计算机程序设计典型例题精解

杨克昌　编著

国防科技大学出版社

计算机程序设计典型例题精解

杨克昌 编著

国防科技大学出版社
·长沙·

图书在版编目（CIP）数据

计算机程序设计典型例题精解/杨克昌编著. - 长沙: 国防科技大学出版社, 1999.3

ISBN 7-81024-531-7

I 计算机程序设计典型例题精解
II 杨克昌
III ①计算机程序设计 ②典型例题
IV TP31

国防科技大学出版社出版发行
电话: (0731)4555681　　邮政编码: 410073
E-mail:gfkdcbs@public.cs.hn.cn
责任编辑: 黄 煌　　责任校对: 文 慧
新华书店总店北京发行所经销
国防科技大学印刷厂印装
*
787×1092　1/16　印张: 17.75　字数: 410千
1999年3月第1版第1次印刷　　印数:1-4000册
*

定价: 20.00 元

内 容 提 要

本书面向大中专学生选讲计算机程序设计典型问题求解，旨在帮助学生熟练掌握程序设计的基本方法与技巧，培养与提高他们通过程序设计解决实际问题的能力。

本书以程序设计求解问题为主线，取材注重典型性与趣味性，问题分类精选，内容新颖丰富。所选讲的问题包括典型的数学问题求解，新颖的数字游戏，有趣的逻辑推理与智力测试，实用的近似计算与高精度计算，引人入胜的著名数学猜想验证，常见的数据处理等。既有引导入门的基本题、常规题，也有难度较大的综合题、引申题，深入浅出，难度适宜。

选讲的例题求解注重算法引导分析与不同算法的比较，程序选用目前使用较广的Turbo C，Qbasic，Foxpro（Foxbase）等语言编写，注重结构化与可读性，实用性强。为便于程序设计对照比较，对其中有些问题采用多种算法求解与多种语言设计程序实现。

为方便程序设计查阅，附录中列出了上述几种常用程序设计语言的语法提要。

本书可供本专科学生作为计算机教学、计算机等级考试以及程序设计竞赛参考用书，也可供中等学校学生计算机选修与程序设计竞赛培训选用。

前　言

计算机是20世纪人类最辉煌最伟大的科技发明之一。以计算机为核心的信息科学的迅猛发展和广泛应用正在对人类社会的发展进程以及人们的工作方式与思维方式的改变产生深远的影响。处在世纪之交的今天，计算机技术的应用与发展，是一个国家综合国力的体现与科技发展水平的象征。进入信息社会，计算已成为与理论研究、科学实验相并列的第三种研究方法。作为人类智慧的结晶与人脑功能的延伸，计算机已经成为人类进行复杂计算与模拟探索的"人类通用智能工具"，广泛应用于社会生活的各个领域，并发挥着越来越大的作用。

从当前大中专各专业计算机课程的系统开设，到各中小学计算机课外活动的广泛开展，说明了计算机教育作为一种新的文化，正在中国广泛普及。顺应信息产业不断发展与计算机教育不断深入的潮流，帮助广大青少年与大中专在校学生掌握计算机的基本理论与基本技能，在程序设计中熟悉基本算法，开拓求解思路，解决实际问题，培养创新意识，不断提高程序设计的水平与应用计算机解决实际问题的能力，是我们计算机教育工作者义不容辞的职责。

继实施全国计算机专业技术资格与水平考试之后，1994年国家教委考试中心推出面向社会的全国计算机等级考试（1～4级）。各省市也相继开设了面向高校非计算机专业学生的计算机水平等级考试。同时，高等学校组织的各种计算机竞赛也在逐步推出。大中专在校学生在学习了计算机文化基础与计算机应用基础以及一些计算机高级语言程序设计的基础上，要参加一系列的计算机考试与竞赛，其中一个重点也是难点就是程序设计，即通过程序设计解决实际问题。通常程序设计课堂教学以语法为主线，算法相对薄弱，题型相对狭窄，学生程序设计思路单调，应用设计程序解决实际问题的能力与设计变通能力较低，很多学生难以适应这些等级考试，甚至在程序设计竞赛中无所适从。为此，作为课堂教学的一个补充，笔者以程序设计问题求解为主线，分类精选了程序设计中的百余个典型例题，包括重新归纳在课堂初步了解的基础题，常规题，以及渐进的实用性较强、难度较大的引申题、综合题，以帮助指导程序设计教学的应用提高及计算机等级考试与竞赛前的程序设计复习。

