21世纪高职高专规划教材

计算机应用系列

计算机软件技术基础

马世霞 主编　　刘丹 程跃华 副主编

清华大学出版社

21世纪高职高专规划教材
计算机应用系列

计算机软件技术基础

马世霞　主　编
刘　丹　程跃华　副主编

清华大学出版社
北　京

内容简介

本书以软件基础知识为中心，目的是通过有限的篇幅，使学生掌握开发应用软件所必备的基础知识、方法和技能，建立开发软件系统的总体思路。本书共分9章，主要内容包括：算法、数据结构、操作系统、数据库系统、VB程序设计基础、软件工程、计算机网络、网页制作、动画制作基础。本书强调基本概念、技术和方法的阐述，注重理论联系实际。书中列举许多实例，每章都有习题，有利于读者提高解决实际问题的能力。

本书可以作为高职高专计算机类的教材及职业培训教材，也可作为其他专业学生的选学教材，还可以作为计算机初学者的参考书。

图书在版编目（CIP）数据

计算机软件技术基础/马世霞主编．—北京：清华大学出版社，2010.7
（21世纪高职高专规划教材．计算机应用系列）
ISBN 978-7-302-22642-0

Ⅰ．①计…　Ⅱ．①马…　Ⅲ．①软件－高等学校：技术学校－教材　Ⅳ．①TP31

中国版本图书馆CIP数据核字(2010)第074520号

责任编辑：张龙卿
责任校对：袁　芳
责任印制：王秀菊

出版发行：清华大学出版社　　**地　　址**：北京清华大学学研大厦A座
http://www.tup.com.cn　　**邮　　编**：100084
社　总　机：010-62770175　　**邮　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质　量　反　馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：北京市人民文学印刷厂
装 订 者：三河市兴旺装订有限公司
经　　销：全国新华书店
开　　本：185×260　　**印　　张**：19.5　　**字　　数**：467千字
版　　次：2010年7月第1版　　**印　　次**：2010年7月第1次印刷
印　　数：1～3000
定　　价：29.00元

产品编号：034470-01

前　言

随着信息技术的快速发展，处于其核心地位的软件技术，在经济发展和社会进步中发挥着越来越大的作用，因此，掌握计算机应用技能成为时代对大学生素质的基本要求。

通过本教材的学习，学生可掌握计算机软件的基础知识和网络知识，为学习后续课程打下坚实的基础。

参编本书的教师都是具有多年教学经验的一线老师，他们总结多年来教学的实践经验，在结合当前教学要求的基础上编写了本书，全书共分 9 章。第 1 章介绍算法的基本概念及描述方法。第 2 章介绍了数据结构，包括栈和队列等基本的线性数据结构，树、图等非线性数据结构，以及排序和查找等基本的程序操作。第 3 章介绍操作系统的形成和发展，操作系统的作用与类型，处理器管理，存储管理，设备管理和文件管理等。第 4 章介绍常用关系数据库系统的类型、关系数据库理论基础及关系数据库管理系统的设计等。第 5 章通过实例讲述了 VB 的使用方法。第 6 章介绍了软件工程的概念、生命周期、UML 以及软件测试与调试等内容。第 7 章介绍了计算机网络的概念、计算机网络体系结构、因特网和网络安全技术等内容。第 8 章以 Dreamweaver 8 为例介绍了网页的制作方法。第 9 章介绍了 Flash 动画的制作方法。本书各部分内容相对独立，自成体系，讲授时可根据教学需要酌情取舍。各章后面附有小结和习题。

本书由马世霞主编，其中第 1 章和第 2 章由马世霞编写，第 3 章由崔艳编写，第 4 章由程跃华编写，第 5 章由牛波编写，第 6 章由陆璐编写，第 7 章由李邦编写，第 8 章和第 9 章由刘丹编写。

