21世纪

中国高等职业技术教育研究会推荐
高职高专计算机专业规划教材

数据结构
（C语言版）

主编　张群哲
主审　刘喜勋

西安电子科技大学出版社
http://www.xduph.com

□ 中国高等职业技术教育研究会推荐

高职高专计算机专业规划教材

数据结构

(C语言版)

主　编　张群哲

主　审　刘喜勋

西安电子科技大学出版社

2008

内 容 简 介

本书是针对高职高专院校计算机和相关专业“数据结构”课程的特点而编写的。本书详细介绍了数据结构的基本概念、基本结构和算法等重要内容。为突出高职教学特点，全书共安排了与教学进度相配合的16个实训指导，在附录A中提供了两个课程设计指导。每章之后还配有丰富的习题，在附录B中还提供了部分习题的参考答案，以利于读者理解课本内容和适应考试。

本书结构严谨、重点突出、通俗易懂，注重实践能力培养，既便于教学又便于自学。本书可作为高职高专院校计算机专业及相关专业的教材，对从事计算机应用的工程技术人员也是一本很有价值的参考书。

★ 本书配有电子教案，有需要的老师可以登录出版社网站，免费下载。

图书在版编目(CIP)数据

数据结构：C语言版 / 张群哲主编. —西安：西安电子科技大学出版社，2008.2

中国高等职业技术教育研究会推荐. 高职高专计算机专业规划教材

ISBN 978-7-5606-1974-3

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

中国版本图书馆CIP数据核字(2008)第008139号

策　　划　臧延新

责任编辑　杨宗周

出版发行　西安电子科技大学出版社(西安市太白南路2号)

电　　话　(029)88242885　88201467　　邮　编　710071

http: //www.xduph.com　　E-mail: xdupfxb@pub.xaonline.com

经　　销　新华书店

印刷单位　渭南市邮电印刷厂

版　　次　2008年2月第1版　2008年2月第1次印刷

开　　本　787毫米×1092毫米　1/16　印　张　15

字　　数　350千字

印　　数　1～4000册

定　　价　21.00元

ISBN 978-7-5606-1974-3/TP・1021

XDUP 2266001-1

序

进入21世纪以来，高等职业教育呈现出快速发展的形势。高等职业教育的发展，丰富了高等教育的体系结构，突出了高等职业教育的类型特色，顺应了人民群众接受高等教育的强烈需求，为现代化建设培养了大量高素质技能型专门人才，对高等教育大众化作出了重要贡献。目前，高等职业教育在我国社会主义现代化建设事业中发挥着越来越重要的作用。

教育部2006年下发了《关于全面提高高等职业教育教学质量的若干意见》，其中提出了深化教育教学改革，重视内涵建设，促进“工学结合”人才培养模式改革，推进整体办学水平提升，形成结构合理、功能完善、质量优良、特色鲜明的高等职业教育体系的任务要求。

根据新的发展要求，高等职业院校积极与行业企业合作开发课程，根据技术领域和职业岗位群任职要求，参照相关职业资格标准，改革课程体系和教学内容，建立突出职业能力培养的课程标准，规范课程教学的基本要求，提高课程教学质量，不断更新教学内容，而实施具有工学结合特色的教材建设是推进高等职业教育改革发展的重要任务。

为配合教育部实施质量工程，解决当前高职高专精品教材不足的问题，西安电子科技大学出版社与中国高等职业技术教育研究会在前三轮联合策划、组织编写“计算机、通信电子、机电及汽车类专业”系列高职高专教材共160余种的基础上，又联合策划、组织编写了新一轮“计算机、通信、电子类”专业系列高职高专教材共120余种。这些教材的选题是在全国范围内近30所高职高专院校中，对教学计划和课程设置进行充分调研的基础上策划产生的。教材的编写采取在教育部精品专业或示范性专业的高职高专院校中公开招标的形式，以吸收尽可能多的优秀作者参与投标和编写。在此基础上，召开系列教材专家编委会，评审教材编写大纲，并对中标大纲提出修改、完善意见，确定主编、主审人选。该系列教材以满足职业岗位需求为目标，以培养学生的应用技能为着力点，在教材的编写中结合任务驱动、项目导向的教学方式，力求在新颖性、实用性、可读性三个方面有所突破，体现高职高专教材的特点。已出版的第一轮教材共36种，2001年全部出齐，从使用情况看，比较适合高等职业院校的需要，普遍受到各学校的欢迎，一再重印，其中《互联网实用技术与网页制作》在短短两年多的时间里先后重印6次，并获教育部2002年普通高校优秀教材奖。第二轮教材共60余种，在2004年已全部出齐，有的教材出版一年多的时间里就重印4次，反映了市场对优秀专业教材的需求。前两轮教材中有十几种入选国家“十一五”规划教材。第三轮教材2007年8月之前全部出齐。本轮教材预计2008年全部出齐，相信也会成为系列精品教材。