为了开拓程序设计的思路，对每一个问题注重算法分析以及多种算法的对

照比较。程序采用目前大学生中使用率排在前三位的Turbo C，Qbasic，Foxpro（Foxbase）语言编写。对其中有些问题，采用了多种语言以不同算法与表现形式设计出不同的程序，使之更适合于读者学习。在各个问题程序设计之后，尽可能给出问题的输出结果或运行示例，以帮助读者对所设计程序有更清晰的了解。

本选讲中部分问题参考或选用了各届国际国内奥林匹克信息学（计算机）竞赛有代表性的试题，以帮助计算机爱好者在程序设计上开阔视野，有一个深层次的练习与提高。

为方便读者练习交流，本选讲中的所有程序在程序开头的注释语句中给出了程序编号。编号统一设置三个字符：章号字母（对应第1章为A，第16章为P），题号，程序号，如G32是第7章第3题（7.3）的第2个程序。

在书稿的整理阶段，王岳斌副教授审阅了全部书稿，并运行了书中涉及的所有程序，笔者谨在此表示深深的谢意。

尽管每一个程序都经反复检查与运行分析，因涉及内容较广，难免存在各种差错，恳请读者予以指正。

杨克昌

1998.8

目 录

第 1 章　和积入门

求和求积以及和积综合常规求解是计算机程序设计的基础，也是计算机等级考试与程序设计竞赛的基本内容。

简单和积处理作为程序设计的入门，从求解算法到程序实现不存在太多难点。我们着眼通过基本和积问题的求解来培养良好的程序设计风格，同时学习一些基本程序结构与相应编程语句的应用技巧。

1.1　有规律数据求和

常见的数据求和分为对一组有规律数据的求和与对若干个无规律的离散数据的求和两种。

有规律的一组数据通常可以写出它的第 i 项的通项式 f(i)，在设置的求和 i 循环中，使用赋值语句 s=s+f(i)，把 f(i)累加到 s 中，即可简便地实现求和。

1.1.1　整数求和

求：$s = 1 \cdot 2 \cdot 3 + 3 \cdot 4 \cdot 5 + \cdots + 99 \cdot 100 \cdot 101$

易知通项 f(i)=i*(i+1)*(i+2)，i=1，3，…，99。于是可简单地由以下 Foxpro 程序实现求和：

```
* 求  s=1*2*3+3*4*5+…+99*100*101 (A11)
set  talk  off
s=0
for  i=1  to  99  step  2
  s=s+i* (i+1)*(i+2)
endfor
? "1*2*3+3*4*5+…+99*100*101="+str(s)
return
```

运行程序得：

```
    1*2*3+3*4*5+…+99*100*101=13002450
```

对于这一求和问题，可把通项改写为 f (i) = (i－1) * i * (i+1)，i=2，4，…，100。于是，求和程序可改写为：

```
* 求  s=1*2*3+3*4*5+…+99*100*101 (A12)
set  talk  off
s=0
for  i=2  to  100  step  2
    s=s+(i－1)*i*(i+1)
endfor
```

```
? "1*2*3+3*4*5+…+99*100*101="+str(s)
return
```

运行程序所得结果完全相同。

可见，求解一个问题，程序设计是灵活的，是可以变通的。程序设计的变通比较是提高程序设计能力的一个有效方法。

1.1.2 实数求和

求和

$$s=\frac{\sqrt{2}}{1}+\frac{\sqrt{3}}{2}+\cdots+\frac{\sqrt{n+1}}{n}$$ （正整数 n 从键盘输入）

