计算机基础教育丛书

第三版

True BASIC 程序设计题解

谭浩强 张基温 马素霞

清华大学出版社

True BASIC 程序设计题解

（第三版）

谭浩强　张基温　马素霞

清华大学出版社

(京)新登字 158 号

内 容 简 介

本书包含 200 多个习题，大多数习题有参考答案。在参考答案中，不仅给出程序，而且按结构化程序设计方法，采取自顶向下、逐步细化的方法，详细介绍算法，并给出 N-S 流程图。

习题的类型较多，面也较广，既有复习基本知识的习题，又有难度较高的习题。

本书可作为大专和中专院校以及计算机培训班的师生学习 True BASIC 语言的参考书，也可供自学参考。

书　名：True BASIC 程序设计题解(第三版)
作　者：谭浩强　张基温　马素霞
出版者：清华大学出版社(北京清华大学校内，邮编 100084)
印刷者：北京昌平环球印刷厂
发行者：新华书店总店北京科技发行所
开　本：787×1092　1/16　印张：11.5　字数：272 千字
版　次：1997 年 4 月第 3 版　1997 年 4 月第 1 次印刷
书　号：ISBN 7-302-02455-3/TP・1240
印　数：00001—15000
定　价：12.50 元

《计算机基础教育丛书》出版说明

近年来，我国的计算机应用事业迅速发展，大批科技人员、大中学生、管理人员以及各行各业的在职人员都迫切要求学习计算机知识，他们已经认识到，计算机知识是当代知识分子的知识结构中不可缺少的重要部分。

计算机应用人才的队伍由两部分人组成：一部分是从计算机专业毕业的计算机专门人才，他们是计算机应用人才队伍中的骨干力量；另一部分是各行各业中从事计算机应用的人才。他们既熟悉本专业的业务，又掌握计算机应用的技术，人数众多，是计算机应用人才队伍的基本力量。他们掌握计算机知识的情况和应用计算机的能力在相当大程度上决定了我国计算机应用的水平。因此，在搞好计算机专业教育的同时，在广大非计算机专业中开展计算机基础教育是十分必要的。

非计算机专业中的计算机教学，无论就目的、内容、教学体系、教材、教学方法等各方面都与计算机专业有很大的不同，它以应用为目的，以应用为出发点。如果不注意这个特点，将会事倍功半。广大非计算机专业的师生、在职干部迫切希望有一套适合他们的教材，以便循序渐进地迈入计算机应用领域，并且不断地提高自己的水平。我们在前几年陆续编写了一些适合初学者使用的教材，受到广大群众的欢迎。许多读者勉励我们在此基础上进一步摸索和总结规律，为我国的广大非计算机专业人员编写一整套合适的教材。

近年来，全国许多专家、学者在这个领域作了有益的探索，写出了一批受到群众欢迎的计算机基础教育的教材。特别是全国高等学校计算机基础教育研究会作了大量的工作，在集思广益的基础上，提出了在高等学校的非计算机专业中进行计算机教育的四个层次的设想，受到广泛的注意和支持。我们认为：计算机的应用是分层次的，同样，计算机人才的培养也是分层次的；非计算机专业中各个领域的情况不同，也不能一律要求，在进行计算机教育时也应当有不同的层次。对于每一个学习计算机知识的人，还有一个由浅入深、逐步提高的过程。

我们认为，编辑出版一套全面而有层次的计算机基础教育的教材，目前不仅是十分必要的，而且是完全有条件的。在全国高等学校计算机基础教育研究会和许多同志的积极推动及清华大学出版社的大力支持下，我们决定编辑《计算机基础教育丛书》。它的对象是：高等学校非计算机专业的学生、计算机继续教育或培训班的学员、广大在职自学人员。

本丛书包括计算机科学技术的一些最基本的内容，例如计算机各种常用的高级语言、计算机软件技术基础、计算机硬件技术基础、微型计算机的原理与应用、算法与数据结构、数据库基础、计算机辅助设计基础、微机网络与应用、系统分析与设计等，形成多层次的结构，读者可以根据需要与可能选学。