教材建设是高职高专院校教学基本建设的一项重要工作。多年来，高职高专院校十分重视教材建设，组织教师参加教材编写，为高职高专教材从无到有，从有到优、到特而辛勤工作。但高职高专教材的建设起步时间不长，还需要与行业企业合作，通过共同努力，出版一大批符合培养高素质技能型专门人才要求的特色教材。

我们殷切希望广大从事高职高专教育的教师，面向市场，服务需求，为形成具有中国特色和高职教育特点的高职高专教材体系作出积极的贡献。

中国高等职业技术教育研究会会长 [signature]

2007年6月

高职高专计算机专业规划教材
编审专家委员会

主　任：温希东　（深圳职业技术学院副校长，教授）

副主任：徐人凤　（深圳职业技术学院电子与通信工程学院副院长，高工）

刘中原　（上海第二工业大学计算机与信息学院副院长，副教授）

李卓玲　（沈阳工程学院信息工程系主任，教授）

委　员：（按姓氏笔画排列）

丁桂芝　（天津职业大学电子信息工程学院院长，教授）

马宏锋　（兰州工业高等专科学校计算机工程系副主任，副教授）

王　军　（武汉交通职业学院信息系副主任，副教授）

王　雷　（浙江机电职业技术学院计算机应用工程系主任，高工）

王养森　（南京信息职业技术学院计算机科学与技术系主任，高工）

王趾成　（石家庄职业技术学院计算机系主任，高工）

汤　勇　（成都职业技术学院国际软件学院副院长，副教授）

朱小平　（广东科学技术职业学院计算机学院副院长，副教授）

齐志儒　（东北大学东软信息学院计算机系主任，教授）

孙街亭　（安徽职业技术学院教务处处长，副教授）

张　军　（石家庄职业技术学院计算机系，高工）

李成大　（成都电子机械高等专科学校计算机工程系副主任，副教授）

苏传芳　（安徽电子信息职业技术学院计算机科学系主任，副教授）

苏国辉　（黎明职业大学计算机系副主任，讲师）

汪临伟　（九江职业技术学院电气工程系主任，副教授）

汪清明　（广东轻工职业技术学院计算机系副主任，副教授）

杨文元　（漳州职业技术学院计算机工程系副主任，副教授）

杨志茹　（株洲职业技术学院信息工程系副主任，副教授）

胡昌杰　（湖北职业技术学院计算机科学与技术系副主任，副教授）

聂　明　（南京信息职业技术学院软件学院院长，副教授）

章忠宪　（漳州职业技术学院计算机工程系主任，副教授）

眭碧霞　（常州信息职业技术学院软件学院院长，副教授）

董　武　（安徽职业技术学院电气工程系副主任，副教授）

蒋方纯　（深圳信息职业技术学院软件工程系主任，副教授）

鲍有文　（北京联合大学信息学院副院长，教授）

前　言

“数据结构”是计算机及相关专业的基础课程，在计算机专业课程中起着承前启后的作用，为“操作系统”、“编译原理”、“数据库系统”等后续课程奠定基础。了解数据结构的基本概念、基本结构和基本算法是提高计算机程序设计能力和软件系统设计能力的必要条件。