显然，第 i 项的通项公式为 SQR(i+1)/i（i=1，2，…，n）。于是在 i 循环中把该通项表达式累加到和变量 s 即可完成求和。

```
REM  求和 s= SQR (2)+ SQR (3)/2+…+ SQR (n+1)/n   (A13)
INPUT  "请输入 n:";  n
s = 0
FOR  i = 1  TO  n
  s = s + SQR(i + 1) / i
NEXT  i
PRINT  "s=";  s
END
```

运行程序，输入 n=100，得

s=19.67395

1.1.3 舍罕王的失算

相传国际象棋是古印度舍罕王的宰相达依尔发明的。舍罕王十分喜爱象棋，决定让宰相自己选择何种赏赐。这位聪明的宰相指着 8 × 8 共 64 格的象棋盘说：陛下，请您赏给我一些麦子吧，就在棋盘的第 1 个格中放 1 粒，第 2 格放 2 粒，第 3 格放 4 粒，以后每一格都比前一格增加一倍，依此放完棋盘上 64 格，我就感恩不尽了。舍罕王让人扛来一袋麦子，他要兑现他的许诺。

请问，国王能兑现他的许诺吗？共要多少麦子赏赐他的宰相？合多少立方米？（1 立方米麦子约 1.42e8 粒）

1. *算法分析*

这是一个典型的等比数列求和问题。

第 1 格 1 粒，第 2 格 2 粒，第 3 格 $4=2^2$ 粒，……，第 i 格为 2^{i-1} 粒，于是总粒数为

$$s=1+2+2^2+2^3+\cdots+2^{63}$$

设置求和 i 循环，把每一项的通项 2^{i-1}（或由 t=t*2 得到的累乘量 t）累加到和变量 s 中，即可实现该等比数列求和。

2. 程序设计与运行结果

```
* 求 s=1+2+4+8+…+2^63  (A14)
set  talk  off
t=1
s=1
for  i=1  to  63
  t=t*2
  s=s+t
endfor
? "总麦粒数为:"+str(s,11)
? "折合体积为:"+str(s/1.42e8,11)+" 立方米"
return
```

运行程序，得

总麦粒数为: .18446e20

折合体积为: .12990e12 立方米

这是一个非常庞大的数值，相当于全世界若干世纪的全部小麦。看来舍罕王失算了，他无法兑现他的诺言。

3. 注意

程序中设置通项量 t，从 t=1 开始，应用 t=t*2 累乘计算通项。显然，当 i=1 时，t=2；当 i=2 时，t=4……这一处理技巧是常用的。尤其是在要求通项与和必须是准确值时，常采用上述累乘而不用 2^i。

1.2 无规律离散数据处理

对若干无规律的离散数据求和，通常把已有数据置于 DATA 语句或某一数组中，或从键盘输入，并在数据的最后设置一个区别已有数据的终止标志。在循环中每读一个数或一组数据，作相应的处理，直至读到终止标志时输出结果后结束。

1.2.1 数据求平均

已给若干个无规律的离散正数置于 DATA 语句中，设置 −1 作为终止标志（若数据中有负数，则终止标志可以设置一个大数，例如 1e30)。设置变量 x 在循环中读取数据，若 x 非终止标志，则累加到和变量 s 中：s=s+x。为求平均需要，还要用一个变量统计数据的个数：n=n+1。直至 x 为终止标志时结束。

设计离散数据求和的程序设计:

```
REM  离散数据求平均 (A21)
DATA  9.36,8.95,9.32,9.27,9.44,8.91,9.20,-1
n = 0
READ  x
WHILE  x > 0
```

```
    PRINT   x; : s = s + x: n = n + 1
    READ   x
WEND
PRINT : PRINT "以上";  n;  "个数据之和为:";  s
PRINT   "平均值为:";  INT(s / n * 100 + .5) / 100
END
```