本丛书的宗旨是针对广大非计算机专业的需要和特点来组织教材，从实际出发，用读者容易理解的体系和叙述方法，深入浅出、循序渐进地帮助读者更好地掌握课程的基本内容。希望我们的丛书能在这方面具有自己的风格。在实践中接受检验。

本丛书的作者大多数是高等学校中有较丰富教学经验的教师。但是，由于计算机科学技术的飞速发展以及我们的水平有限，丛书肯定会存在许多不足，丛书的书目和内容也应当不断发展和更新。我们热情地希望得到社会各界和广大读者的批评指正。

主编　谭浩强　林定基　刘瑞挺

1988.10

前　言

True BASIC 是 BASIC 的创始人 John G. Kemeny 和 Thomas E. Kurtz 于 1984 年提出的最新的 BASIC 版本。它在原来的 BASIC 语言基础上作了重大的改进与扩充，严格遵循美国国家标准 BASIC 的规定，完全能适应结构化程序设计的要求，是一种较理想的结构化、模块化的程序设计语言。True BASIC 既保持了原来 BASIC 语言容易学习、便于使用的优点，又符合结构化设计的要求，功能强，不论对计算机的初学者还是计算机的应用人员，都是十分适宜的。

1989 年，我们编写了《True BASIC 程序设计》一书（谭浩强、张基温编著，清华大学出版社出版），出版后受到读者欢迎，认为是一本全面介绍 True BASIC 的较好的教材，不少学校和计算机学习班选其为正式教材。1994 年该书修订后出了第二版。该书例题丰富，而且附有大量习题。如果能认真阅读、消化这些例题，并动手编制习题中的程序，将会大大提高程序设计能力，掌握算法的特点和应用。

为了帮助读者更好地掌握 True BASIC 程序设计，我们编写了这本《Ture BASIC 程序设计题解》，它包括 200 多个习题，并给出大部分习题的参考解答。在参考解答中，不仅给出程序，而且对多数程序还按照结构化程序设计方法扼要地分析算法，并给出相应的 N-S 流程图。这些习题和解答也可以作为例题来讲授或自学。读者在掌握一定数量的例题的基础上，有可能举一反三，提高自己处理程序设计问题的能力。

本书包含的问题面广、类型多，既包含了若干基本要求的习题，还有一些难度较大的习题，以适应不同程度的读者。有些习题可以在学完全书后再返过来做，或作为提高题选做，以锻炼自己的能力。

应当说明，本书所介绍的算法和程序并不是唯一的，只是一种参考解答。对同一个问题，可以有不同的解决方法。读者可以另外写出算法和程序，并与本书介绍的相比较，这样收获会更大。

本书可以作为学习《True BASIC 程序设计》的参考书，或作为学习任何一本 True BASIC 教材的参考书，也可以在初步学习了 True BASIC 之后单独阅读本书，以检查和提高自己掌握 True BASIC 程序设计的能力。学习和使用其它语言的读者也可以从本书得到有益的启发。

本书按内容分章分节组织，读者可以根据需要查找有关内容。这些章节是和《True BASIC 程序设计》一书的章节一致的，它包括了《True BASIC 程序设计》一书第一章到第九章的绝大部分习题的解答。对一些比较容易的习题，考虑到节约篇幅，不给出解答，读者不难自己做出答案。

本书的第一版于 1991 年出版后受到读者的欢迎，根据广大读者的意见，我们于 1995 年和 1996 年先后两次对本书进行较大的修改。在 1995 年出版的本书第二版时，除改正一些错

误之处外，还补充了第 6～9 章的习题解答。这次我们又根据读者的意见，在原书的基础上，由马素霞老师对第 1～5 章的习题解答全部重做，给出新的参考解答。第三版的质量比前两版有所提高。

本书一定还有不少缺点，请广大读者指正。

编者

1996.11

目　录

第1章 计算机算法

1.1 算法与计算机

1.1.1 有人说:“人工解题的最好方法不一定是计算机解题的最好方法。”对吗?为什么?