本书按照高职高专“数据结构”课程教学大纲的要求，详细介绍了数据结构的基本概念、基本理论、各种具体的数据结构，对逻辑结构、物理结构、运算、算法和应用进行了重点讨论，还介绍了很有应用价值的查找和排序方法等。

为适应高职高专教学特点，保证实训教学时数占课程总时数的 50%，本书在每章之后都安排了与教学进度相配合的实训指导。全书共 16 个实训，实训内容与教学内容紧密结合，是对算法的具体实现与深化。同时，在附录 A 中提供了两个课程设计指导，可以更好地培养学生对知识与技能的综合应用能力。每章之后还配有丰富的习题，有利于读者理解知识内容、掌握操作方法，并在附录 B 中提供了部分习题的参考答案。

本书共分为 9 章：

第 1 章介绍数据结构和算法的基本概念、时间复杂度的估算方法；

第 2 章介绍线性表，讨论线性表的概念、运算、顺序存储结构、链式存储结构和在不同存储结构下的基本操作算法；

第 3 章介绍栈和队列，讨论栈和队列的特点及其各种存储结构与基本操作的实现，介绍了栈和队列的应用实例；

第 4 章介绍数组和矩阵，讨论数组的概念、存储、访问，特殊矩阵的存储和数组的应用等；

第 5 章介绍串，讨论串的概念、各种存储结构及其基本操作的实现；

第 6 章介绍树和二叉树，讨论树、二叉树和森林的定义、性质、表示、存储结构以及二叉树的基本操作，讨论哈夫曼树的基本概念及其应用；

第 7 章介绍图，讨论图的概念、术语、存储方式，在顺序存储结构和链式存储结构下的算法，还介绍了图的应用；

第 8 章介绍查找，讨论非常具有实用价值的顺序查找、二分查找、二叉排序树和散列技术的概念、查找方法和算法；

第 9 章介绍排序，讨论插入排序、交换排序、选择排序、归并排序、基数排序等五大类，共九种排序方法的概念和算法。

本书在内容的选取、概念的引入、文字的叙述以及例题、习题、实训的选择等方面充分考虑了目前高职高专学生的知识结构、能力结构和素质结构，力求做到由浅入深、循序

渐进，突出实用性和应用性，注重培养读者分析问题和解决问题的能力，特别适合教学和自学。书中所有的数据结构和算法都采用 C 语言描述，全部算法例题、实训源程序都在 TurboC 2.0 下调试通过。

本书可作为高职高专计算机类专业教材，也可作为计算机专业的成人教育、自学考试和各类培训的教材，对从事计算机应用的工程技术人员也是一本很有价值的参考书。

本书由张群哲任主编，参加编写的还有李正军、高登、董婷和宋剑杰，全书由张群哲统稿和定稿。

由于编者水平有限，书中难免有疏漏之处，敬请读者批评指正。

编　者

2007 年 12 月

目　录

第1章 绪 论

【教学目标】通过对本章的学习，熟悉各名词和术语的含义，理解数据结构及其有关的概念，了解算法的基本特征和算法的评价标准，掌握估算算法的时间复杂度的方法。

运用计算机对一批数据进行处理时，必须解决好三个方面的问题：第一，根据数据之间的逻辑关系如何组织这批数据？第二，如何将这批数据存储在计算机的存储器中？第三，对于存储在计算机中的这批数据可以进行哪些操作？如何实现这些操作？对同一问题的不同操作方法如何进行评价？这些问题就是数据结构这门课程所要研究的主要问题。

1.1 数据结构的概念

1.1.1 基本概念和术语

在阐述数据结构的概念之前，先对数据结构中常用的几个基本概念和术语给出确切的定义。

1. 数据(Data)

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。换句话说，数据是对客观事物采用计算机能够识别、存储和处理的形式所进行的描述。简而言之，数据就是计算机化的信息。

数据一般可分为数值型数据和非数值型数据，如数学中的整数、实数等都是数值型数据，字符、表格、图形、图像和声音等都是非数值型数据。又如对于 C 源程序，数据概念不仅是源程序所处理的数据，源程序本身也是被 C 编译程序处理的数据。编译程序相对于源程序是一个处理程序， 它加工的数据是字符流的源程序(.c)，输出的结果是目标程序(.obj)；对于链接程序来说，它加工的数据是目标程序(.obj)，输出的结果是可执行程序(.exe)，如图 1.1 所示。