运行程序，得

以上 7 个数据之和为:64.45

平均值为:9.21

注意：上述求平均使用了四舍五入精确到小数点后 2 位的表达式。

1.2.2 电阻计算

求如图 1.1 所示电路中 A,B 间的等效电阻 r。

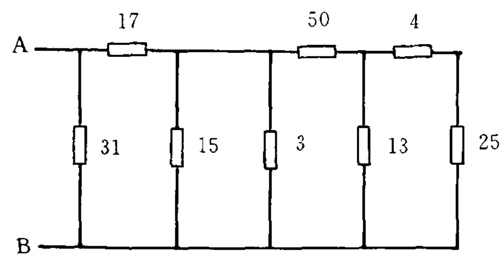

图 1.1

1. *算法分析*

把整个电路分为 4 级，从左至右逐级递推处理。

首先注意把电路中的电阻数据按顺序写入 DATA 语句：最右端电阻在最前；.以后每级横向电阻在前，竖向电阻在后。第 3 级没有横向电阻，相应位置应置数据 0。最后，置两个 − 1 作为数据的终止标志。

操作时用 r 读入最右端电阻。进入 i 循环（预置 20 级），读入 x，y 两个电阻，x 与右边级等效电阻串联，然后与 y 并联，根据电阻计算公式作操作:

r = r + x: r = (r * y) / (r + y)

当读入的 x 为负时，即已计算完毕，退出循环输出等效电阻 r。

2. *程序设计与运行结果*

```
REM   电阻计算 (A22)
PRINT   "(注意把电路中的电阻值写入 DATA 语句:)"
DATA   25,4,13,50,3,0,15,17,31, - 1, - 1
READ   r
FOR   i = 1   TO   20
    READ   x, y
    IF   x < 0   THEN   EXIT   FOR
    r = r + x: r = (r * y) / (r + y)
NEXT   i
PRINT   "A,B 间的等效电阻为:";  INT(r * 1000 + .5) / 1000
END
```

运行程序，得

A,B 间的等效电阻为:11.932

3. 注意

本程序只适用于Π形式的电路电阻计算。如果对应位置没有电阻，必须相应补 0。同时，在程序中循环体内判断 x<0 语句应安排在具体计算语句之前。否则，将导致计算结果出错。

1.3 求加减代数和

求一组有规律数据作加减符号有规律变化的代数和，要具体根据符号变化规律，通过设置符号变量，或对每一项通过条件判断选择加减操作，以达到所求代数和的目的。这里，给出三个有代表性的加减代数和求解实例。

1.3.1 设置符号变量实现加减相间变化

试求：s=1－1/2＋1/3－1/4＋ … 1/n（n 从键盘输入）

设置符号变量 f，通过语句 f=－1*f 实现正负交替变号，以达到加减相间求和。

```
REM  求代数和(符号一正一负)  (A31)
PRINT  "s=1－1/2+1/3－1/4+…1/n"
INPUT  "请输入 n:";  n
s = 0: f =  －1
FOR  i = 1  TO  n
    f =－1 * f: s = s + f / i
NEXT  i
IF  n  MOD  2 = 0  THEN  x$ = "－"  ELSE  x$ = "+"
PRINT  "s=1－1/2+1/3－1/4+…";  x$;  "1/";  n;  "=";  s
END
```

运行程序，输入 n=100，得

s=1－1/2+1/3－1/4+…－1/100=0.6881718

注意：符号变量 f 的初值应由第 1 项的符号决定。本例中代数和的第 1 项为正，因而赋初值 f=－1。根据项数 n 的奇偶性决定输出时该项的“+”、“－”符号 x$是必要的。

1.3.2 通过条件判断选择加减操作

试求：s=1－1/2－1/3＋1/4－…1/n（n 从键盘输入）

注意到每 3 项中有一项为加、两项为减，即每作一次加后作两次减，通过对循环变量 i 除 3 的余数（i MOD 3 = 1 时为加操作）作为条件，选择指定的加减操作。

```
REM  求代数和(符号一正两负) (A32)
PRINT  "s=1－1/2－1/3+1/4－…1/n"
INPUT  "请输入 n:";  n
s = 0
FOR  i = 1  TO  n
    IF  i  MOD  3 = 1  THEN  s = s + 1 / i  ELSE s = s－1 / i
```

```
NEXT  i
IF  n  MOD  03 = 1   THEN   x$ = "+"   ELSE   x$ = " - "
PRINT  "s=1 - 1/2 - 1/3+1/4+···";  x$;  "1/";  n;  "=";  s
END
```