【解】 人工解题的最好方法不一定就是计算机解题的最好方法。

首先,人脑与电脑的物理组成不同;其次,人脑与电脑的工作方式不同。因此,人工解题与计算机解体的机制是不同的;人工解题时不仅可以利用算术方法,还可以利用公理系统进行复杂的推理及演算。而计算机并不具有思维能力,不能像人一样用公理系统按演绎方式解题。计算机只能执行数据传送、逻辑运算、算术运算、比较、判断及转移等简单的基本操作,用计算机解题的方法必须能够分解成计算机能够执行的基本操作。由于计算机所能执行的操作与人所能执行的操作不同,因此,人能执行的某些操作而计算机还不能直接执行。

计算机运算速度高,适合于执行多次重复性的操作。当求解过程能被描述为多次重复的简单操作时,则更能发挥计算机的解题优势。这一点是人所不能比拟的。

1.1.2 为什么说:“计算机科学就是研究算法的科学?”

【解】 做任何事情都需要按一定的过程进行,这些过程都包括一系列基本操作。对于不同的问题及在不同的环境下,对基本操作有不同的定义。无论基本操作如何定义,我们都将为解决问题所要进行的操作序列称为“算法”。同样,要用计算机解题,也要根据具体问题,制定解题步骤,即算法。计算机是否能够正确、可靠而高效地完成解题任务,关键取决于算法的好坏。因此,从某种意义上说,算法的研究就成为计算机科学的核心。

1.2 算法的表示

1.2.1 为什么要规定算法的三种基本结构?

【解】 早期的程序设计并不限制程序流程的控制结构,也不限制 GOTO 语句的使用,人们在设计算法时具有很大的灵活性。在设计算法时,人们更看重算法的效率及编程的技巧。由于不限制程序的控制结构,使得编出的程序往往无规律可言,程序的结构复杂、难于阅读及修改。随着程序规模的不断增加,人们在程序设计时花费的工作量越来越大,而得到的却是难于阅读、修改及维护的质量不高的产品。人们将这种现象称为“软件危机”。

解决“软件危机”的一个重要方法是:在设计算法时限定只能使用顺序、选取及循环这三种基本控制结构。由于这三种基本结构具有单入口、单出口的性质,将算法结构限制为只使用三种基本结构后,可以使算法的各基本结构之间形成顺序执行关系。这样,不同基本结构之间的关系简单,互相依赖性较少,算法呈块状结构。这样的算法容易阅读、理解和修改,算法的质量显著提高。

1.2.2 下面是打印出最初 N 个素数的算法,请用 N-S 图表示。

(1) P 置初值 2,K 置初值 1,$S=0$,输入 N 的值

(2) 打印 P

(3) 使 $P+1\Rightarrow P$

(4) 当 $K<N$ 时,重复做下面的工作,否则转(5)

(i) $2\Rightarrow i$

(ii) 当 $i\leqslant\sqrt{P}$ 且 $S=0$ 时,重复做以下工作:

(a) $\frac{P}{i}$的余数$\Rightarrow r$

(b) 如 $r\neq 0$,使 $i+1\Rightarrow i$

否则 $S=-1$

(iii) 如 $S=0$,则打印 P,$K+1\Rightarrow K$

(iv) $S=0$

(v) $P+2\Rightarrow P$

(5) 结束

请弄清楚这个算法。P 是待判断的数。K 表示已找到的素数的个数。S 作为“标志”,当 $S=-1$ 时表示 P 能被某一数整除,P 不是素数。

【解】 N-S 图见图 1.1。

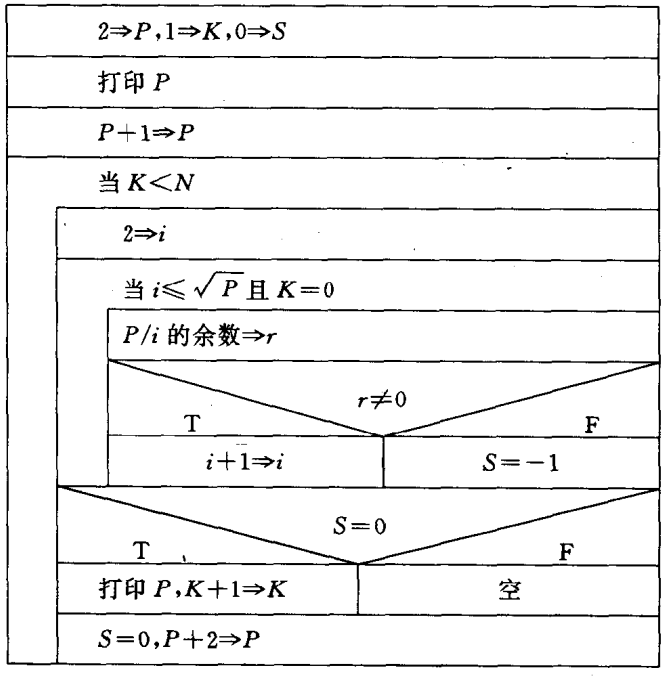

图 1.1

1.3 用“逐步细化”方法设计程序

用逐步细化的方法设计下列各题的算法(最后用 N-S 图描述)

1.3.1 计算 $S=1+2+2^2+2^3+\ldots+2^{15}$。

【解】 初步算法：

S1：0⇒S； S2：计算 S 的值； S3：输出 S 的值。

对 S2 细化：设变量 P 存每项的值，变量 i 控制循环次数。

S2.1：1⇒P，0⇒i

S2.2：当 $i\leqslant 15$，重复以下操作：

S2.2.1：$P+S\Rightarrow S$

S2.2.2：$P*2\Rightarrow P$

S2.2.3：$i+1\Rightarrow i$

算法的 N-S 图如图 1.2 所示。

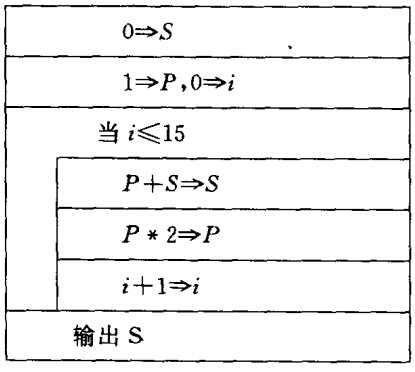

图 1.2

1.3.2 计算 $S=a_0+a_1X+a_2X^2+\cdots+a_{10}X^{10}$。

【解】 初步算法：

S1：0⇒S。

S2：计算 S 的值。

S3：输出 S。

对 S2 细化：设变量 P 存 X^i 的值，变量 i 控制循环的次数，当 i 从 0 到 10 变化时，将 a_iX^i 累加到变量 S 上。今只设一个变量 a 存系数，故 11 个系数不能同时输入，需要边输入边计算，具体算法如下：

