:用 realPart 返回复数 Z 的实部值。

GetImag(Z, &ImagPart)

初始条件:复数已存在。

操作结果:用 ImagPart 返回复数 Z 的虚部值。

Add(z1,z2, &sum)

初始条件:z1,z2 是复数。

操作结果:用 sum 返回两个复数 z1,z2 的和值。

} ADT Complex

抽象数据类型可用(D,S,P)三元组表示

其中,D 是数据对象,S 是 D 上的关系集,P 是对 D 的基本操作集。

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〉

} ADT 抽象数据类型名

其中,数据对象和数据关系的定义用伪码描述,基本操作的定义格式为

基本操作名(参数表)

初始条件:〈初始条件描述〉

操作结果:〈操作结果描述〉

基本操作有两种参数:赋值参数只为操作提供输入值;引用参数以 & 打头,除可提供输入值外,还将返回操作结果。

“初始条件”描述了操作执行之前数据结构和参数应满足的条件,若不满足,则操作失败,并返回相应出错信息。

“操作结果”说明了操作正常完成之后,数据结构的变化状况和应返回的结果。若初始条件为空,则省略之。

了解一个数据类型的数据对象(变量、常量)在计算机内的表示是有用的,但也是危险的。这使得应用程序可直接编写算法操纵该数据对象,但一旦改变该对象的存储表示,则必须改变所有使用该对象的程序。目前,普遍认为对使用者隐藏一个数据类型的对象的表示是个好的设计策略,即用户只能通过使用该类型提供的函数操纵该对象。这样,当数据对象的表示或实现函数的算法改变时,只要不改变函数的调用方式,应用程序将无须改变。

一个抽象数据类型(abstract data type,ADT)是一个数据类型,其主要特征包括该类型的数据对象及其运算的规范,它与数据对象的表示和运算的实现分离,实行封装和信息隐蔽。在一个抽象数据类型上应当定义哪些运算,这取决于应用。若一组运算是完备的,那么