运行程序，输入 n=100，得

s =1 - 1/2 - 1/3+1/4+···+1/100= - 0.7516997

1.3.3 设置多个符号变量

试求：s=1 - 1/2 - 1/3+1/4+1/5 - ···1/n （n 从键盘输入）

对于一些符号变化较复杂的代数和，可以依据具体情况设置多个符号变量，以达到选择加减操作的要求。上述代数和开始一项为正，以后二负二正类推。这时，设置两个符号变量可达到相应操作的选择。

```
REM  求代数和(符号一正开始,然后为二负二正类推.) (A33)
PRINT  "s=1 - 1/2 - 1/3+1/4+1/5 - ···1/n"
INPUT  "请输入 n:";  n
s = 0: f =  - 1: t = 1
FOR  i = 1  TO  n
    f =  - 1 * f: t = f * t: s = s + t / i
NEXT  i
IF  t < 0  THEN  x$ = " - "  ELSE  x$ = "+"
PRINT  "s=1 - 1/2 - 1/3+1/4+1/5 - ···";  x$;  "1/";  n;  "=";  s
END
```

运行程序，输入 n=100，得

s =1 - 1/2 - 1/3+1/4+1/5 - ···+1/100=0.4387752

注意：程序中设置符号变量 f，t 协同实现二正二负的变化要求。

1.4 积与和积综合求解

本节设计求解常见的产值翻番，阶乘 n!与常数 e 的近似计算等简单积运算与和积综合处理问题。

1.4.1 产值翻番

工业产值的增长率为每年 c%。当 c 分别为 6，8，10，12 时，试求工业产值分别过多少年实现翻番（设当年产值为 100）。

把年增长率置于 DATA 语句中，在循环中用变量 c 读取。产值赋初值 100，增长率 c%，每增长一年，年数 y 增 1，产值为

$$s = s * (1+c/100)$$

当 s<200（循环条件），继续增长，直至 s>200 时，已达到翻番，打印输出相应的增长率 c%，翻番所需年数以及翻番后的产值。设计的产值翻番 Qbasic 程序：

```
REM 产值翻番 (A41)
DATA 6,8,10,12
PRINT "年增长率","翻番年数","翻番后产值"
FOR i = 1 TO 4
    READ c: s = 100: y = 0
    WHILE s < 200
        y = y + 1: s = s * (1 + c / 100)
    WEND
    PRINT c; "%", y, INT(s * 100 + .5) / 100
NEXT i
END
```

运行程序，得

年增长率	翻番年数	翻番后产值
6 %	12	201.22
8 %	10	215.89
10 %	8	214.36
12 %	7	221.07

注意：百分号“%”在程序中只能作为字符显示，不能作为运算。语句 s=s* (1+c%) 在程序中是行不通的，必须转换为 s = s* (1+c/100)。

1.4.2 求阶乘n!

试求阶乘：n!=1 × 2 × … × n（n 从键盘输入）。

求阶乘是典型的数值求积。注意避免在循环中的求积操作中累乘结果为零，累乘变量在循环之前赋初值不能赋零，通常赋 1。设计求阶乘的 Foxpro 程序：

```
* 求 n! (A42)
set talk off
? "求阶乘:n!=1 · 2 · … · n。"
input "请输入 n:" to n
t=1
for i=1 to n                          && 循环阶乘
    t=t*i
endfor
? ltrim(str(n))+"!="+ltrim(str(t,20))  && 输出阶乘 n!的结果
set talk on
return
```

