高等学校试用教材

True BASIC 语言模块化结构化程序设计

邓德祥

高等教育出版社

True BASIC 语言模块化结构化程序设计 邓德祥

高等学校试用教材

True BASIC 语言
模块化结构化程序设计

邓 德 祥

高等教育出版社

内 容 提 要

本书全面地介绍了符合美国国家标准的开拓 BASIC 新纪元的 True BASIC 语言，同时介绍了结构化程序设计的基本方法和结构化流程图，以及在 IBM 微型机上True BASIC的上机操作方法。

本书以结构化程序设计为主线，针对初学者的特点对各部分内容作了较合理的安排。本书的特点是由浅入深、循序渐进、例题丰富、概念清楚、言简意明。学习本书可满足各种科学计算、数据处理和画图的需求。

本书可作为高等学校计算机基础教育的教材，也可供中专、职工大学、职业高中和短训班使用，还可供各类工程技术人员、管理干部自学或参考。

高等学校试用教材

True BASIC 语言

模块化结构化程序设计

邓 德 祥

*

高等教育出版社出版

新华书店北京发行所发行

河北省香河县印刷厂印装

*

开本 787×1092 1/16　印张 15.25　字数 350 000

1988 年 10 月第 1 版　1988 年 10 月第 1 次印刷

印数 0 001—9 110

ISBN7-04-001634-6/TP·25

定价 3.60 元

前　言

BASIC 语言是美国计算机语言学家 John G. Kemeny 和 Thomas E. Kurtz 吸收FORTRAN IV 和 ALGOL60 语言的优点于 1964 年推出的. 由于它具有小型、通用、会话、易学的特点，因而很快得到了推广和应用. 但由于它是非标准化的语言，因而 BASIC 程序通用性较差. 另外，近年来结构化程序设计方法已成为受软件工作者欢迎的程序设计方法，但原 BASIC 语言功能差，不能适应结构化程序设计方法的要求.

1984 年美国公布了"美国国家标准 BASIC 程序设计语言". Kemeny 和 Kurtz 等于 1985 年推出了严格按照美国国家标准实现的模块化、结构化程序设计语言——True BASIC.

True BASIC(真的 BASIC)保持了原 BASIC 语言"小型、通用、会话、易学"的特点. 和当代其它的高级语言比较，它具有语言功能强、语法规定自然、程序结构清晰的特点，而且具有良好的操作环境. 因此 True BASIC 被誉为"开拓 BASIC 新纪元的软件". 学会了 True BASIC 完全可以满足各种科学计算、数据处理和画图的需要. 为使广大科技工作者、管理干部和青年能尽快地掌握它，作者对 True BASIC 软件进行了消化和开发，编写调试了各种程序一百多个，基于多年教学和编写教材的经验，按照高等学校教学的要求，写成了本书奉献给读者.

本书包含了 True BASIC 语言、结构化程序设计方法和上机操作方法三部分内容. 结构化程序设计能力的培养是本书的出发点和归宿. 本书以结构化程序设计为主线，根据程序设计的要求逐步引入了各种语句. 全书共分 12 章. 第 1，2 章讲述了学习本书的基本知识. 第 3 章到第 5 章分别讲述了顺序、选择和循环结构程序设计. 第 6 章讲述了数组与矩阵运算. 第 7 章讲述了画图功能. 第 8 章讲述了函数、子程序和图画三种模块和库的设计. 第 9 章讲述了正文文件、字节文件和记录文件的使用. 第 10 章讲述了音响程序设计. 第十一章讲述了其它程序设计，供进一步提高时阅读. 第 12 章讲述了上机操作方法，包括IBM 微机系统介绍，操作系统基本命令，True BASIC 程序的输入、调试和运行，以及汉字的输入和输出.

本书内容齐全，例题丰富，由浅入深，循序渐进，可作为高等学校计算机基础教育的教材，也可供职工大学和有关短训班使用，还可供各类工程技术人员自学或参考.

全国高等学校计算机基础教育研究会副会长谭浩强教授担任了本书的主审工作，北京自动化学院田淑清副教授阅读了书稿，北方工业大学田成方教授阅读了初稿，他们都给了很大支持，并提出了许多宝贵意见，在此表示诚挚的感谢.

