FORTRAN
语言程序设计

许钦明　庄长淞　范承亚　编

石油大学出版社

FORTRAN 语言程序设计

庄长淞　范承亚　许钦明　编

石油大学出版社

内 容 简 介

本书以 ANSI FORTRAN 77 内容为基础，深入浅出地介绍 FORTRAN 语言的基本语法规则和程序设计技巧，以及用 FORTRAN 语言进行结构程序设计的方法。全书共十章。第一章电子计算机的一般知识，第二章 FORTRAN 语言概貌，第三章顺序程序设计，第四章分支程序设计，第五章数组与循环程序设计，第六章函数和子例程，第七章字符处理，第八章文件系统和格式 I/O 补充，第九章公用、等价和数据块子程序，第十章程序结构和结构程序设计。可供各类大专院校教学或工程技术人员自学使用。

FORTRAN 语言程序设计

庄长淞　范承亚　许钦明　编

*

石油大学出版社出版

山东省　东营市

石油大学出版社激光排版室排版

石油大学印刷厂印刷

全国新华书店发行

*

开本 787×1092　1/16　18.125 印张　427 千字

1993 年 4 月第 1 版　1991 年 12 月第 2 次印刷

印数 5001—11000 册

ISBN 7-5636-0170-8/TP·Q6

定　价：8.50 元

前　言

FORTRAN 语言是科学与工程计算领域使用最为广泛的一种高级语言，它是现代科技人员掌握使用计算机的重要语言工具。因此，在高校各理工科专业，它是对学生进行计算机基础教育的最主要课程之一。

本书是作者在石油大学讲授 FORTRAN 语言课十几年经验基础上编写的。原教材已在石油大学校内使用十余年，先后修订印刷过三版，这次重新编写正式出版，内容上又作了较大的充实提高。

本教材主要根据 ANSI FORTRAN 77 标准(ANSI X3.9—1978)编写，但同时也兼顾了部分至今仍在使用的 FORTRAN 66 编译系统。由于 FORTRAN 77 标准基本上与 FORTRAN 66 兼容(向上兼容)，因此，书中所介绍的内容，绝大部分同时适合于 FORTRAN 66 和 FORTRAN 77 两种系统。这部分内容，书中一般以“FORTRAN 标准规定”之类语言描述。而对于只适合于 FORTRAN 77 的部分，则一般均明确指出为“FORTRAN 77 标准规定”。此外，为便于读者掌握，附录三中还给出 FORTRAN 77 和 FORTRAN 66 的标准对照表。

结构程序设计是当代公认的一种好的程序设计方法，已在各软件开发领域广泛应用。为适应程序设计技术的这一发展形势，本教材在介绍程序设计方法时，始终贯穿“结构化”这一基本思想，使初学者在学习与实践中逐步养成良好的习惯，从一开始就建立起良好的程序设计风格，最后再上升为理论。

全书共分十章。第一章电子计算机的一般知识；第二章 FORTRAN 语言概貌；第三章顺序程序设计；第四章分支程序设计；第五章数组与循环程序设计；第六章函数和子例程；第七章字符处理；第八章文件系统和格式 I/O 补充；第九章公用、等价和数据块子程序；第十章程序结构和结构程序设计。教材内容约需 64 个讲授学时。对学时不足的专业，书中带有 * 号的章节可根据学时酌情删减。

为便于学生上机实习和广大科技人员自学，并指导学有余力者进一步提高，本教材还配有相应的一套〈FORTRAN 习题集及上机操作指导〉。习题集部分每章都包含有内容提要，并备有各种不同难度的习题，部分习题附有答案。实习操作指导部分包含有 VAX 11 VMS 系统及 IBM PC DOS 系统的详细上机操作说明。

本书第二、三章由范承亚执笔，第四、五章由许钦明执笔，第三章 § 3.5 节及第一、六、七、八、九、十章由庄长淞执笔，最后由庄长淞统编全稿。

作者对本书的编写，主观愿望是力求做到概念叙述准确，方法条理清楚。在语言叙述上尽可能做到既通俗易懂，又简炼明确。力争以最少的篇幅把最有用的内容介绍给读者。但由于水平所限，书中缺点和错误恐怕也在所难免。因此，恳切希望同行和广大读者批评指正。