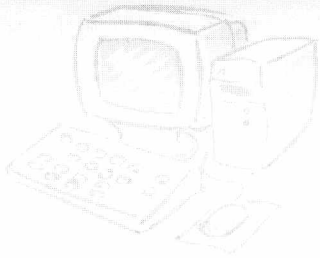
本书在编写的过程中，参考了大量计算机方面的书籍、资料（包括互联网上的资料），在此对有关作者、译者表示感谢。由于开设本课程的高校目前尚不普遍，课程内容也不尽一致，恳请广大读者对本教材的内容与不足提出意见，以利于以后改进。

本书配有电子课件，可以从清华大学出版社的网站 www.tup.com.cn 下载。

编　者

2010 年 2 月

目　录



第1章 算 法

本章学习目标：

- 掌握算法的定义。
- 了解算法的特性及算法设计的要求。
- 掌握算法的表示法。

算法是程序设计的灵魂，在程序设计时，先要进行算法的设计，对于算法的描述要做到直观、清楚，才能方便程序的设计。下面就从算法定义及特性、算法描述、算法性能分析与度量三个方面对算法进行介绍。

1.1 算法的概念

日常生活中做任何事情，一般都要按照一定规则和步骤进行。比如汽车的生产，先把汽车零件按一道道工序加工，然后再按一定法则组装成完整的汽车，汽车安装的工艺流程就是算法。又如科学计算的数值问题，解线性方程等的计算方法，是数值计算的算法；文字处理、图像图形等的排序、分类、查找，是非数值计算的算法。

在我们身边处处都有算法，如乐谱是乐队演奏的算法，菜谱是做菜肴的算法，珠算口诀是使用算盘的算法。算法并不是给出问题的精确解，而是说明如何才能得到解。每一个程序算法都是由一系列的操作指令组成的，这些操作包括加、减、乘、除、判断等，按顺序、分支、重复等结构组成，所以研究算法的目的就是研究如何将各种类型的问题的求解过程分解成一些基本的操作。

1.1.1 算法的定义

瑞士著名的计算机科学家 N. Wirth 提出了著名公式“程序＝算法＋数据结构”。所谓算法，就是为解决特定问题而采取的步骤和方法，即是对特定问题求解步骤的一种描述，是指令的有限序列，其中每一条指令表示一个或多个操作。

【实例 1-1】 计算：9＋5×(7－2)。

9＋5×(7－2)的计算步骤如下：

第一步，先算括号内的 7－2＝5；

第二步，再做乘法 5×5＝25；

第三步，最后做加法 9＋25＝34。

提示 简单地说，算法就是解决问题的方法和步骤。人或计算机解决问题都需要遵循一定的方法和思路并正确地列出各个求解步骤，这种解题步骤就称为算法。

【实例 1-2】 农夫过河问题。一个农夫带着一只狼、一只羊和一棵白菜，处于河的南岸，他要将这些东西带到北岸。他面前只有一条小船，而且每次他只能带一件东西过河。由于食物链的关系，他不能将狼和羊同时留下，也不能将羊和白菜同时留下，但可以将狼和白菜同时留下。请问，农夫如何过河？

农夫过河的操作步骤如下：

第一步，农夫带羊过河；

第二步，农夫独自回来；

第三步，农夫带狼过河；

第四步，农夫带羊回来；

第五步，农夫带蔬菜过河；

第六步，农夫独自回来；

第七步，农夫带羊过河。

算法写好后，要检查其正确性和完整性，然后才是用某种高级语言编写的程序。程序设计的关键是设计一个好的算法，下面讨论算法的特性。

1.1.2 算法的特性

一个算法应该具有下列特性。

(1) 有穷性。一个算法必须在有穷步之后结束，即必须在有限时间内完成。事实上“有穷性”往往指“在合理的范围之内”。如果让计算机执行一个历时 1000 年才结束的算法，这虽然是有穷的，但超过了合理的限度，人们不认为它是有效算法。

如下列代码就不合理，执行时会死循环：

```
main()
{
    while(1>0)
    printf("无限循环");
}
```

(2) 确定性。算法中的每一个步骤都应当是确定的，而不应当是含糊的、模棱两可的。算法的含义应当是唯一的，而不应当产生歧义性。

(3) 有效性。算法中的每一步都可以通过已经实现的基本运算的有限次执行得以实现。每个语句必须有效地执行，并得到确定的结果。比如 a/b，当 b＝0 时，则 a/b 是不能有效执行的。

