



应用型本科信息大类专业“十二五”规划教材

数据结构

韩桂华 程桂卿 主 编



华中科技大学出版社

<http://www.hustp.com>

应用型本科信息大类专业“十二五”规划教材

数 据 结 构

主 编	韩桂华	程桂卿	
副主编	王海文	李香菊	孟桂英
	赵志鹏	陈 灯	李登辉



华中科技大学出版社
中国·武汉

内 容 介 绍

本书系统地介绍了各种类型的数据结构,并且对在每一种数据结构上定义的抽象数据类型都详细地阐述了其基本概念和具体的实现方法,对每个算法的具体实现给出了完整的C语言源代码描述。

本书内容丰富,逻辑性强,层次分明,概念准确,文字简练,对算法的注解十分细致,既注重了理论知识,又强调了工程实用。

为了方便教学,本书还配有教学课件等教学资源包,任课教师和学生可以登录“我们爱读书”网(www.ibook4us.com)免费注册下载,或者发邮件至 hustpeiit@163.com 免费索取。

本书可作为高等院校计算机及相关专业“数据结构”课程的教材,也可作为从事计算机工程与应用相关领域从业人员的参考用书。

图书在版编目(CIP)数据

数据结构/韩桂华 程桂卿 主编. —武汉:华中科技大学出版社, 2013.9
应用型本科信息大类专业“十二五”规划教材
ISBN 978-7-5609-8946-4

I. 数… II. ①韩… ②程… III. 数据结构-高等学校-教材 IV. TP311.12

中国版本图书馆 CIP 数据核字(2013)第 102788 号

数据结构

韩桂华 程桂卿 主编

策划编辑:康 序

责任编辑:康 序

封面设计:李 嫒

责任校对:马燕红

责任监印:张正林

出版发行:华中科技大学出版社(中国·武汉)

武昌喻家山 邮编:430074 电话:(027)81321915

录 排:武汉正风天下文化发展有限公司

印 刷:武汉市首壹印务有限公司

开 本:787mm×1092mm 1/16

印 张:15

字 数:384千字

版 次:2013年9月第1版第1次印刷

定 价:35.00元



本书若有印装质量问题,请向出版社营销中心调换
全国免费服务热线:400-6679-118 竭诚为您服务
版权所有 侵权必究

只有无知，没有不满。

Only ignorant, no resentment.

.....迈克尔·法拉第(Michael Faraday)

迈克尔·法拉第(1791—1867)：英国著名物理学家、化学家，在电磁学、化学、电化学等领域都作出过杰出贡献。

应用型本科信息大类专业“十二五”规划教材

编 审 委 员 会 名 单

(按姓氏笔画排列)

卜繁岭	于惠力	方连众	王书达	王伯平	王宏远
王俊岭	王海文	王爱平	王艳秋	云彩霞	尼亚孜别克
厉树忠	卢益民	刘仁芬	朱秋萍	刘 锐	刘黎明
李见为	李长俊	张义方	张怀宁	张绪红	陈传德
陈朝大	杨玉蓓	杨旭方	杨有安	周永恒	周洪玉
姜 峰	孟德普	赵振华	骆耀祖	容太平	郭学俊
顾利民	莫德举	谈新权	富 刚	傅妍芳	雷升印
路兆梅	熊年禄	霍泰山	魏学业	鞠剑平	

前言

PREFACE

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,对于计算机专业的学生来说,高级语言程序设计、数据结构、算法设计与分析是三门重要的必修课。这三门课程都是教学生如何去编写计算机程序,在学习编写计算机程序的过程中,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。数据结构是程序设计的中级课程,它主要用于培养学生分析数据、组织数据的能力,让学生掌握如何编写效率高、结构好的程序。这便是数据结构课程中将要学习和研究的内容,也是编写计算机程序的重要基础。随着计算机科学技术及其应用的飞速发展,计算机学科的许多领域都发生了根本的变化,因此,数据结构的教学内容和教学方法也在不断更新。但是数据结构课程在计算机技术教育中的地位和作用始终没有改变,所以本课程被列为计算机等相关专业重要的专业基础课程。

本书遵循数据结构课程的教学大纲的要求,详细介绍了各种基本类型的数据结构。第1章介绍了数据结构研究的内容和方法,以及算法和算法分析的基本概念;第2章介绍了线性表;第3章介绍了栈和队列;第4章介绍了串,讨论的是线性结构;第5章介绍了数组及其应用;第6章和第7章分别讨论了树形结构和图状结构;第8章和第9章着重分析了查找技术和排序技术;第10章介绍了文件组织和管理方面的内容。