石油大学计算机系和郑州轻工业学院控制工程系的领导和教师对本书的出版给予了

热情的关心和支持，谨此致谢！

编者

1990 年 9 月

目　录

第一章　电子计算机的一般知识 …… (1)
§1.1　电子计算机的一般特点 …… (1)
§1.2　电子计算机系统的组成 …… (1)
§1.3　算法和算法框图 …… (3)
§1.4　程序设计和程序设计语言 …… (8)
§1.5　计算机的作业处理方式 …… (10)
练习一 …… (11)
第二章　FORTRAN 语言概貌 …… (12)
§2.1　FORTRAN 语言发展简史 …… (12)
§2.2　FORTRAN 程序举例 …… (12)
§2.3　FORTRAN 语言的基本成分 …… (18)
练习二 …… (34)
第三章　顺序程序设计 …… (36)
§3.1　终端的输入/输出语句 …… (36)
§3.2　算术赋值语句 …… (37)
§3.3　END、STOP 和 PAUSE 语句 …… (39)
§3.4　PROGRAM、DATA 和 PARAMETER 语句 …… (41)
§3.5　格式 I/O 初步 …… (44)
§3.6　顺序程序设计举例 …… (61)
练习三 …… (64)
第四章　分支程序设计 …… (67)
§4.1　算术关系表达式和逻辑表达式 …… (67)
§4.2　逻辑条件语句 …… (71)
§4.3　无条件转语句(GO　TO 语句) …… (74)
§4.4　块 IF 语句 …… (76)
§4.5　分支程序设计举例 …… (85)
*§4.6　算术条件、计算转、赋标号与赋值转语句 …… (88)
练习四 …… (92)
第五章　数组与循环程序设计 …… (94)
§5.1　数组 …… (94)
§5.2　WHILE 循环 …… (100)
§5.3　UNTIL 循环 …… (103)
§5.4　DO 循环 …… (106)
§5.5　CONTINUE 语句 …… (112)

§5.6 循环嵌套 …… (115)
§5.7 I/O 语句和 DATA 语句中的隐 DO 循环 …… (119)
§5.8 程序设计举例 …… (123)
练习五 …… (127)
第六章 函数和子例程 …… (130)
§6.1 语句函数 …… (130)
§6.2 外部函数 …… (134)
§6.3 内部函数 …… (139)
§6.4 子例程 …… (141)
§6.5 可调数组 …… (146)
* §6.6 内部语句和外部语句 …… (148)
* §6.7 子程序的多重入口和子例程的多路返回 …… (153)
* §6.8 SAVE 语句 …… (156)
练习六 …… (157)
第七章 字符处理 …… (161)
§7.1 字符型常数、变量、数组和子串 …… (161)
§7.2 字符数据的输入输出 …… (163)
§7.3 字符表达式和字符赋值语句 …… (165)
§7.4 字符串的比较和字符关系表达式 …… (168)
§7.5 有关字符运算的内部函数 …… (170)
§7.6 程序设计举例 …… (171)
练习七 …… (178)
第八章 文件系统和格式 I/O 补充 …… (181)
§8.1 文件系统概念 …… (181)
§8.2 READ/WRITE 语句的一般格式 …… (186)
§8.3 辅助输入/输出语句 …… (191)
§8.4 程序设计举例 …… (200)
* §8.5 格式 I/O 补充 …… (202)
练习八 …… (215)
第九章 公用、等价和数据块子程序 …… (218)
§9.1 COMMON(公用)语句 …… (218)
§9.2 EQUIVALENCE(等价)语句 …… (222)
§9.3 数据块子程序 …… (226)
练习九 …… (227)
* **第十章 程序结构和结构程序设计** …… (229)
§10.1 FORTRAN 源程序的模块式结构 …… (229)
§10.2 模块内部的基本程序结构 …… (231)
§10.3 结构程序设计简介 …… (233)
§10.4 改善程序可读性的若干措施 …… (237)

练习十…… (245)
附录一　信息编码表…… (247)
附录二　内部函数表…… (251)
附录三　FORTRAN 标准对照表 …… (254)
附录四　VAX FORTRAN 系统使用简介 …… (257)
附录五　IBM－FORTRAN 系统使用简介 …… (268)