S2.1：输入 X 的值。

S2.2：0⇒i，1⇒P。

S2.3：当 $i\leqslant 10$ 重复执行以下操作：

S2.3.1：输入 X^i 的系数 $a_i\Rightarrow a$。

S2.3.2：$a*P+S\Rightarrow S$。

S2.3.3：$P*X\Rightarrow P$。算法的 N-S 图如图 1.3 所示。

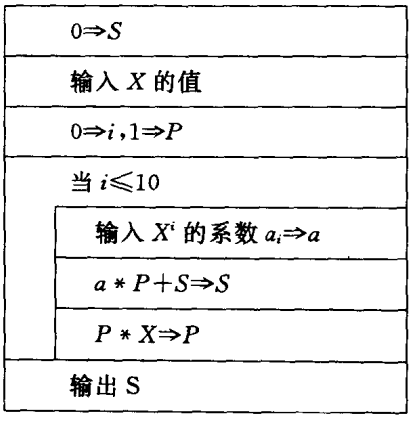

图 1.3

1.3.3 求方程 $aX^2+bX+c=0$ 的根。

【解】 算法：

S1：输入方程的系数 a,b,c，并限制 a,b 不同时为 0。

S2：如果 $a=0$，则计算一次方程 $bX+c=0$ 的根并输出。

否则，求二次方程 $aX^2+bX+c=0$ 的根并输出。

求二次方程 $aX^2+bX+c=0$ 的根的算法如下：

S2.1：求判别式 b^2-4ac 的值⇒disc。

S2.2：如果 disc＞0，则计算两个不相等的实根 $X_{1,2}=\dfrac{-b\pm\sqrt{\text{disc}}}{2a}$

如果 disc＝0，则计算两个相等的实根 $X_1=X_2=-b/(2a)$。

如果 disc＜0，计算两个虚根 $X_{1,2}=-\dfrac{b}{2a}\pm\dfrac{\sqrt{-\text{disc}}}{2a}\text{i}$

算法的 N-S 图如图 1.4 所示。其中，计算二次方程的根的算法如图 1.5 所示。

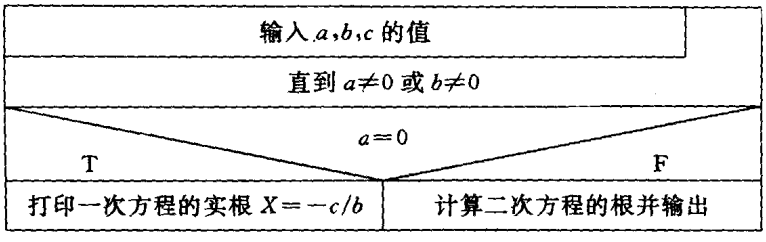

图 1.4

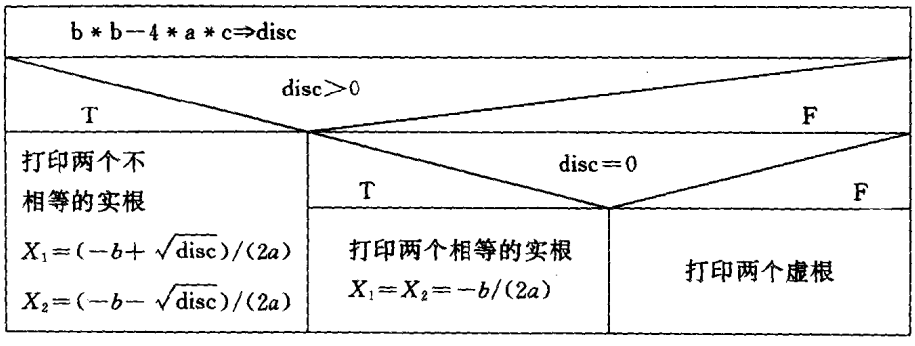

图 1.5

1.3.4 给定一个年份，判断是不是闰年？

【解】 判断闰年的条件是：被 4 整除且不能被 100 整除或能被 400 整除的年份。

初步算法：

S1：输入年份 year。

S2：判断 year 是否是闰年。

S3：输出结果。

设变量 leap 来记 year 是否是闰年。leap＝1 表示 year 是闰年；leap＝0 表示 year 不是闰年。算法的 N-S 图如图 1.6 所示。

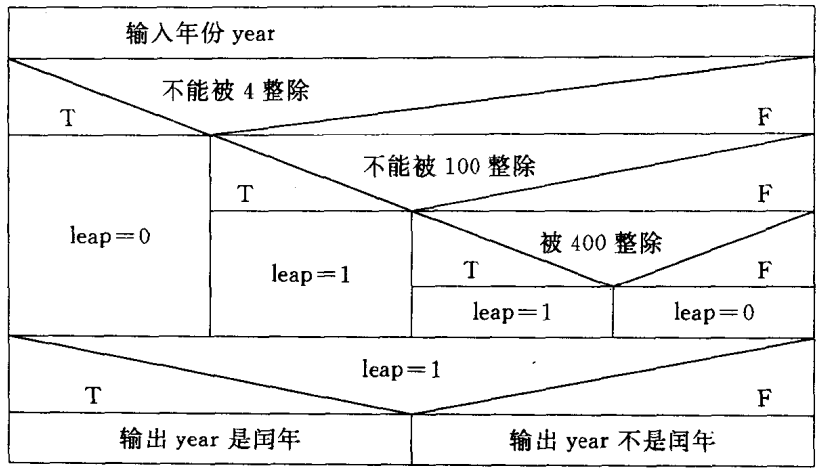

图 1.6

1.3.5 验证:任何一个大偶数都是两个素数之和。请将 6～100 之间的偶数表示为两个素数之和(如 6=3+3,8=3+5 …)。这就是著名的哥德巴赫(Goldbach)猜想命题。