运行程序，输入 n=16，得

16!=20922789888000

1.4.3 求常数e

求 e=1+1/1!+1/2!+1/3!+…+1/n!（其中 n 为从键盘输入的正整数）。

求和中每一项是乘数个数变化的积的形式，即涉及到和积综合计算处理，要注意循环中和与积的协调配合。以下 Foxpro 程序实现 e 的近似求解:

```
* 求常数 e (A43)
set  talk  off
? "求 e=1+1/1!+1/2!+…+1/n!"
input  "请输入 n:"  to  n
t=1
s=1
for  i=1  to  n
   t=t*i                                  && t 为 i!
   s=s+1/t                                && 1/i!累加到 s
endfor
? "e=1+1/1!+1/2!+…+1/"+ltrim(str(n))+"!="+ltrim(str(s,20,6))
set  talk  on
return
```

运行程序，输入 n:10，得

e=1+1/1!+1/2!+…+1/10!=2.718282

注意：如果输入的 n 较小，例如 n=4 或 n=5，所得 e 的精度会比较差。e 的精度随 n 的增加而提高。当达到某一精度的结果之后，继续增大整数 n，所得 e 的值不再变化。因为计算的精度是受计算机的具体有效数字限制的。

1.5 阶乘和数

一个正整数如果等于组成它的各位数字的阶乘之和，该正整数称为阶乘和数。

例如，145=1!+4!+5!，则 145 是一个三位阶乘和数。是否还有其它三位阶乘和数？共有多少个阶乘和数？

1.5.1 求三位阶乘和数

试求出所有三位阶乘和数：m=abc=a!+b!+c!（其中 a 为百位数字，b 为十位数字，c 为个位数字。约定 0!=1）。

1. *算法分析*

通过循环累乘设计一个求阶乘的函数（子程序）：jc(x)=x!。

对任意一个三位数 m，分解其百位数字 a，十位数字 b，个位数字 c。条件判别：若 m 等于 jc(a)+jc(b)+jc(c)，则作打印输出。也可通过 a，b，c 三重循环组合为三位数 m=a*100+b*10+c，然后作条件判别。

2. *求三位阶乘和数的 Foxpro 程序*

```
*  三位阶乘和数 (A51)
set  talk  off
for  m=100  to  999              &&  枚举三位数 m
```

```
    a=int(m/100)                && 分解百位数字 a
    b=mod(int(m/10),10)         && 分解十位数字 b
    c=mod(m,10)                 && 分解个位数字 c
    if m=jc(a)+jc(b)+jc(c)      && 阶乘和条件判别
       ? m
    endif
endfor
return

func  jc                        && 定义求阶乘函数:jc(x)=x!
para  x
p=1
for  i=1  to  x                 && 循环累乘:p=1 · 2 · … · x
   p=p*i
endfor
return  p
```

运行程序，得三位阶乘和数有: 145。可见 145 是唯一的三位阶乘和数。

3. 求三位阶乘和数的 Turbo C 程序

```
/* 求三位阶乘和数 (A52) */
long  jc(int x)                          /* 定义阶乘函数 jc(x)=x! */
{ int  i; long  p=1;
  for(i=1; i<=x; i++)  p*=i;
  return (p);}

main()
{ int a,b,c,m,n;
  printf("三位阶乘和数有:");
  for(a=1; a<=9; a++)                    /* a,b,c 分别为三位数的百位,十位,个位数字*/
     for(b=0; b<=9; b++)
     for(c=0; c<=9; c++)
        {m=a*100+b*10+c; n=jc(a)+jc(b)+jc(c);   /* 阶乘和条件判别*/
          if(m==n)  printf("%d\n", m);}
}
```

1.5.2 求所有阶乘和数

1. 算法分析

设阶乘和数的位数为 n，n 可以为 1，2，…容易证明：$n<8$。

事实上，若阶乘和数达到 8 位（或更大），阶乘和最大（即设定每位数字都为 9）也要比最小的 8 位数小，即

$$8 \times 9! = 8 * 362880 < 10000000$$

由此可见，阶乘和数最多只可能为 7 位。