第一章　电子计算机的一般知识

§1.1　电子计算机的一般特点

电子计算机通常指的是通用电子数字计算机，但习惯上人们有时也常直接称之为计算机。

顾名思义，电子计算机是一种可用于计算的电子装置。但是，这么说还不够确切，因为它尚不足以反映现代一般意义下的计算机的基本特点。现代通称的计算机，确切地说，它应该具有以下三个基本特征：

1. 计算机是一种具有高速运算能力的电子装置。

2. 计算机必顺具有记忆和逻辑判断能力。

3. 它还必须具有能够接收和自动执行指令的能力。

由于计算机具有能够接收、记忆并自动执行指令的能力。因此，人们想要让计算机做些什么，就可以用一条条指令的形式，把所要做的事情事先告诉计算机。然后计算机就可以准确无误地加以执行。

通常，我们把这种由人们事先准备好的，用来指挥计算机工作的指令系列，称为程序。并把存放在计算机中的程序和数据，统称为信息。

可见。计算机之所以区别于其它各种运算装置，并具有巨大威力，其原因就在于它能够记忆信息（即存贮程序和数据），而且能够在程序的控制下自动、高速、准确地执行运算。计算机能够存贮记忆的信息越多，执行指令的速度越快，它的威力就越大。

计算机是科学技术发展的产物，反过来，计算机的应用又推动着科学技术的不断进步。在当代社会里，计算机的应用程度，已成为社会科学技术发展水平的重要标志。在一些发达国家，计算机的应用几乎已进入一切领域。无论在科学计算、数据处理、自动控制，还是在辅助设计、辅助教学、人工智能等等方面，计算机都正在发挥着越来越重要的作用。像气象预报、火箭、人造卫星和航天飞机的发射等过去不可想象的事情，在计算机的帮助下，早已成为现实。相信计算机在全社会普及，并进入一般家庭的生活事务管理，也是举日可待的事情。

§1.2　电子计算机系统的组成

电子计算机是一个极其复杂的系统，其组成一般可分为硬件和软件两大部分。

硬件指计算机系统中的各种电子线路和电、磁、机械部件。也就是通常我们所看得见摸得着的那些具体的物理设备。

软件指在计算机系统中指挥机器工作的各种程序和数据表格。也就是存贮在计算机中的各种信息。

一、硬件（Hardware）

计算机系统的硬件部分通常由主机和外设两部分组成。

（一）主机(Main frame)

主机通常包括控制器(Control Unit)、运算器(Arithmetic—Logic Unit)和存贮器(Memory)三大部件。

1. 控制器：用于解释和执行指令，并指挥机器各部协调工作。

指令是计算机的操作命令。一般，每台计算机所能够执行的操作种类是有限的(少者几十条，多者几百条)。每条指令都有其特定的编码形式。这都是硬件线路设计时规定好了的。计算机能够执行的所有指令的集合，称为该计算机的指令系统。

2. 运算器：用于执行算术和逻辑运算。

运算器的运算速度很快，一般为每秒几十万次到几百万次，而一些大型或巨型机的运算速度，则甚至可高达几千万次到几亿次。

控制器和运算器合称中央处理机(CPU)，它一般由集成电路或大规模集成电路组成。

3. 存贮器：用于存放指令和数据。

存贮器一般由磁芯或半导体元件组成，通常它又称为内存或主存。

存贮器的存贮单位称为“单元”(或叫做“字”)，一个存贮单元一般可存放一个信息。为便于寻找，存贮器中所有存贮单元都 按一定的顺序编号(从 0 开始编号)。单元的编号称为地址。而存贮器中存贮单元的总数，就称为存贮器的容量。由于存贮器的容量一般比较大，因此，容量的计算常以 K 为单位($1K=2^{10}=1024$)，或者以 M(兆)为单位(1M＝1 千 K)。

指令和数据在存贮器中都是以二进制的编码形式存放的。每个存贮单元能够存放的二进位的位数称为存贮单元的字长。一般，计算机字的字长可以有 16 位、32 位或 64 位不等，而有些微型机则采用 8 位字长。按照国际常用的习惯，8 个二进位又称为一个字节。计算机的一个存贮单元(字)，随着计算机的不同，可以是由 1 个字节、2 个字节、4 个字节或 8 个字节组成。

存贮器存放信息的一般宏观特点是：当我们给某单元存入新数时，单元内原有的内容就被新数所取代；而从某单元取出数据时，则取出后该单元的原有内容不变

（二）外设(Peripheral device)