本书以深入浅出、突出重点、讲深讲透为原则,精选了各章的内容,特别是各章中的应用实例。书中还注重融入抽象数据类型的思想,在讨论一种数据结构时,都采用抽象数据类型的观点进行描述,以便让学生更好地掌握理论知识,提高抽象思维的能力;另一方面,对于数据结构和算法的描述,本书没有使用伪代码,而是采用了大多数学生熟悉的C语言进行描述,每个算法都用C语言的函数形式给出。希望本书能够让学生对数据结构的研究内容有较系统了解,并且能够让学生初步掌握数据结构的研究方法。

学习数据结构的目的是编写高效率的程序。因此,学习本课程的过程也是

进行复杂程序设计的训练过程,所以请读者注意上机实习环节。

参与本书编写的人员都是高等院校计算机专业具有丰富教学经验的教师。其中,由程桂卿(湖北工业大学商贸学院)负责全书的统稿工作;第1章和第2章由韩桂华(湖北工业大学商贸学院)编写,第3章由陈灯(湖北工业大学商贸学院)编写,第4章和第5章由孟桂英(沈阳航空航天大学北方科技学院)编写,第6章由王海文(大连工业大学艺术与信息工程学院)编写,第7章由李香菊(东南大学成贤学院)编写,第8章和第10章由赵志鹏(桂林电子科技大学信息科技学院)编写,第9章由李登辉(桂林电子科技大学信息科技学院)编写。

为了方便教学,本书还配有教学课件等教学资源包,任课教师和学生可以登录“我们爱读书”网(www.ibook4us.com)免费注册下载,或者发邮件至 hustpeiit@163.com 免费索取。

由于编者水平有限,书中难免出现错误和不足,恳请广大读者批评指正。

编 者

2013年5月

目
录

CONTENTS

第 1 章 绪论	(1)
1.1 基本术语	(1)
1.2 数据结构的内容	(2)
1.3 算法描述	(5)
1.4 算法分析	(9)
习题 1	(12)
第 2 章 线性表	(14)
2.1 线性表的定义及运算	(14)
2.2 线性表的顺序存储结构	(15)
2.3 线性表的链式存储结构	(20)
2.4 循环链表	(27)
2.5 双向链表	(28)
2.6 多项式相加	(31)
习题 2	(34)
第 3 章 栈和队列	(36)
3.1 栈的定义及抽象数据类型	(36)
3.2 栈的实现	(37)
3.3 栈的应用举例	(42)
3.4 队列的定义及抽象数据类型	(44)
3.5 队列的实现	(45)
3.6 队列的应用举例	(52)
习题 3	(56)
第 4 章 串	(58)
4.1 串定义	(58)

4.2 串的存储方法	(59)
4.3 串操作	(61)
习题4	(68)
第5章 矩阵与广义表	(69)
5.1 矩阵	(69)
5.2 特殊矩阵	(72)
5.3 稀疏矩阵的运算	(78)
5.4 广义表	(85)
5.5 广义表的运算	(89)
习题5	(95)
第6章 树	(96)
6.1 树的定义和术语	(96)
6.2 二叉树	(97)
6.3 遍历二叉树和线索二叉树	(104)
6.4 二叉树的转换	(111)
6.5 二叉树的应用	(116)
6.6 哈夫曼树及其应用	(120)
习题6	(125)
第7章 图	(131)
7.1 图的定义与基本术语	(131)
7.2 图的存储结构	(136)
7.3 图的遍历和生成树	(142)
7.4 最小生成树	(150)
7.5 最短路径	(157)
7.6 拓扑排序	(162)
7.7 关键路径	(165)
习题7	(167)
第8章 查找	(170)
8.1 顺序查找	(170)
8.2 折半查找	(172)
8.3 分块查找	(174)
8.4 树型查找	(175)
8.5 散列查找	(188)
习题8	(195)
第9章 内排序	(197)
9.1 排序的基本概念	(197)
9.2 直接插入排序	(198)