由于作者水平有限、资料甚少、时间仓促，错漏和不妥之处在所难免，敬请各位读者批评指正.

作者

于1988.2

目　录

第1章　计算机、语言和程序设计

随着计算机的普及和推广，有愈来愈多的人迫切地希望学会使用计算机．人们步入“计算机世界”，自然要使用一种工具与计算机交流信息，这个工具通称为计算机语言．作科学计算和事务处理，通常使用过程型命令语言，人们不仅要告诉计算机“做什么”，还要告诉计算机“怎样做”．怎样做的步骤要用一系列的语句(或指令)表示，这个能完成一定任务的语句(或指令)序列就叫程序．而编写程序的过程就叫程序设计．计算机的一切操作是按程序执行的．所以，要想使用电子计算机，就要学会计算机语言程序设计．

BASIC 语言是国内外广泛应用的计算机语言之一．True BASIC 是 BASIC 创始人 John G. Kemeny 和 Thomas E. Kurtz 等人于 1985 年严格按照 1984 年“美国国家标准 BASIC 程序设计语言”实现的．它一方面吸收了现有计算机高级语言的优点，另一方面又保持了 BASIC 的小型、通用、会话、易学的特点．所以命名为真(true)BASIC．True BASIC 被誉为开拓 BASIC 语言新纪元的软件，它是模块化、结构化的程序设计语言．

这一章先简单地介绍一下计算机的组成，计算机语言和程序设计，为学习 True BASIC语言程序设计奠定基础．

1.1　计算机简介

世界上第一台电子计算机 ENIAC 是 1946 年问世的．计算机的发展非常迅速，应用特别广泛，对于人类社会的发展起着巨大的推动作用，致使人类社会进入了一个新阶段——“信息社会”．

1.1.1　计算机的特点

(1) 具有计算能力．计算速度快、精度高、可靠性强．

计算机的计算速度已经可以达到每秒几十亿次，一般为每秒几百万次到每秒几千万次．这样的速度是人手工计算不能比拟的．计算机的计算精度一般为有效数字 6 位，也可以为十几位，甚至上百位．这也是任何其它计算工具所望尘莫及的．计算机的可靠性很强，它可以连续工作几个月甚至几年不出错误．

(2) 具有记忆能力．计算机能存储程序和数据．

(3) 具有逻辑判断能力．

计算机能对两个信息进行比较，根据比较结果决定下一步的操作，实现自动控制．

概括地说，计算机是一种能高速、自动、准确进行“计算”(信息处理)的电子设备．已被广泛

地用于科学计算、数据处理(DP)、自动控制、人工智能(如机器人、专家系统)和计算机辅助设计(CAD),计算机辅助教学(CAI)等方面.

1.1.2 计算机的组成

计算机并不神秘,它的组成可分两大部分,一部分是机器系统(硬件),另一部分是程序系统(软件).

(1) 硬件.

如图 1-1 所示,其中←——表示数据传送路线,←┈┈表示控制信号传送路线. 计算机硬件分五大部分.

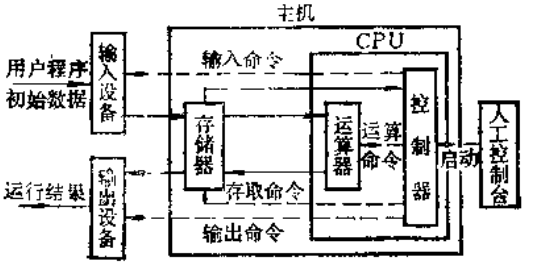

图 1-1 计算机硬件结构示意图

a) 运算器. 主要用来做算术运算和逻辑运算,相当于算盘.

b) 存储器. 用来存储程序和数据,具有记忆功能,相当于纸. 存储器的特点是"取之不尽,存之更新".

c) 控制器. 通过程序的执行,控制、协调各部分的操作.

d) 输入设备. 用来输入程序和初始数据等,常用设备有键盘、磁盘机、磁带机、卡片输入机和光电机(纸带)等.

e) 输出设备. 用来输出程序清单、计算结果等. 常用设备有显示器、打印机、磁盘机、磁带机、卡片输出机、穿孔机(纸带)和绘图仪等.