源程序 —C编译程序→ 目标程序 —C链接程序→ 可执行程序

图 1.1 编译程序示意图

2. 数据元素(Data Element)

数据元素是数据的基本单位，在计算机程序中通常作为一个整体进行考虑和处理。有时一个数据元素可以由若干个数据项组成，数据项是具有独立含义的最小单位。数据元素

有时也叫做结点、元素、顶点、记录等。例如：在数据库管理系统中，数据库中的一条记录就是一个数据元素，这条记录中的每一个字段就是构成这个数据元素的数据项，如表 1-1 所示。

表 1-1 学 籍 表

学号	姓名	性别	籍贯	出生年月	住址
101	赵红玲	女	河北	1983.11	北京
…	…	…	…	…	…

记录

数据项

3. 数据对象(Data Object)

数据对象是性质相同的数据元素的集合，是数据的一个子集。例如：整数数据对象是集合N={0，±1，±2，…}，字母字符数据对象是集合 C={'A'，'B'，…，'Z'}。表 1-1 所示的学籍表也可看做一个数据对象。由此可看出，不论数据元素集合是无限集(如整数集)、有限集(如字符集)，还是由多个数据项组成的复合数据元素(如学籍表)，只要性质相同，都是同一个数据对象。

4. 数据类型

数据类型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。数据类型中定义了两个集合，即该类型的取值范围，以及该类型中可允许使用的一组运算。例如高级语言中的数据类型就是已经实现的数据结构的实例。从这个意义上讲，数据类型是高级语言中允许的变量种类，是程序语言中已经实现的数据结构(即程序中允许出现的数据形式)。在高级语言中，整型类型可能的取值范围是–32 768～+32 767，可用的运算符集合为加、减、乘、除、乘方、取模(如 C 语言中的 +、–、*、/、%)。

1.1.2 数据结构的定义

对于什么是数据结构，目前还没有一个公认的定义，对数据结构的概念，可以从以下两个角度来理解：一是把数据结构作为一门独立开设的课程，看“数据结构”在计算机专业中的地位、性质、任务以及它研究的内容等；二是把数据结构作为计算机所处理的数据的组织方式来理解。

1. “数据结构”课程

“数据结构”作为一门独立的课程在国外是从 1968 年开始设立的，我国从 1978 年开始，各高等院校才先后开设了这门课程，现在“数据结构”已是计算机专业的一门综合性的专业基础课。它是“操作系统”、“编译原理”、“数据库系统”、“计算机图形学”、“人工智能”等课程的先修课。“数据结构”的研究不仅涉及到计算机硬件的研究范围，而且和计算机软件的研究有着密切的关系，可以说，“数据结构”是介于数学、计算机硬件和计算机软件三者之间的一门综合性课程，可以用一个定义描述如下：“数据结构”是研究非数值计算的程序设计问题中计算机操作对象以及它们的关系和操作的一门学科。

2. 数据结构的概念

如果从数据元素之间的组织形式来看，可以认为数据结构指的是计算机所处理的数据元素之间的组织形式和相互关系。在任何问题中，数据元素都不是孤立存在的，它们之间必定存在着某种内在的或者是根据需要人为定义的关系，这种关系就是这些数据元素的数据结构。

数据结构一般包括以下三个方面的内容：

(1) 数据元素之间的逻辑关系，即数据的逻辑结构，是用户根据需要而建立起来的。

(2) 数据元素及其关系在计算机存储器内的表示，即数据的存储结构，又称数据的物理结构。

(3) 数据的运算，即对数据元素所进行的操作。

可以把数据结构定义为：对计算机处理的一批数据，首先按某种逻辑关系把它们组织起来，再按一定的存储表示方式把它们存储在计算机的存储器中，然后给这批数据规定一组操作。

1.2 数据的逻辑结构和存储结构

1.2.1 逻辑结构

根据数据元素之间的不同特性，可以把数据的逻辑结构分为四种基本类型：集合、线性结构、树形结构和图形结构。其逻辑结构关系如图 1.2 所示。

1. 集合

集合中的数据元素之间除了“同属于一个集合”的关系外，没有其它任何关系。大街上熙熙攘攘的人群可以看成是一个集合。集合是数据结构的一种特例，本书不予讨论。

2. 线性结构

线性结构中的数据元素之间存在一对一的线性关系。即除了第一个元素外，其它的每个元素都有且仅有一个直接前驱，除了最后一个元素外，每个元素都有且仅有一个直接后继。火车站长长的购票队列就是线性结构。