计算机的外部设备通常分为 输入/输出 设备(I/O device)和辅助存贮设备(Auxiliary memory)二大类。

输入/输出 设备用来输入程序和数据，并输出结果。常用的 输入/输出 设备有：纸带阅读/穿孔机、卡片阅读机、宽行打印机、电传打字机、CRT (Cathode Ray Tube)终端、绘图仪和模数转换仪等。

辅助存贮设备又称为外存，它是主机内存贮器的后援。外存的速度一般比内存慢，但外存价格低，容量大。因此，计算机中暂时不用的大量信息都保存在外存贮器中。常用的辅助存贮设备有磁盘和磁带等。

计算机的基本硬件结构见图 1－1。

二、软件(Software)

前面我们已经看到，计算机硬件是一套极其复杂的电子装置，它具有很强的功能和很高的运算速度。可是，要是没有软件的控制和管理，计算机的硬件再好也是没有用的，因为它根本无法使用，也无法工作。(没有配备软件的计算机称为裸机。)因此，为了充分发挥计

算机的潜在能力和提高工作效率，使计算机系统中的各个组成部分（统称为计算机系统资源）更加完备地协调工作，也为了便于人们使用，一般，计算机出售时，除硬件外，还必须配有相应的一套系统软件。

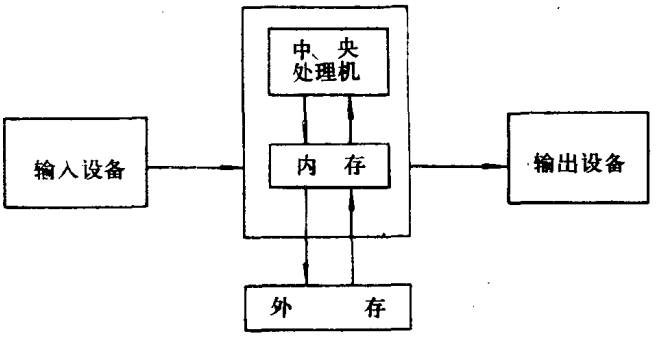


图 1－1　计算机基本硬件结构

系统软件一般包括操作系统、语言翻译系统和系统实用程序三大部分：

1. 操作系统（Operating system）：

这是一套用来管理计算机内部各部分软、硬件资源的程序，也就是整个计算机系统的总控制程序。计算机系统中的所有设备和其它程序都必须在它的控制下运行。

2. 语言翻译系统（Language translation system）：

这是一套用来把用户编写的语言程序翻译成计算机硬件能够理解并执行的一系列机器指令的程序。配置这套软件的目的，主要是为了便于用户使用程序设计语言来开发自己所需要的程序。

3. 系统实用程序（System utility programs）：

这是一些用于增强系统功能、完成某些特定系统操作（如系统信息的维护、管理，软件的查错、修改等）的系统程序。

关于系统软件的更深入知识，已超出本课程范围，这里就不予详叙了。

§1.3　算法和算法框图

一、算法

大家知道，做每一样事情都有每样事情的具体工作步骤和方法。这种具体的工作步骤或解题方法，用一个特定的术语来说，就是所谓的算法。也就是说，算法就是为解决某一特定问题而采用的具体工作步骤或方法。人做事需要“算法”，计算机也不例外。实际上，计算机算题的时候，它的每一步动作都需要由人们事先编好的程序来控制。因此，人们需要计算机计算什么问题，就必须根据计算机的特点，事先设计好计算机的算法，然后再使用某种程序设计语言，把该计算问题的算法步骤编写成程序，交给计算机去执行。计算机能否有效地工作，在很大程度上决定于人们是否能为计算机设计一个好的算法。

那么，什么样的算法才是一个好的算法呢？目前人们普遍认为，一个好的算法除满足正确性要求之外，还应当具备结构性好和效率高两个特点。所谓结构性好就是算法要结构

清晰、步骤简单、容易阅读理解和容易在计算机上实现。而效率高则包括要尽可能少花计算机的运算时间和存贮空间两个方面。

例 1 设计在计算机上计算 $s=\sum_{i=1}^{100}a_i$ 的算法。

算法一：在内存中设一累加用的存贮单元 S。

第 1 步：将存贮单元 S 清零。
第 2 步：输入第一个数 a_1
第 3 步：把 a_1 加到存贮单元 S 中。
第 4 步：输入第二个数 a_2。
第 5 步：把 a_2 加到存贮单元 S 中。
⋮ ⋮
第 200 步：输入第 100 个数 a_{100}。
第 201 步：把 a_{100} 加到存贮单元 S 中。
第 202 步：输出存贮单元 S 中的结果。