运算器和控制器合起来叫中央处理机(CPU). CPU 加上存储器叫主机. 输入和输出设备统称为外部设备. 计算机的机器系统又称为裸机,或称为硬件.

一台裸机要正常工作和充分发挥效能,还需要配上功能完善的程序系统(软件),以使计算机能按照用户的指令自动地进行各种操作.

(2) 软件.

确切地说,软件是程序系统和文档资料(有关程序资料及使用说明等)的总称. 这里是把文档资料看作程序不可缺少的附件,关于这一点今后不再说明. 简单地说,软件是程序的总称,它可分为:

a) 系统程序(系统软件). 常指与硬件配套出售的基本软件,包括管理整个计算机的操作系统,翻译各种语言的编译系统以及检查、诊断计算机自身的服务系统等.

b) 应用程序(应用软件). 为便于使用计算机而开发的程序,通常以软件包的形式出现. 例如,数学计算软件包、绘图软件包和企业管理软件包等.

有了硬件和软件就构成了计算机完整的系统. 那么这样一台计算机又如何工作的呢?

1.1.3 计算机运行的示意过程

如图 1-1 所示,计算机运行过程可分为如下几个步骤:

a) 启动(开机). 通过人工控制使机器进入系统软件控制之下.

b) 输入用户程序和初始数据. 使用计算机的人用计算机语言表示的解题步骤称为用户

程序.

c) 运行用户程序,输出计算结果.

d) 如果还有程序运行则重复第二、第三步,直至所有程序运行完毕

e) 关机.

1.2 计算机语言概述

1.2.1 机器语言

如前所述,计算机语言是人与计算机交流信息的工具.这种工具最初级的形式就是二进制代码,即按特定规则由0和1组成的一定位数的数字代码.它能控制计算机进行特定的动作(例如加、减、取数、存数等).由于这种二进制代码能被计算机识别,所以称为机器指令.这些指令的集合称为机器语言.机器语言是因机而异的,就是不同机型,机器语言不同,因此可移植性差.用机器语言写出来的程序称为机器语言程序.这种程序的优点是计算机能直接运行,效率高.例如,求两个数值之和(56+43),用Z-80指令编写的程序如表1-1所示.

表1-1 计算56+43的Z-80机器语言程序

存储单元地址(16进制数①)	存储单元内容(二进制)	说明
00 40 00 41	00 11 11 10 00 11 10 00 (56)	取56送A累加器
00 42 00 43	00 00 01 10 00 10 10 11 (43)	取43送B寄存器
00 44	10 00 00 00	把B加到A上
00 45 00 46	00 11 00 10 01 00 10 00	把A内容送到0048单元
00 47 00 48	00 00 00 00 01 10 00 10 (99)	空操作(存结果的单元)
00 49	01 11 01 10	停机

① 16进制的基数 0 1 2 3 4 5 6 7 8 9 A B C D E F

对应的10进制数 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

作为一个用户,最关心的是如何学习和使用计算机语言.通过上述小小的例子可以看出,机器语言难学难用,而且易错难改,十分枯燥无味.为了便于记忆,计算机专家们又创造了汇编语言.

1.2.2 汇编语言

汇编语言是用特定的助记符号代表数字代码,帮助人们记忆,它和机器指令基本上是一一对应的.这个改进不算大,所以人们把机器语言和汇编语言通称为低级语言.上述计算用Z-80汇编语言编写的程序如表1-2所示,汇编程序语义和机器语言程序一一对应.

表 1-2　计算 56＋43 的 Z-80 汇编程序

汇编语言程序	说　　明
START'LD A, 38H'	38H 表示38是16进制数
LD B, 2BH	2BH表示2B 是16进制数
ADD A, B	A 为累加器
LD (18H), A	B 为寄存器
NOP	空语句
HALT	们语句

1.2.3　高级语言

第一台电子计算机问世后，又经过十年的努力，到 1956 年，在 IBM704 机上实现了 FORTRAN1. 这是由低级语言到高级语言的飞跃. 高级语言之所以称之为"高级"是因为它是用英语和人们熟悉的数学公式来表达的，并且又能为计算机所接受，具有较好的通用性. 上述例子的 True BASIC 程序，如表 1-3 所示.