3. 树形结构

树形结构中的数据元素之间存在一对多的层次关系。即每个结点最多只有一个直接前驱，但每个结点都可以有多个直接后继。其中根结点没有前驱结点，叶子结点没有后继结点。军队中师、团、营、连、排的层次结构就是一种典型的树形结构，三军总司令是根，而士兵就是叶子。

4. 图形结构

图形结构中的数据元素之间存在多对多的任意关系。即各个结点可以有多个直接前驱，也可以有多个直接后继。城市之间四通八达的公路网就是图形结构。

有时也可以把数据逻辑结构简单地分为两种类型，即线性结构和非线性结构。非线性结构包括树形结构和图形结构。

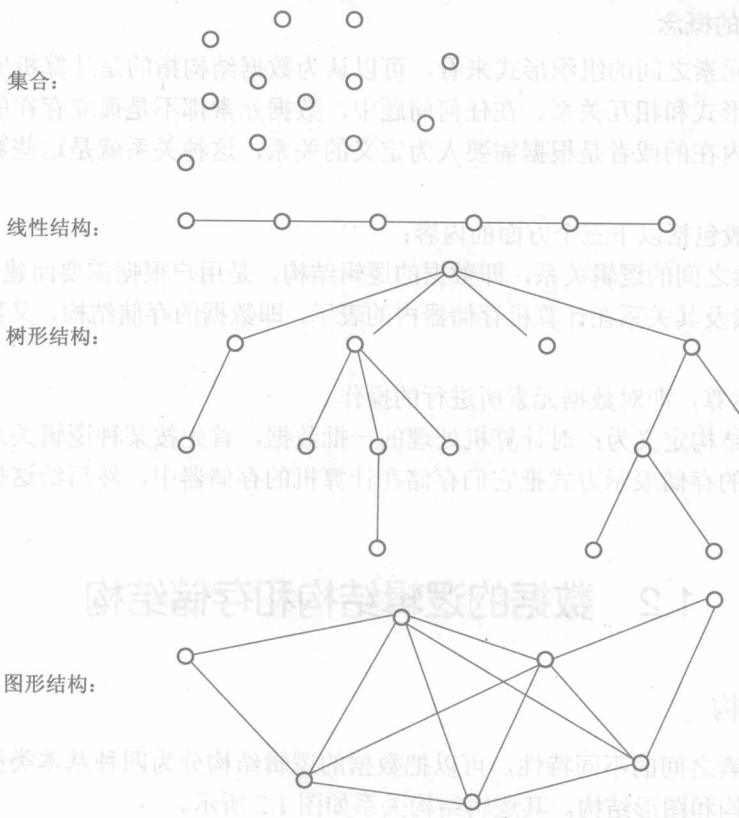

图 1.2 四种基本逻辑结构关系图

1.2.2 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像，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。逻辑结构与存储结构的关系为：存储结构是逻辑关系的映像与元素本身的映像。逻辑结构是数据结构的抽象，存储结构是数据结构的实现，两者综合起来建立了数据元素之间的结构关系。

存储结构可以分为顺序存储结构和非顺序存储结构两大类。顺序存储的特点是将数据元素按其逻辑顺序依次存储在内存中一组地址连续的存储单元中，即逻辑上相邻的结点物理上也必须相邻。非顺序存储则比较灵活，可以将数据分散存储在内存中，即逻辑上相邻的结点物理上不一定相邻。常用的非顺序存储有链式存储、Hash 存储和索引存储。

1.3 算 法

1.3.1 算法的概念及描述

1. 算法(Algorithm)