【解】 初步算法见图 1.7。其中,求素数 A,B 使 $N=A+B$ 的算法如图 1.8 所示。

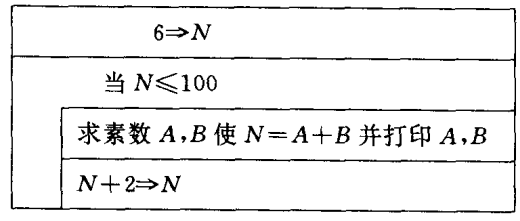

图 1.7

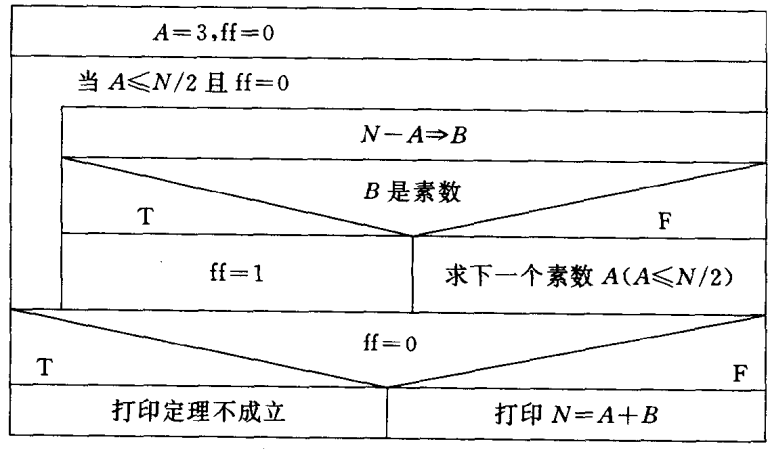

图 1.8

判断 B 是素数的算法(见图 1.9):根据素数的定义,只能被 1 及自身整除的数为素数。对任一整数 B,用 2～$B-1$ 中的每一个数去除 B,如果都不能除尽,则 B 是素数,否则 B 不是素数。但实际上,并不需要用 2～$B-1$ 范围内的所有数去除 B,而只需在 2～$\sqrt{B}$ 范围内判断即可。因为,如果存在 $B1$、$B2(B1\leqslant B2)$ 使 $B=B1*B2$,则必有 $B1\leqslant\sqrt{B}$。因此,如果 2