表 1-3　计算 56＋43 的 True BASIC 程序

True BASIC程序	说　　明
PROGRAM ex01	程序开头语句
LET a=56	把56赋给 A
LET b=43	把43赋给 B
PRINT a+b	打印出A＋B的值
END	结束

这里存在一个问题，即上面提到电子计算机能直接识别二进制代码，但又如何识别这些字符呢？事实上，高级语言程序，通常称为源程序，输入计算机之后，先要被翻译成机器语言程序，通常称为目的程序，然后再执行. 这个翻译工作是由一个系统软件完成的.

True BASIC 有两种方式：一种叫解释程序，这类似于口头翻译，它是边解释边执行，直接用 RUN(或按 F9)命令运行. 另一种叫编译程序，这类似于文件翻译，生成一个机器语言的目标程序，再用RUN(或按 F9)命令运行.

编译程序的功能，如图 1-2 所示.

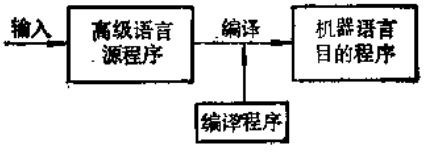

图 1-2　编译程序的功能

目前国内外常用的高级语言有十几种，分别具有不同特点，例如：

BASIC　　(小型会话式语言)

True BASIC　　(模块化结构化程序设计语言)

FORTRAN 66　　(适用于科学计算)

FORTRAN 77	(适用于科学计算、数据处理)
COBOL	(适用于数据处理)
PL/1, ALGOL 68	(大型通用语言)
PASCAL	(结构化程序设计语言)
C	(系统程序设计语言)

1.2.4 超高级语言

高级语言经过 30 年的发展，目前可以说已经接近登峰造极的地步，Ada 语言的出现就是一个明显的标志. 高级语言是过程型命令语言，以赋值为核心，适用于数值/数据处理. 而超高级语言是非过程型命令语言，有逻辑型、函数型以及逻辑/函数混合型等. 以 PROLOG 语言为例，它是以模式匹配与回溯求解为核心，适用于知识信息处理. 它一改传统过程语言的风格，人们只需要把问题所涉及的事物之间的逻辑关系描述清楚，而不必给出求解问题的具体步骤，只要向系统提出问题，系统将自动地求出解答. 1981年日本宣布：他们研制的第五代计算机已将PROLOG语言选作核心语言.

1.3 程序设计概述

如前所述，学习计算机语言的目的是用它来编写程序，解决科学计算、事务处理等问题. 下面通过一个例子说明用计算机解题过程，例题中的一些细节，当前不必深究，以后自然明白.

1.3.1 明确问题

在着手解决问题之前，首先要弄清问题的条件和最终要求.

例如，“若 A，B 两点有着不可逾越的障碍，求 A，B 两点之间的距离”. 这样的问题条件和要求都很明确. 如果问题是“求两点之间的距离”，要求虽明确，但条件不太明确. 若能直接用米尺测量，那么，一般无需编程序用计算机计算了.

1.3.2 建立模型

一个现实的问题，往往比较复杂，其影响因素是多方面的，这时要进行试验分析，抓住主要矛盾，丢掉次要矛盾，把原来的问题化作一个理想的模型. 例如，对于上述测量 A，B 两点间距离这样一个问题，根据测量学，选择一点 C，如图 1-3 所示.

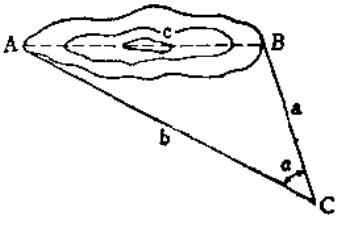

图 1-3

于是可测得：

$$a=1.5,\quad b=3.2,\quad c=0.8\text{(弧度)}$$

这样就把实际问题化为“已知三角形两边和夹角，求第三边 c” 的数学问题了. 这一步也叫建立数学模型.