9.3 简单选择排序	(200)
9.4 起泡排序	(202)
9.5 堆排序	(204)
9.6 快速排序	(209)
9.7 归并排序	(212)
9.8 基数排序	(215)
9.9 7 种排序方法的比较	(218)
习题 9	(219)
第 10 章 文件	(221)
10.1 概述	(221)
10.2 文件组织	(222)
10.3 动态索引	(225)
习题 10	(229)
参考文献	(230)

第1章 绪 论

计算机的应用领域非常广泛,如科学计算、情报检索、信息处理等,而它的功能归根结底只有一个,即数据处理。

计算机所处理的数据不是杂乱无章的,而是相互之间有着某种内在联系。为了能合理的组织数据和高效的处理数据,故必须弄清楚数据的内在联系。

本章主要介绍数据结构课程中的一些常用术语,几种常见数据类型的表示,以及算法时间复杂度和空间复杂度的分析。

1.1 基本术语

首先介绍数据结构中的几个常见术语。

1. 数据

在计算机科学中,数据(data)的定义很广泛,它指一切能被计算机识别和处理的符号。即数据是描述客观事物的数值、字符,以及能输入机器且能被处理的各种符号的集合。例如,数字、字母、图形、图像、声音等都称为数据。

2. 数据元素

数据元素(data element)是组成数据的基本单位,是数据集合的个体,在计算机中通常作为一个整体进行考虑和处理,因而数据元素也称为结点。

【例 1-1】 学生成绩表是数据,则每一个学生的记录就是一个数据元素,如表 1-1 所示。

一般情况下,一个数据元素中包含若干个字段(也称为数据项),字段(数据项)是具有独立含义的最小单位,此时的数据元素也称为记录。表 1-1 中每一个学生的记录就是一个数据元素,每个数据元素由学号、姓名、英语、高数和 C 语言程序设计 5 个字段(数据项)构成。

表 1-1 学生成绩表

学 号	姓 名	英 语	高 数	C 语言程序设计
20081513101	张玲	85	79	92
20081513102	李伟	76	84	88
20081513103	王青	82	64	75
20081513104	林雪	81	80	68
.....				

3. 数据对象

数据对象(data object)是性质相同的数据元素的集合,是数据的一个子集。例如,整数数据对象的集合可表示为 $N=\{0,\pm 1,\pm 2,\pm 3,\dots\}$,大写字符数据对象的集合可表示为 $C=\{'A','B','C',\dots,'Z'\}$,表 1-1 也是一个数据对象。由此可知,不管数据元素的集合是有限集还是无限集,只要数据元素的性质相同,就属于同一个数据对象。

4. 数据结构

数据结构(data structure)是指相互之间存在一种或多种特定关系的数据元素的集合,是带有结构的数据元素的集合,它指的是数据元素之间的相互关系,即数据的组织形式。

5. 数据类型

数据类型(data type)是一组性质相同的值集合,以及定义在这个值集合上的一组操作的总称。例如,高级语言中用到的整型数据类型(以2字节为例),是指由-32768~32767中的整型数构成的集合及一组操作(如加、减、乘、除、乘方等)的总称。

6. 抽象数据类型

抽象数据类型(abstract data types,简称 ADT)通常是指由用户定义的、用于表示应用问题的数据模型,抽象数据类型由基本数据类型组成,并包括一组相关的操作。抽象数据类型类似于 C/C++ 语言中的结构体类型,但它增加了相关的操作。

【例 1-2】 给出自然数(natural number)的抽象数据类型定义。

```
ADT Natural Number
```

```
{
```

数据元素:所有数据元素属于同一数据对象,整型。

结构关系:一个整数的有序子集,它开始于 1,终止于机器所能表示的最大整数。

基本操作:对于所有 $x, y \in \text{Natural Number}$, 定义如下操作。

add(x, y):求 $x+y$ 。

sub(x, y):求 $x-y$ 。

mul(x, y):求 $x \times y$ 。

div(x, y):求 $x \div y$ 。

equal(x, y):判断 x, y 是否相等。

```
}ADT Natural Number
```

本书中,描述一种抽象数据类型采用如下书写格式。

ADT <ADT 名>

```
{
```

数据对象:<数据对象的定义>

结构关系:<结构关系的定义>

基本操作:<基本操作的定义>

```
}ADT <ADT 名>
```

1.2 数据结构的内容

数据结构是指相互之间存在一种或多种特定关系的数据元素所组成的集合。具体来说,数据结构包括三方面的内容,即数据的逻辑结构、数据的存储结构和对数据所施加的运算。这三个方面的关系如下。