简单地说，算法就是解决特定问题的方法。严格地说，算法是由若干条指令组成的有

穷序列，其中每条指令表示计算机的一个或多个操作。例如：将一组给定的数据由小到大进行排序，解决的方法有若干种，而每一种排序方法就是一种算法。

一个算法必须具有以下五个特性：

(1) 有穷性。一个算法在执行有限条指令后必须要终止，且每条指令都要在有限时间内完成。不能形成无穷循环。

(2) 确定性。算法中每条指令必须有确切的含义，不会产生二义性

(3) 可行性。算法中描述的操作都是可以通过已经实现的基本运算执行有限次来实现的。

(4) 输入。一个算法有零个或多个的输入，这些输入取决于某个特定的对象的集合。

(5) 输出。一个算法有一个或多个结果输出。

算法和程序有所不同，程序可以不满足上述的有穷性。例如，Windows 操作系统在用户未操作之前一直处于“等待”的循环中，直到出现新的用户操作为止。

2. 算法、语言和程序的关系

(1) 算法。算法描述了数据对象的元素之间的关系(包括数据逻辑关系和存储关系)。

(2) 语言。算法可用自然语言、框图或高级程序设计语言进行描述。自然语言简单但易于产生二义性，框图直观但不擅长表达数据的组织结构，而高级程序语言则较为准确且比较严谨。

(3) 程序。程序是算法在计算机中的实现(与所用计算机及所用语言有关)。

1.3.2 算法的评价标准

对于一个特定的问题，采用不同的存储结构，其算法描述一般是不相同的，即使在同一种存储结构下，也可以采用不同的求解策略，从而有许多不同的算法。那么，对于解决同一问题的不同算法，选择哪一种算法较为合适，以及如何对现有的算法进行改进，从而设计出更好的算法，这就是算法评价问题。评价一个算法的优劣主要有以下几个标准：

1. 正确性

一个算法能否正确地执行预先的功能，这是评价一个算法的最重要也是最基本的标准。其要求是：

(1) 所设计的程序没有语法错误。

(2) 所设计的程序对于几组输入数据能够得出满足要求的结果。

(3) 所设计的程序对于精心选择的典型、苛刻而带有刁难性的几组输入数据能够得到满足要求的结果。

(4) 程序对于一切合法的输入数据都能产生满足要求的结果。

例如，求 5 个数的最大值问题，给出示意算法如下：

```
max=0;
for(i=1;i<=5;i++){
    scanf("%f ", &x);
    if (x>max)   max=x;
}
printf("%f ", max);
```

表面上看来该算法是正确的，输入 10.1、2.5、−3.4、−6.5、8.4 验证，结果也正确，可是当输入数据全部是负数的时候，输出结果却是 0，所以该算法并不正确。

2. 可读性

算法主要是为了人的阅读与交流，其次才是机器执行。即使算法已转变成机器可执行的程序，也需考虑人能较好地阅读理解。可读性好有助于人对算法的理解，这既有助于对算法中隐藏错误的排除，也有助于算法的交流和移植。

3. 健壮性

算法应具有很强的容错能力，即算法能对非法数据的输入进行检查和处理，不会因非法数据的输入而导致异常中断或死机等现象。

4. 运行时间

运行时间是指算法在计算机上运行所花费的时间，它等于算法中每条语句执行时间的总和。对于同一个问题如果有多个算法可供选择，应尽可能选择执行时间短的算法，一般来说，执行时间越短则算法的效率越高、性能越好。

5. 占用空间

占用空间是指算法在计算机存储器上所占用的存储空间，包括存储算法本身所占用的存储空间，算法的输入、输出数据所占用的存储空间和算法运行过程中临时占用的存储空间。算法占用的存储空间是指算法执行过程中所需要的最大存储空间。对于一个问题如果有多个算法可供选择，应尽可能选择存储量需求低的算法。