1.3.3 算法设计

所谓算法设计，就是把数学模型规定的任务，转化为具体的解题步骤，即产生解题的逻辑蓝

图. 如果说数学模型规定“做什么”，那么算法设计就是解决“怎样做”. 例如对于上述问题，由三角学知道，

$$c=\sqrt{a^2+b^2-2ab\cos\alpha}$$

把已知值代入，即可算出 c.

1.3.4 编写程序

编写程序就是用计算机语言把算法描述出来，如上述问题用 True BASIC 语言可编程序如下：

```
1  PROGRAM ex01-1
2  REM computing the side of a triangle
3        PRINT "a, b, alpha? "
4        INPUT a, b, alpha
5        LET d=a*a+b*b-2*a*b*cos(alpha)
6        LET c=sqr(d)
7        PRINT "a=";a, "b=";b,"alpha=";alpha,"c=";c
8  END
```

True BASIC 程序一句一行，一行也只能写一句. 第 1 行是主程序开始语句；第 2 行是注释行；第 3 行是打印语句，输出一个字符串，作为第 4 行的提示；第 4 行是键盘输入语句；第 5,6 行是赋值语句；第 7 行打印输出结果；第 8 行结束语句.

1.3.5 编译运行

这一步也叫上机操作. 第一步把编好的程序输入计算机内存；第二步编译运行：在系统软件支持下自动地先检查语法，后执行. 如果发现“编译错误”(语句不符合语法规定)，则编译中断，这时光标移至程序出错处，并给出“出错信息”. 根据出错信息提示，进行修改，然后再运行，逐步去掉语法错误. 上述程序运行时，

显示：a, b, alpha?

输入：1.5, 3.2, 0.8↵(↵ 表示 enter 键)

显示：a=1.5 b=3.2 alpha=0.8 c=2.408

如果输入数据不合理，还会出现“运行错误”(在程序运行过程中出现的错误)，如溢出等.

1.3.6 结果分析

通过结果分析，可以检查是否还隐藏着逻辑性错误（程序未准确表示算法或题意，如用 0 除等）. 上述解题过程，可以概括地用图 1-4 表示.

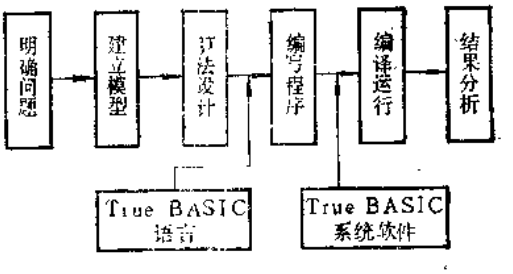

图 1-4 用计算机解题步骤

本课程的重点是讲述 True BASIC 语言和程序设计，其它步骤也会涉及到，但不作为重点. 例如编译运行，将涉及到计算机原理，操作系统和编译方法等课程的知识，已超出本书范围，不宜详述.

1.4 程序设计的辅助手段——控制流程图

对一个稍微复杂的问题，根据算法直接写出程序是不太容易的，人们常用图形把解题步骤形象直观地描述出来，这样的图形就叫控制流程图。简单地说，流程图是算法的逻辑蓝图，是算法与程序之间的桥梁。画流程图是编写程序和阅读程序时常用的辅助手段。除此之处，还可采用 N-S 图，或伪代码等辅助手段。

例如，已知三角形的两边(a 和 b)和夹角(α)，求第三边(c)，初始数据如下：

a	b	α 弧度
1.2	2.0	0.2
2.0	4.0	0.5
0.0	0.0	0.0(结束标志)

可画流程图，如图 1-5 所示。

按图 1-5 可编程序如下：

```
1   PROGRAM ex01_2
2   REM computing the side of a triangle
3   DECLARE DEF side
4   PRINT "    a     b     alpha"
5   INPUT a,b,alpha
6   DO while a>0 and b>0 and alpha>0
7         LET c=side (a,b,alpha)
8         PRINT a,b,alpha,"c=";c
9         INPUT a,b,alpha
10  LOOP
11  END
12  ! function subprogram
13  DEF side (a1,b1,,alpha1)
14        LET d=a1*a1+b1*b1-2*a1*b1*cos(alpha1)
15        LET side=sqr(d)
16  END DEF
```