算法二：在内存中设一累加单元 S 和一计数单元 I。
第 1 步：将累加单元 S 清零。
第 2 步：将计数单元 I 清零。
第 3 步：输入一个数 A。
第 4 步：把 A 加到累加单元 S 中。
第 5 步：计数单元 I 的值增加 1。
第 6 步：若 I=100 则执行下一步，否则转回第 3 步。
第 7 步：输出结果 S。

上述两种算法显然都能够正确实现所要求的计算。但是，相比之下不难看出，第二种算法要比第一种算法好，因为它更为简单明确，而且也更为实用。一旦问题要求改为计算 1000 个数的和，按第一种算法就必须写出 2002 步，而按第二种算法，则只须把控制条件 I=100 改为 I=1000 即可。

二、算法框图

上面我们采用了文字描述的方法表示算法。实际上，算法还可以有其它多种表示方法。如用程序设计语言表示，用伪码（一种介于自然语言和程序设计语言之间的伪代码语言）表示和用图示法表示等等。用自然语言表示适合人们的习惯用法，但书写起来不方便。用程序设计语言表示算法是程序设计的最终目的，但由于程序设计语言是一种形式化语言，比较抽象难懂，且细节又过于严格，在算法设计阶段对比较复杂的问题一般难于直接使用。因此，作为算法的表达工具，在算法设计阶段，人们一般多采用图示法或伪码方法。而图示法由于具有简单和直观的优点，初学者用它作为程序设计的辅助和过渡工具，更具有独特的好处。因此，下面我们只介绍算法的图示方法。

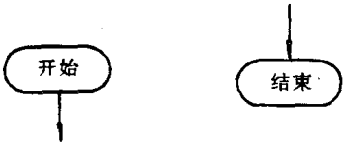
算法的图形表示又称为算法框图。有两种常用的算法框图。

1. 算法流程图

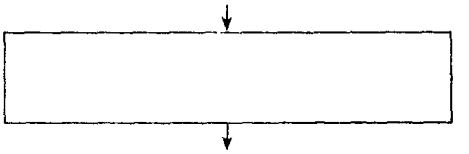
算法流程图通常采用以下具有特定含义的图框和流线来表示算法：

(1) 箭头(流线)：表示流程路线。

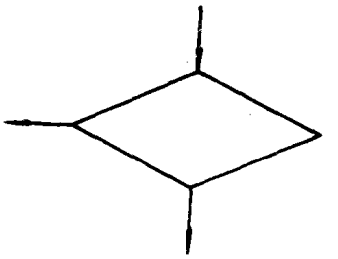
(2) 椭园框(端框)：表示流程的起点和终点。



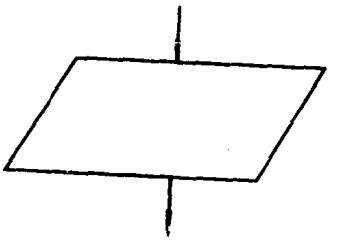
(3) 方框(处理框)：表示运算处理。



(4) 尖框(判断框)：表示逻辑判断。根据逻辑判断结果选择一种执行路径。



(5) 斜框(输入输出框)：表示输入输出。



例如，前面求 100 个数之和的算法二，可用流程图表示如下(图 1－2)：

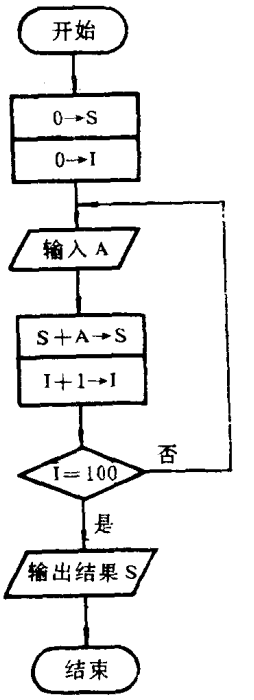


图 1－2

流程图能形象地表示算法，但结构不紧凑，处理复杂问题时容易造成流程上的混乱。因此目前多倾向于采用下面的结构框图方法。

2. 结构框图

结构框图规定使用三种基本结构图作为算法设计的基本单元。三种基本结构是：

(1) 顺序结构(Secquence structure)