$\sim\sqrt{B}$中不存在整除 B 的数，则 $\sqrt{B}+1\sim B-1$ 范围内也一定不存在整除 B 的数。

设变量 f 来标识是否判断出 B 是素数，如已判断出 B 不是素数，$f=0$，否则 $f=1$。

求下一素数 $A(A\leqslant N/2)$的算法：

当 $A\leqslant N/2$ 且 A 不是素数时，重复执行以下操作：

S1：$A+2\Rightarrow A$

S2：判断 A 是否是素数。

设变量 fa 标识还未找到下一个素数 A，算法的 N-S 图如图 1.10 所示。

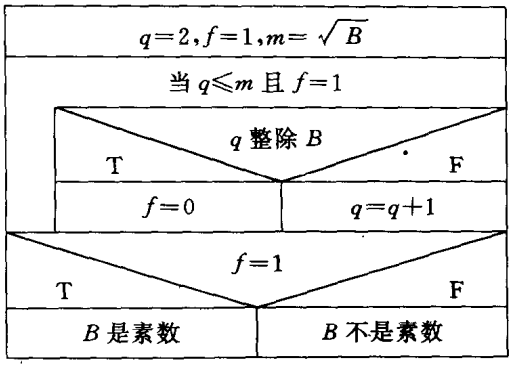

图 1.9

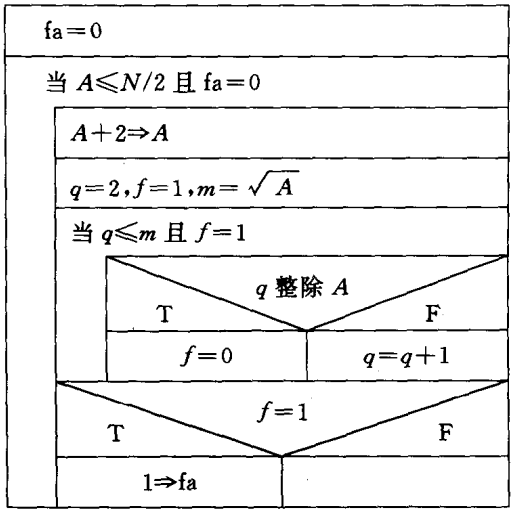

图 1.10

1.3.6 水手分椰子：有 5 个水手带 1 只猴子来到南太平洋的一个岛上，发现那里有 1 大堆椰子。由于旅途的颠簸，大家都很疲倦，很快就入睡了。第一个水手醒来后，把椰子分成 5 堆，将多余的 1 个分给猴子，他藏起 1 堆便又去睡了。第二、第三、第四、第五个水手后来也陆续起来，把见到的椰子和第 1 个水手一样重新分为 5 堆，把多余的一个给猴子，自己藏起 1 堆，又去入睡。天亮之后，大家一齐起来，把余下的椰子重新分为 5 堆，每人 1 堆，恰好余下 1 个又给了猴子。试问原先有多少椰子？

【解】 根据题意可知，此问题描述了一个递推过程。5 个人每人分 1 次，最后大家一起分 1 次，共分了 6 次，每次分后都余 1 个给猴子。

此题可采用倒推的方法。设最后每一堆的椰子数为 z 个，则第六次分之前(即第五次分之后)的椰子数为

$$5*z+1\Rightarrow y。$$

前五次分椰子时，每次都是将椰子平均分成 5 堆，自己藏起一堆，把多余的一个给猴子。设每次分之前的椰子数为 x 个，分之后的椰子数为 y 个。则有

$$4\times\frac{x-1}{5}=y$$

$$x=(5\times y)/4+1$$

因此，重复执行五次以下递推公式，就能求得原先的椰子数。其中，重复执行的条件是 $(5\times y)/4$ 为整数。

$$\begin{cases}(5\times y)/4+1\Rightarrow x\\ x\Rightarrow y\end{cases}$$

如在上面的计算过程中，$(5\times y)/4$ 不为整数，则需要改变 z 的值(可令 $z=z+1$)，并按上述方法重新计算 x 的值，直到计算出的 x 都为整数为止，则 x 是求得的一个解。

现在的问题是 z 的值未知。由于 z 的值应为整数，可先将 z 置初值 1，并设变量 f 来标识此次运算中是否存在 4 不能整除 $5\times y$ 的现象，如存在，f 的值置 1，否则，f 的值置 0。具体算法如下：