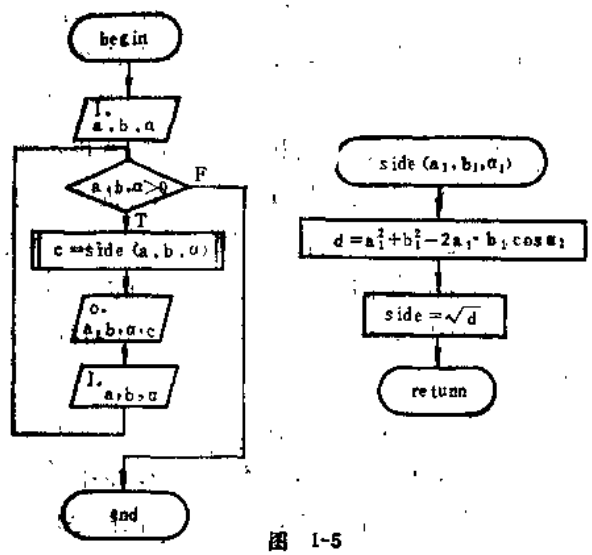

图 1-5

在流程图中，程序的逻辑关系和处理顺序是以一个框代表一段程序(包括一个或几个语句)，以箭头表示程序执行方向。在程序设计时，流程图可先粗后细，逐步细化，直到可方便地写出程序。在阅读别人的程序时，也可以把程序执行过程用流程图表示出来，便于阅读。总之借助流程图编写或阅读程序，能使程序逻辑关系清楚，层次分明，减少差错。图 1-6 是流程图中常用的符号，这与 1963 年美国标准协会电子计算机和信息处理委员会发布的一套流程图符号基本一致。在

图 1-6 中(a)为矩形框，亦称“处理框”，表示一般处理功能；(b)为菱形框，亦称“判断框”，表示在几个可以选择的路径中，判断选择某一路径；(c)为两头尖框，表示循环；(d)为平形四边形框，表示输入/输出，本书用在平行四边形框中写入 I 或 O，分别表示输入(INPUT)或输出(OUTPUT)；(e)为圆弧边框，亦称“端框”，表示流程开始或结束；(f)为圆圈，表示连接点；(g)为双线矩形框，表示调用辅程序；(h)为指向线，表示流程的路径和方向.

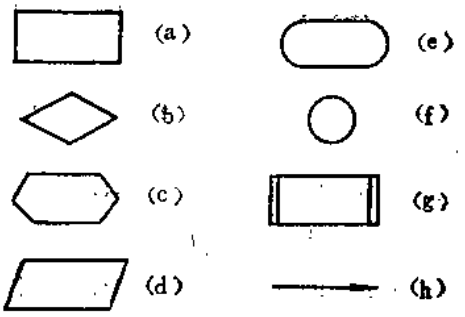

图 1-6 流程图常用符号

说明：(1)框内注释文字没有统一规定，以表达意思清楚为准。(2) 流程图符号中没有非执行语句可用的符号，通常也不允许把它们填入任何流程图符号中.

1.5 结构化程序设计简介

在 60 年代末以前，由于计算机的运算速度低，存储量小，评价一个程序的优劣总是把效率放在首位. 为了提高效率，程序员常常为减少一条语句或节约一个存储单元苦思冥想，结果程序越好越难读. 随着计算机科学的发展，一方面计算机运算速度和存储容量千百倍地提高；另一方面程序规模越来越大，验证和维护程序的费用不断上升. 因此人们评价一个程序的优劣，已不再把效率放在首位，而是把易读易验证和易维护放在优先的地位. 为此，荷兰学者戴克斯特拉(E. W. Dijkstra)提出了“结构化程序设计”(Structured programming). 简要地说，结构化程序设计是“按照一组能够提高程序易读性与易维护性的规则而进行程序设计的方法”. 从 60 年代末期到现在，结构化程序设计已发展成为一套系统的科学的方法，得到了软件工作者的广泛承认和欢迎.