顺序结构通常是一系列顺序执行的运算和处理。用前面的流程图表示，就是一系列顺序执行的处理框(图 1－3)。顺序结构的结构框图用图 1－4 表示。

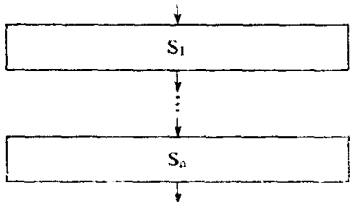


图 1－3 顺序结构流程图

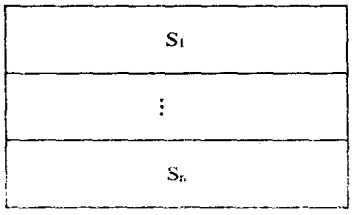


图 1－4 顺序结构的结构框图

(2) 选择结构(Selection structure)

选择结构又称为判断结构(Decision structure)，这种结构通常是根据一个逻辑条件的成立与否，来选择下一步应该执行哪一种处理。选择结构的流程图如图 1－5。其结构框图表示为图 1－6。

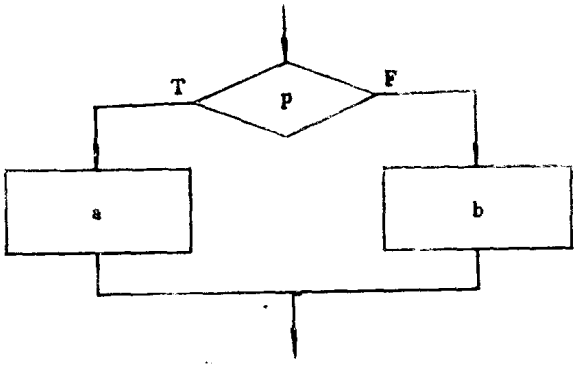


图 1－5 选择结构流程图

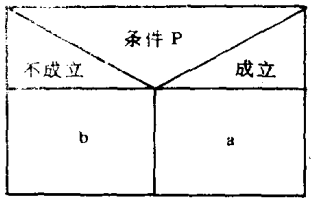


图 1－6 选择结构的结构框图

上面框图表示：若条件 P 成立，则执行处理框 a，否则执行处理框 b。

(3) 循环结构(Loop structure)

循环结构又称为重复结构(Repetition structure)，它根据条件 P 的成立与否来决定是否重复执行某处理框 S。这种结构通常有"先判后做"和"先做后判"两种结构形式。

"先判后做"循环结构表示：当条件 P 成立时，重复执行处理框 S，否则执行下一结构。这类循环称为当型(WHILE)循环，其流程图表示为图 1－7，结构框图表示为图 1－8。

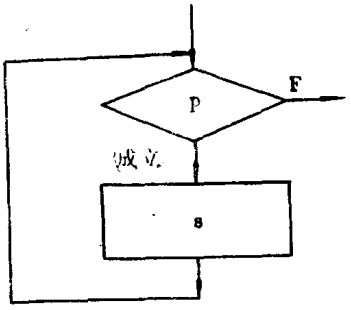


图 1－7　当型循环流程图

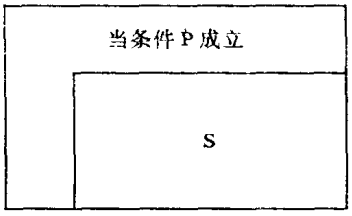


图 1－8　当型循环结构框图

“先做后判”循环结构表示：重复执行处理框 S，直到条件 P 成立时再执行下一结构。这类循环称为直到型(UNTIL)循环。其流程图表示为图 1－9。结构框图表示为图 1－10。