- 数据的逻辑结构独立于计算机,是数据本身所具有的。
- 数据的存储结构是逻辑结构在计算机存储器中的映像,必须依赖于计算机。
- 运算是指所施加的一组操作的总称。运算的定义直接依赖于逻辑结构,但运算的实现必须依赖于存储结构。

1. 逻辑结构

数据的逻辑结构是指数据的直接逻辑关系。大多数情况下,数据的结构可以用二元组表示:

$$S=(D,R)$$

其中, D 表示数据元素的集合; R 是 D 上关系的有限集,表示 D 上的一种关系。

【例 1-3】二元组 $L=(D,R_1)$, $T=(D,R_2)$, $G=(D,R_3)$ 分别描述了 5 个结点的三种不同的逻辑结构,分别如下。

$$D=\{a,b,c,d,e\}$$

$$R_1=\{\langle a,b\rangle,\langle b,c\rangle,\langle c,d\rangle,\langle d,e\rangle\}$$

$$R_2=\{\langle a,b\rangle,\langle b,c\rangle,\langle b,d\rangle,\langle b,e\rangle\}$$

$$R_3=\{\langle a,b\rangle,\langle b,c\rangle,\langle d,a\rangle,\langle c,e\rangle,\langle e,d\rangle,\langle d,c\rangle\}$$

则 R_1, R_2, R_3 的逻辑结构分别如图 1-1 所示。

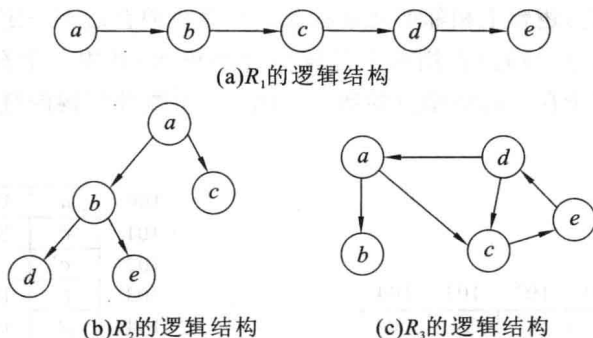


图 1-1 逻辑结构示意图

根据数据元素之间关系的不同特性,通常有下列四种基本结构,如图 1-2 所示。

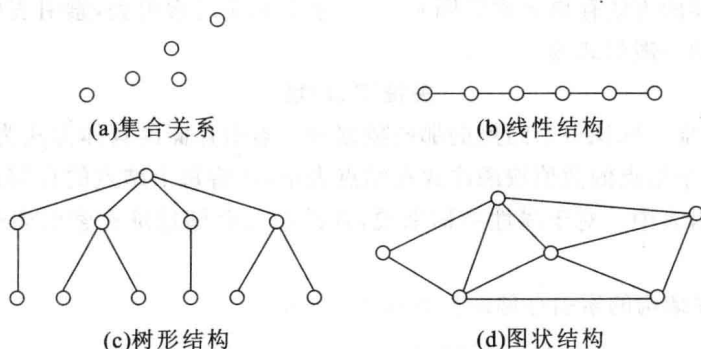


图 1-2 四种基本数据结构示意图

(1) 集合关系:该结构中的数据元素之间除了同属于一个集合的关系外,无任何其他关系。

(2) 线性结构:该结构中的数据元素之间存在着一对一的线性关系。

(3) 树形结构:该结构中的数据元素之间存在着一对多的层次关系。

(4) 图状结构或网状结构:该结构中的数据元素之间存在着多对多的任意关系。

由于集合关系中只有属于或不属于这种简单的属于关系,故可以用其他的结构代替。因此,数据的四种基本逻辑结构可概括如下。

逻辑结构 { 线性结构——线性表、栈、队、字符串、数组、广义表
非线性结构——树、图

2. 存储结构

存储结构(又称物理结构)是逻辑结构在计算机中的存储映象,是逻辑结构在计算机中的实现,它包括数据元素的表示和关系的表示。数据结构从存储结构上划分为以下两种形式。

1) 顺序存储

顺序存储(向量存储)是所有数据元素存放在一片连续的存储单元中,逻辑上相邻的元素存放到计算机内存中仍然相邻。

假设每个数据元素占用一个存储单元,数据从 100 号单元开始由低地址向高地址存放,则例 1-3 中线性结构的顺序存储结构如图 1-3 所示。

100	101	102	103	104
a	b	c	d	e

图 1-3 顺序存储结构

100	a	103
101	c	104
102	e	
103	b	101
104	d	102

图 1-4 链式存储结构

3) 索引存储

使用索引存储的方法存储元素的同时,还需建立相应的索引表,索引表中的每一项称为索引项。索引项的一般形式为:

(关键字、地址)

其中,关键字是能唯一标识一个结点的那些数据项。索引存储的具体方法为:①建立索引表和结点表;②将每个结点的数值按顺序放在结点表中,而将每个结点的存储地址则用顺序存储方法存放在索引表中。对于线性结构来说,各结点的存储地址在索引表中是按结点的序号依次排列的。

例 1-3 中线性结构的索引存储结构如图 1-5 所示。

200	100	100	a
201	103	101	c
202	101	102	e
203	104	103	b
204	102	104	d

图 1-5 索引存储结构

4) 散列存储

散列存储方式通过构造散列函数,用函数的值来确定数据元素存放的地址。散列存储的具体方法为:以结点中某个字段的值作为自变量,通过某个函数(称为散列函数)计算出对应的函数值 i ,再将 i 作为结点的存储地址。

将例 1-3 中线性结构改为散列存储结构,可选用如下函数来计算各结点的地址。

$$\text{LOC}(k) = k \% 7$$

其中, k 为各结点数值的 ASCII 值,计算结果如下。

a	b	c	d	e
6	0	1	2	3

于是,可以得到如图 1-6 所示的散列存储结构。

b	c	d	e			a
0	1	2	3	4	5	6

图 1-6 散列存储结构

同一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,应视具体要求而定,主要需考虑运算方便及算法的时空要求。

3. 运算集合

讨论数据结构的目的是为了在计算机中实现操作,因此,在结构上的运算集合是很重要的部分,数据结构就是用于研究某一类数据的表示及其相关的运算操作。例如,表 1-1 的学生成绩表,数据元素之间采用顺序存放,该表采用的是线性表的逻辑结构。因为学生成绩表可以有成千上万名学生,既可以采用顺序存放,也可以采用非顺序存放,具体采用何种方式,这要看具体问题来分析。对于学生成绩表来说,若学生情况有异动,则可以对表格进行添加、删除、修改等操作。这里的增、删、查、改就是数据的集合操作。

4. 数据结构三方面的关系

数据的逻辑结构、数据的存储结构及数据的运算这三方面是一个整体。如果孤立地去理解其中某一个方面,而不注意它们之间的联系是不可取的。

存储结构是数据结构不可缺少的一个方面,同一逻辑结构的不同存储结构可以冠以不同的数据结构名称来标识。

【例 1-4】 线性表是一种逻辑结构,若采用顺序存储方法,可称其为顺序表;若采用链式存储方法,则可称其为链表;若采用散列存储方法,则可称其为散列表。

数据的运算也是数据结构中不可分割的一个方面。在给出了数据的逻辑结构和存储结构之后,若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 1-5】 若对线性表上的插入、删除运算限制在表的一端进行,则该线性表称之为栈;若将插入运算限制在表的一端进行,而删除运算限制在表的另一端进行,则该线性表称之为队列。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,以及顺序队列或链队列。



1.3 算法描述

著名计算机科学家 N. Wirth 曾给出过如下公式

$$\text{程序} = \text{算法} + \text{数据结构}$$

这个公式表明,数据结构和算法是程序的两大要素,二者相辅相成,缺一不可。

对于任何一种运算,如果数据的存储结构不同,则其算法描述一般是不同的,即使在存

储结构相同的前提下,由于可以采用不同的求解策略,往往也可以有很多不同的算法。那么,如何选择一个好算法呢?首先,需要理解算法。

1.3.1 基本概念

1. 算法的定义

算法(algorithm)是规则的有限集合,是为了解决特定问题而规定的一系列操作。

2. 算法的特性

- (1) 有限性 算法在有限步骤之内应正常结束,不能形成无穷循环。
- (2) 确定性 算法中的每一个步骤必须有确定含义,使得无二义性得以实现。
- (3) 输入性 有多个或0个输入。
- (4) 输出性 至少有一个或多个输出。
- (5) 可行性 原则上能精确进行,操作上可通过已实现的基本运算执行有限次而完成。