(4) 输入。一个算法具有 0 个或多个输入，这些输入取自特定的数据对象集合。

(5) 输出。一个算法具有一个或多个输出，这些输出同输入之间存在着某种特定的关系。

注意 算法的含义与程序十分相似，但又有区别。一个程序不一定满足有穷性。例如，操作系统用来管理计算机资源，控制作业的运行，只要整个系统不遭破坏，它将永远不会停止，即使没有作业需要处理，它仍处于动态等待中。因此，操作系统不是一个算法。另一方面，程序中的指令必须是机器可执行的，而算法中的指令则无此限制。算法代表了对问题的解，而程序则是算法在计算机上的特定的实现。一个算法若用程序设计语言来描述，则它就是一个程序。

1.1.3 算法设计的要求

要设计一个好的算法，通常要考虑以下的要求。

(1) 正确。算法的执行结果应当满足预先规定的功能和性能要求。正确性包含以下四个方面的含义。

- 程序不含语法错误。
- 程序对几组输入有正确输出。
- 程序对几组典型、苛刻的输入有正确输出。
- 程序对一切合法输入有正确输出。

(2) 可读。一个算法应当思路清晰、层次分明、简单明了、易读易懂。

(3) 健壮。当输入不合法数据时，应能作适当处理，不至引起严重后果。

(4) 高效。有效使用存储空间和有较高的时间效率。

1.2 算法描述

算法可以使用各种不同的方法来描述。常用的算法描述方式如表 1-1 所示。

表 1-1 算法描述方式

算法描述种类	算法描述说明
自然语言	日常生活中使用的语言
流程图	特定的表示算法的图形符号
伪语言	包括程序设计语言的三大基本结构及自然语言的一种语言

最简单的方法是使用自然语言。自然语言是人们日常生活中使用的语言，可以是汉语、英语或其他语言。用自然语言表示算法通俗易懂；缺点是文字冗长，容易出现“歧义性”。自然语言表示的含义往往不太严格，要根据上下文才能判断其正确含义。假如有这样一句话：“小明对小华说他考试过关了。”请问是小明考试过关了还是小华考试过关了呢？光从这句话本身难以判断。此外，用自然语言描述包含分支和循环的算法，不太方便，因此，除了很简单的问题外，一般不用自然语言描述算法。

注意 使用自然语言描述算法时要使描述要求尽可能精确、详细。

流程图也称程序框图，它是算法的一种图形化表示方法，是用一组几何图形表示各种类型的操作，在图形上用简明扼要的文字和符号表示具体的操作，并用带有箭头的流线表示操作的先后次序。

用流程图描述算法的特点是形象、直观，容易理解，逻辑结构明显，描述过程简洁、明了，对于初学者来说是最合适的一种算法描述方法。

提示 常用的流程图图形符号是我们应该牢记的。

用自然语言和流程图描述的算法不能够直接在计算机上执行，若要将它转换成可执行的程序，还有一个编程的问题。可以直接使用某种程序设计语言来描述算法，不过直接使用程序设计语言并不容易，而且不太直观，常常需要借助于注释才能使人看明白。

流程图适宜于表示一个算法，但在设计算法过程中使用不是很理想(尤其是当算法比较复杂、需要反复修改时)。为了设计算法时方便，常用一种称为伪代码的工具。

伪代码是用介于自然语言和计算机语言之间的文字与符号来描述算法。它忽略高级程序设计语言中一些严格的语法规则与描述细节，它如同一篇文章一样，自上而下地写下来。每一行(或每几行)表示一个基本操作。它不用图形符号，因此书写方便、格式紧凑，既易懂又便于向计算机语言算法(即程序)过渡。可以用英文、汉字、中英文混合表示算法，以便于书写和阅读为原则。用伪代码写算法并无固定的、严格的语法规则，只要把意思表达清楚，并且格式要写成清晰易读的形式。