S1：$z=1$；

S2：重复执行以下操作：

S2.1：$5*z\Rightarrow y, i=1, f=0$；

S2.2：当 $i\leqslant 5$ 且 $f=0$ 时，重复执行以下操作：

如果 4 整除 $(5*y)$，则执行

$(5*y)/4+1\Rightarrow x$

$i+1\Rightarrow i$

$x\Rightarrow y$

否则，$f=1$

S2.3：如果 $f=1$，则 $z=z+1$；

直到 $f=0$ 为止。

S3：输出 x。

算法的 N-S 图如图 1.11 所示。

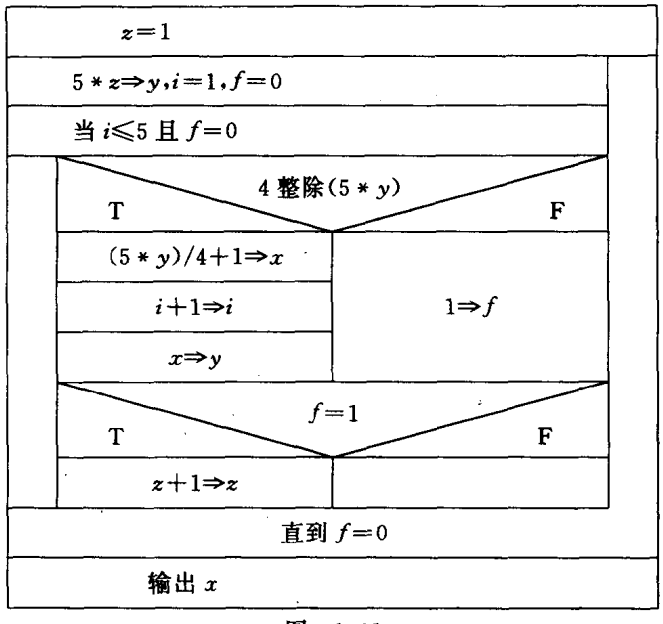

图 1.11

1.4 程序设计语言

1.4.1 一般说来，一种面向过程的程序设计语言能提供哪些功能？

【解】 一般说来，一种面向过程的程序设计语言应能够提供数据描述、数据运算、数据传送、流程控制、过程抽象等几种功能。

第2章　True BASIC程序设计初步

2.1　概　　述

2.1.1　试述True BASIC程序的组成。

【解】（略）

2.1.2　“注释是程序中无关紧要的部分”，这句话对吗？为什么？

【解】　这句话不对。注释虽然是程序不可执行的部分，它对程序的运行结果和运行速度不产生影响，但它能帮助人们阅读和理解程序，提高程序的清晰性及可维护性。

2.1.3　什么叫窗口？True BASIC能提供几种窗口？它们各有何用途？

【解】　窗口是在屏幕上开设的一块显示区域。True BASIC能提供两种窗口：编辑窗口及背景窗口。编辑窗口用来显示及编辑True BASIC源程序。背景窗口用来键入命令、显示错误信息、提示信息及运行结果。

2.1.4　解释执行方法与编译执行方法各有何特点？True BASIC如何把它们统一起来？

【解】　解释执行方式的特点是翻译一句执行一句，不产生整个目标程序。调试程序时用解释执行方式比较方便，但执行速度慢。编译执行方式的特点是将整个程序全部翻译成目标代码，并产生目标程序。目标代码的运行速度高，所占空间比源程序代码要少得多。

True BASIC允许解释与编译并用。用户将源程序输入编辑窗口后，在背景窗口键入命令“RUN”即进入解释执行状态。用解释执行方式将程序调试好后，键入命令“COMPILE”将程序翻译成目标代码（在翻译之前最好用“SAVE”命令保存源程序，否则源程序有可能被破坏），再键入命令“RUN”即开始执行。

2.2　数据描述

2.2.1　什么叫数据？数据类型有什么意义？True BASIC能提供哪几种数据类型？

【解】　我们将描述客观事物的数、字符及所有能输入计算机并为计算机系统所处理的符号系统称为数据。

在高级语言程序中，每一个数据都属于一个特定的类型。在同一个高级语言中，不同类型的数据的表示、存储及操作的种类是不相同的。例如，在True BASIC程序中，字符型数据需用双引号括起来，在存储时一个字符占一个字节的存储空间，且只能进行比较、连接、取子字符串等运算，而不能像数值型数据那样进行算术运算。因此，类型信息可用于检查和防止程序中无意义的操作，也可用于确定怎样表示及用哪一类操作对该数据进行处理。True BASIC能提供基本类型（数值型和字符型）及构造类型（数组和文件）。

2.2.2　什么叫数据结构？为什么进行程序设计时还要确定数据结构？

【解】　反映数据的属性及其数据间联系的数据模型，称为数据结构。