在算法的五大特性中,最基本的是有限性、确定性和可行性。

3. 算法设计的要求

算法设计一般应该具有以下几个基本特征。

1) 算法的正确性

算法的正确性是指算法应该满足具体问题的需求,其中“正确”的含义包括以下三个方面。

- (1) 算法对几组不同的输入数据能够得出满足要求的结构。
- (2) 算法对于精心选择的典型、苛刻而带有刁难性的输入数据能够得出满足要求的结果。
- (3) 算法对于一切合法的输入数据都产生满足要求的结果。

2) 可读性

算法主要是为了方便程序员的阅读与交流,其次才是便于机器执行。可读性好有利于程序员对算法的理解;晦涩难懂的程序易隐藏较多错误,难以调试和修改。

3) 健壮性

健壮性即指算法对于非法输入的抵抗能力,它强调了即使输入了非法数据,算法应能够识别并做出正确处理,而不是产生莫名其妙的输出结果。例如,一个求凸多边形面积的算法是采用求各三角形面积之和的策略来解决问题,当输入的坐标值集合表示的是一个凹多边形时,则不应继续计算,而应报告输入出错。并且,处理错误的方法是返回一个表示错误或错误性质的值,而不是打印错误信息或异常,并同时终止程序的执行,以便在更高的抽象层次上进行处理。

4) 高效率和低存储量

算法的效率通常是指算法的执行时间。对于一个具体的问题,通常有多个算法可以解决,执行时间短的算法其效率就高。所谓存储量,是指算法在执行过程中所需要的最大存储空间。效率与低存储量需求这两者与问题的规模有关。

1.3.2 算法描述的工具

1. 流程图

用流程图描述算法符合人们的思维习惯,直观形象,并且易于理解。

2. 自然语言

用自然语言描述算法通俗易懂,特别适用于对顺序程序结构算法的描述。在使用时,要特别注意算法逻辑的正确性。

3. 其他方法

还可以用数学语言或约定的符号语言来描述算法。

4. 计算机语言

由于是使用计算机实现算法,因此,可以使用计算机语言来表示算法,使用计算机语言描述算法时必须严格遵循所用语言的语法规则。本书中采用的是C语言描述算法。

1.3.3 C语言的语法结构简介

C语言的语法结构主要有以下几个部分。

(1) 预定义常量和类型。

本书中将用到的常量符号有 TRUE、FALSE、MAXSIZE 等,约定用如下的宏定义预先定义。

```
#define TRUE 1
#define FALSE 0
#define MAXSIZE 100
```

(2) 本书中所有的算法都以如下的函数形式加以表示,其中的结构类型使用前面已有的定义。

[数据类型] 函数名([形式参数及说明])

{

内部数据说明;

执行语句组;

} /* 函数名 */

函数的定义主要由函数名和函数体组成,函数体用花括号“{”和“}”括起来。函数中用方括号括起来的部分,如“[形式参数]”为可选项,函数名之后的圆括号不可省略。

(3) 赋值语句。

① 简单赋值语句。

• $\langle \text{变量名} \rangle = \langle \text{表达式} \rangle$, 它表示将表达式的值赋给左边的变量。

• $\langle \text{变量} \rangle ++$, 它表示变量加 1 后赋值给变量。

• $\langle \text{变量} \rangle --$, 它表示变量减 1 后赋值给变量。

② 串联赋值语句。

$\langle \text{变量} 1 \rangle = \langle \text{变量} 2 \rangle = \langle \text{变量} 3 \rangle = \dots = \langle \text{变量} k \rangle = \langle \text{表达式} \rangle$;

③ 成组赋值语句。

$(\langle \text{变量} 1 \rangle, \langle \text{变量} 2 \rangle, \langle \text{变量} 3 \rangle, \dots, \langle \text{变量} k \rangle) = (\langle \text{表达式} 1 \rangle, \langle \text{表达式} 2 \rangle, \langle \text{表达式} 3 \rangle, \dots, \langle \text{表达式} k \rangle)$;

$\langle \text{数组名} 1 \rangle [\text{下标} 1] [\text{下标} 2] = \langle \text{数组名} 2 \rangle [\text{下标} 1] [\text{下标} 2]$;

④ 条件赋值语句。

$\langle \text{变量名} \rangle = \langle \text{条件表达式} \rangle ? \langle \text{表达式} 1 \rangle : \langle \text{表达式} 2 \rangle$;

(4) 条件选择语句,有以下几种形式。