【实例 1-3】 求自然数 1×2×3×4×5 的值，分别用自然语言描述和流程图方法描述。

(1) 用自然语言描述：

第一步，计算 1×2，得 2。

第二步，将第一步中的运算结果 2 与 3 相乘得 6。

第三步，将第二步中的运算结果 6 与 4 相乘得 24。

第四步，将第三步中的运算结果 24 与 5 相乘得 120。

(2) 用流程图描述：

① T=1；

② I=2；

③ T=T×I；

④ I=I+1；

⑤ 如果 I≤5，转③；否则输出 T。

其中，③、④、⑤组成一个循环，在实现算法时要反复多次执行③、④、⑤步骤，直到执行⑤时，经过判断，乘数 T 已超过规定的数为止。算法流程图如图 1-1 所示。

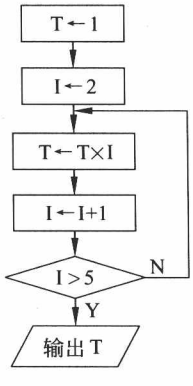


图 1-1 算法流程图

【实例 1-4】 在一个有序序列中查找最大元素算法，采用伪代码来表示算法。

```
int max(int a[])
max:=a1
for(i=2;i<=n;i++)
    if max<a[i] then max:=ai
{max is the largest element}
```

使用伪代码描述算法没有严格的语法限制，书写格式也比较自由，只要把意思表达清楚就可以了，它更侧重于对算法本身的描述。

在伪代码描述中，表示关键词的语句一般用英文单词，其他语句可以用英文语句，也可以用汉语语句。

美国著名计算机科学家克努特教授提出了“计算机科学就是研究算法的科学”的著名论断，说明了算法在设计程序中的重要性，解决任何问题都必须设计算法。对于算法的不同描述方法，各有自己的优缺点，我们要根据自己的不同需求，选择适合自己的最佳算法。

1.3 算法性能分析与度量

一个算法的优劣可以从算法的时间复杂度与空间复杂度来评价。

算法运行的时间分析和程序运行的时间分析有区别。同一算法由不同的程序员编写，程序有好坏之分，运行时间也就不同；程序运行速度与计算机的硬件、编程语言有关。这里我们所考虑的是对解决问题的算法做时间上的度量分析，或对解决同一问题的两种或两种以上的算法运行的时间加以比较。我们称这种度量分析为算法时间复杂度分析。

1.3.1 时间复杂度

算法的执行时间是依据该算法编制的程序在计算机上运行时所消耗的时间来度量。

通常把算法中进行简单操作的次数的多少称为算法的时间复杂度，这是一个算法执行时间的相对度量，它可估算出当问题的规模变大时，算法运行时间增长的速度。这种分析实际上是一种数学化的估算方法。

一个算法是由控制结构和原操作构成的，其执行时间取决于两者的综合效果。为了便于比较同一问题的不同的算法，通常的做法是：从算法中选取一种对于所研究的问题来说是基本运算的原操作，以该原操作重复执行的次数作为算法的时间度量。

当算法简单时，时间复杂度容易计算；当算法比较复杂时，复杂度的计算就相对困难。实际上，一般也没必要计算出算法的精确复杂度，只要大致计算出相应的数量级即可。

若所解决的问题的数据规模（数据量）为 n，那么算法的时间复杂度就是数据规模 n 的一个函数 $f(n)$，假定时间复杂度记作 $T(n)$，则

$$T(n)=O(f(n))$$

它表示算法的时间复杂度 $T(n)$ 的增长率与 $f(n)$ 的增长率相同，其中大写字母为英文 Order（数量级）的第一个字母，$f(n)$ 与 $T(n)$ 只相差一个常数倍。使用大 O 记号表示的算法的时间复杂度，称为算法的渐进时间复杂度。例如：

一个算法 A，其时间耗用函数为：

$$T(n)=20n^2+30n$$

另一个算法 B，其时间耗用函数为：

$$T(n)=5n^2+10n+20$$

那么，这两个算法的时间耗用函数的阶是一样的，都是 $O(n^2)$。

算法执行时间大致为基本运算所需的时间与其运算次数（也称为频度）的乘积。被视为算法基本运算的一般是最深层循环内的语句。

技巧 通常把算法中包含基本运算次数的多少称为算法的时间复杂度，也就是说，一个算法的时间复杂度是指该算法的基本运算次数。只求出 $T(n)$ 的最高阶，忽略其低阶项和常系数，这样既可简化 $T(n)$ 的计算，又能比较客观地反映出当 n 很大时算法的时间性能。