实际上，算法的时间效率和空间效率经常是一对矛盾体，相互抵触，有时增加辅助的存储空间可以加快算法的运行速度，即用空间换取时间；有时因为内存空间不够，必须压缩辅助的存储空间，从而降低了算法的运行速度，即用时间换取空间。通常把算法在运行过程中临时占用的存储空间的大小叫做算法的空间复杂度。算法的空间复杂度比较容易计算，它主要包括局部变量所占用的存储空间和系统为实现递归所使用的堆栈占用的存储空间。

1.3.3 算法的时间复杂度

一个算法的运行时间是该算法中每条语句执行时间的总和，而每条语句的执行时间是该语句的执行次数(也叫语句频度)与执行一次该语句所需时间的乘积。由于同一条语句在不同的机器上执行所需的时间是不相同的，也就是说执行一条语句所需的时间与具体的机器有关，因此要想精确地计算各种语句执行一次所需的时间是比较困难的。实际上，为了评价一个算法的性能，我们只需计算算法中所有语句执行的总次数即可。

任何一个算法最终都要被分解成一系列基本操作(如赋值、转向、比较、输入、输出等)来具体执行，每一条语句也要分解成具体的基本操作来执行，所以算法的运行时间也可以用算法中所进行的基本操作的总次数来估算。在一个算法中，进行简单操作的次数越少，其运行时间也相对越少。为了便于比较同一问题的不同算法，也可以用算法中的基本操作重复执行的频度作为算法运行时间的度量标准。

通常把算法中的基本操作重复执行的频度称为算法的时间复杂度。算法中的基本操作一般是指算法中最深层循环内的语句，因此，算法中基本操作重复执行的频度 T(n)是问题规模 n 的某个函数 f(n)，记作：T(n)= O (f(n))。其中“O”表示随问题规模 n 的增大，算法

执行时间的增长率和f(n)的增长率相同，或者说，用“O”表示数量级(Order of Magnitude)的概念。例如，若$T(n)=2n^2+3n+1$,则$2n^2+3n+1$的数量级与n^2的数量级相同，所以$T(n)=O(n^2)$。

如果一个算法没有循环语句，则算法的基本操作的执行频度与问题规模 n 无关，记作O(1)，也称常数阶。如果一个算法只有一重循环，则算法的基本操作的执行频度随问题规模n的增大而呈线性增大关系，记作O(n)，也称做线性阶。按数量级递增排列，常见的时间复杂度有：常数阶O(1)，对数阶$O(\log_2 n)$，线性阶O(n)，线性对数阶$O(n\log_2)$，平方阶$O(n^2)$，立方阶$O(n^3)$，…，k次方阶$O(n^k)$，指数阶$O(2^n)$。随着问题规模n的不断增大，上述时间复杂度也不断增大，算法执行效率依次降低。

下面举例说明计算算法时间复杂度的方法。

例 1.1 分析以下程序段的时间复杂度。

```
for(i=1;i<n;i++)
{
   y=y+1;                ①
   for(j=0;j<=(2*n);j++)
      x++;               ②
}
```

解：该程序段中语句①的频度是 n−1，语句②的频度是$(n-1)(2n+1)=2n^2-n-1$，则程序段的时间复杂度$T(n)=(n-1)+(2n^2-n-1)=2n^2-2=O(n^2)$。

例 1.2 分析以下程序段的时间复杂度。

```
i=1;                     ①
while(i<=n)
   i=i*2;                ②
```

解：该程序段中语句①的频度是1，设语句②的频度为f(n)，则有$2^{f(n)}\leqslant n$，即$f(n)\leqslant\log_2 n$，取最大值$f(n)=\log_2 n$，所以该程序段的时间复杂度$T(n)=1+f(n)=1+\log_2 n=O(\log_2 n)$。

例 1.3 分析以下程序段的时间复杂度。

```
a=0,b=1;                 ①
for(i=2; i<=n; i++)
{
   s=a+b;                ②
   b=1;                  ③
   a=s;                  ④
}
```

解：该程序段中语句①的频度是2，语句②、③、④的频度都是n−1，则该程序段的时间复杂度$T(n)=2+3*(n-1)=3n-1=O(n)$。

例 1.4 分析下列算法的时间复杂度。

```
prime(int n)
{
      int   i=2;
      while((n%i)!=0 && i*1.0<sqrt(n))
```