“工欲善其事，必先利其器”，为了实现结构化程序设计，要选择合适的计算机语言. 基本 BASIC, FORTRAN 66, 都不能适应结构化程序设计的要求，FORTRAN 77 也是非完全结构化的语言，即使是标准 Pascal——结构化程序设计语言，也不如 True BASIC 结构化程度高.

1.5.1 三种基本结构

1966 年，Bohm 与 Jacopini 在一篇文章中指出，任何程序的逻辑均可用顺序、选择、循环三种基本结构或它们的组合表示. 三种基本结构的流程图如图 1-7 所示.

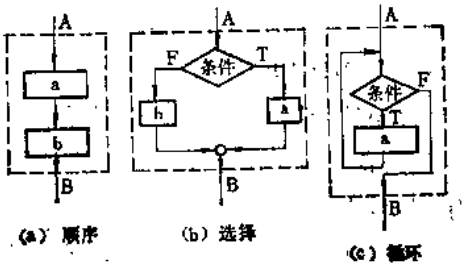

图 1-7 三种基本结构的流程图

(1) 顺序结构.

如图 1-7(a)所示，它有一个入口 A，一个出口 B. 在这个结构中的各块是按它们出现的先后顺序依次执

行的. 这是一种最基本、最简单的结构. 关于"块"的解释,待介绍了三种基本结构之后再作进一步描述.

(2) 选择(分支)结构.

如图 1-7(b)所示,它有一个入口 A,一个出口 B. 选择控制结构执行过程是,如果条件成立(为真),则执行 a 块,否则执行 b 块,再转到出口. 总之,选择控制结构,只选择其中一个分支执行一次.

(3) 循环(重复)结构.

如图 1-7(c)所示,它有一个入口 A,一个出口 B. 循环结构执行过程是:当满足给定的条件时,则重复执行 a 块,否则转到出口 B. 从图 1-7(c)可以看出,循环是选择结构的一种特殊形式,其中一个分支趋向出口,而另一个分支返回结构入口,从而构成重复执行.

"块"可以是一个可执行语句(控制转移语句除外),或一个空块(即无任何可执行的语句),或三种基本结构之一. 由于基本结构可以嵌套基本结构,所以,不难理解上述"任何程序的逻辑均可用三种基本结构或它们的组合来表示".

1.5.2 结构化程序设计准则

一个结构化程序只能由以上三种基本结构组成,并应当满足以下条件:

(1) 单入口,单出口.

(2) 对每个结点(语句或块),都应当有一条从入口到出口的通路通过它,即不允许有永远执行不到的可执行语句.

(3) 不包含无限循环,即执行的时间是有限的.

后两个条件也是传统的非结构化程序所要求的,所以结构化程序的特点在于,它的基本结构都能满足单入口、单出口的条件.

1.5.3 结构化程序设计方法

人们认识解决问题的方法,从逻辑上说,粗略地分为两种,一是从部分到整体,"自底向上",即综合法;二是从整体到部分,"自顶向下",即分析法. 结构化程序设计方法是属于后一种,就是从问题出发,由粗到细,逐步细化,直到很方便的写出程序为止. 概括地说,结构化程序设计方法就是"自顶向下,逐步求精".

1.5.4 结构化流程图——N-S 图

前面我们介绍了一般流程图符号及使用方法,这种流程图有个指向线(即箭头),使得流程图的画法很自由、灵活,非常适合非结构化程序设计使用,因而过去使用很广泛. 它的缺点是:由于流程图的画法过于灵活,使得同一个解题过程(算法)可能会有多种画法,构成多种多样的流程图. 这样,就使得流程图与程序之间的对应关系不那么直观了.

结构化程序设计方法提出之后,为了适应结构化程序设计方法的要求,于 1973 年由 Nassi 和 Shneiderman 首先开发了结构化流程图,这种框图又称为 N-S 图. 本书使用的 N-S 图是在 Nassi 和 Shneiderman 框图的基础上形成的,它正确地表现了"每个基本块是单入口单出口"的结构化程序设计基本原则. N-S 图的优点是:画起来简便,而且和程序有良好的一致性. 例如用