如一个程序的实际执行时间为 $T(n)=2.7n^3+3.8n^2+5.3$，则 $T(n)=O(n^3)$。

【实例 1-5】 分析以下算法的时间复杂度。

```
int fun(int n)
{
    int i,j,k,s;
    s=0;
    for(i=0;i<=n;i++)
        for(j=0;j<=i;j++)
            for(k=0;k<=j;k++)
                s++;                          //基本语句或基本操作
    return(s);
}
```

解：该算法的基本操作是语句 s＋＋，其频度：$T(n)=O(n^3)$，则该算法的时间复杂度为 $O(n^3)$。

技巧 一个没有循环的算法的基本运算次数与问题规模 n 无关，记作 $O(1)$，也称作常数阶。一个只有一重循环的算法的基本运算次数与问题规模 n 的增长呈线性增大关系，记作 $O(n)$，也称线性阶。二重循环的时间复杂度应为 $O(n^2)$，三重循环的时间复杂度为 $O(n^3)$……

通常用 $O(1)$ 表示常数计算时间。常见的渐进时间复杂度有：

$$O(1)<O(\log_2 n)<O(n)<O(n\log_2 n)<O(n^2)<O(n^3)<O(2^n)$$

1.3.2 空间复杂度

一个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、算法中的输入/输出数据所占用的存储空间和算法在运行过程中临时占用的存储空间这三个方面。

算法中输入/输出数据所占用的存储空间是由要解决的问题所决定的，它不随算法的改变而改变。例如 100 个数据元素的排序算法与 1000 个数据元素的排序算法所需的存储空间显然是不同的。

存储算法本身所占用的存储空间是与算法书写的长度有关，算法越长，占用的存储空间越多。

算法在运行过程中临时占用的存储空间随算法的不同而改变，有的算法只需要占用少量的临时工作单元，与待解决问题的规模无关；有的算法需要占用的临时工作单元，与待解决问题的规模有关，随问题的规模的增大而增大。

算法在运行过程中临时占用的存储空间大小被定义为空间复杂度，一般也作为问题规模 n 的函数，以数量级形式给出，记作：

$$S(n)=O(g(n))$$

技巧 若所需额外空间相对于输入数据量来说是常数，则称此算法为原地工作；若所需存储量依赖于特定的输入，则通常按最坏情况考虑。

【实例 1-6】 在三个整数中求最大的一个数，分析其算法的空间复杂度。

解法 1：

```
int Max1(int x,y,z)
{   if(x>y)
    {  if(x>z) return x;                    //比较 x,z
       else return z;
    }
    {  if(y>z) return y;                    //比较 y,z
       else return z;
    }
}
```

算法 Max1 无须额外存储空间，并且只需比较两次，算法为原地工作。

解法 2：

```
int Max2(int a[3])
{   x=a[0];
    for(i=1;i<3;i++)
    if(a[i]>x)x=a[i];
    return x;
}
```

算法 Max2 需 x、a[0]两个额外存储空间。

由于这两个算法中临时变量的个数与问题规模 n 无关，所以空间复杂度均为 $O(1)$。

1.4 小 结

广义地说，为解决一个问题而采取的方法和步骤，就称为算法。计算机算法可分为两大类：数值算法和非数值算法。本章介绍了算法的定义及其特性、算法的时间复杂度和空间复杂度分析。

课程难点：算法设计的要求。

难点解决办法：例题的讲解，使学生逐步感悟算法的优劣。

1.5 习 题

1. 填空题

(1) 算法，就是为解决特定问题而采取的_____和_____。

(2) 一个算法应该具有 5 个特性，_____、_____、_____、_____、_____。

(3) 一个算法必须在有穷步之后结束，即必须在有限时间内完成，这是算法的_____特征。