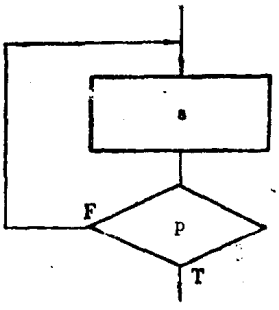


图 1－9　直到型循环流程图

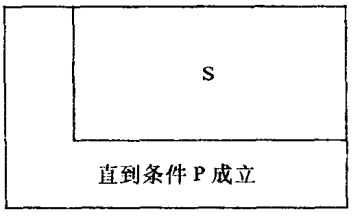


图 1－10　直到型循环结构框图

利用上述框图符号，例 1 求 100 个数之和的算法二可分别用流程图(图 1－2)和结构框图(图 1－11)表示如下：

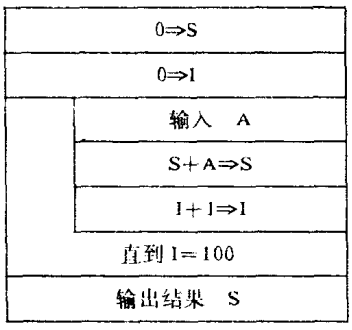
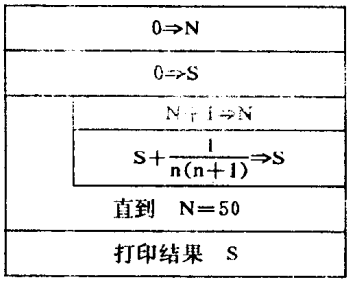


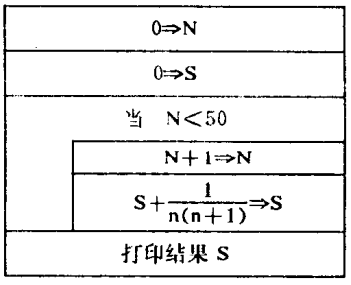
图 1－11　例 1 结构框图

例2　已知 n=50，计算 $s=\frac{1}{1\times2}+\frac{1}{2\times3}+\frac{1}{3\times4}+\cdots+\frac{1}{n\times(n+1)}$，试用结构框图表示其算法。

算法一：



算法二：



结构框图是 1973 年由美国 I. Nassi 和 B. Schneiderman 提出的，故又称为 N－S 框图。

§1.4　程序设计和程序设计语言

前面我们已经知道，计算机算题的时候，它的每一步动作都需要由人们事先编好的程序来控制。因此，人们需要计算机计算什么问题，就必须使用计算机系统所允许的某种特定语言，事先把该计算问题的算法步骤编写成程序，然后再交给计算机去执行。为计算机编写程序的过程，就叫做程序设计。而在程序设计过程中，用于编写程序的语言，就称为程序设计语言。

一、程序设计技术与程序设计语言的发展概况

1. 原始阶段，使用机器语言

在计算机发展的初期（四十年代中～五十年代初），人们直接使用裸机，无软件概念。因此，编写程序的工作也只能直接根据计算机的指令系统（称为机器语言）来编写。这种由人工用机器指令编写的程序称为手编程序。由于机器指令都是一堆编码数字，就象电报编码一样，不经翻译人们是很难看懂的。因此，手编程序既费脑力，又容易出错，而且也不便于阅读、检查、交流。不仅如此，按这种工作方式，往往人们辛勤劳动花费几周甚至几个月的时间编写的程序，电子计算机只要几分钟就计算完了。可见其工作效率是极端不相适应的。

2. 低极语言阶段，使用汇编语言

从五十年代初期开始，为了提高计算机的使用效率，人们就一直在想办法寻找实现程序设计自动化的途径。最先提出的办法，就是采用助忆码和符号地址来代替机器语言中的二进制指令代码和二进制地址，然后再通过一个叫做“汇编程序（Assembler）”的简单翻译程序，来实现代码指令和地址的自动代真。这种采用助忆码和符号地址的语言，就称为汇编语言。用汇编语言编写程序，步骤仍和手编程序差不多，所不同的只是改用了比较便于记忆的符号来代替不便记忆的数字代码。因此汇编语言仍属于低级的程序设计语言。由于不同的计算机一般都具有不同的指令系统，所以不同计算机的机器语言和汇编语言也都各不相同。机器语言和汇编语言都属于面向机器的语言。

3. 高级语言阶段，广泛使用各种高级语言

从五十年代中期开始，随着计算机语言翻译技术的不断发展和完善，程序设计技术也

开始进入了高级语言的阶段。高级语言是一种比较接近于自然语言(主要是英语)和数学语言的语言。在语言特点上,它已经远远地摆脱了计算机具体指令系统的约束。因此,它一般可通用于不同的计算机系统,属于面向过程的语言。高级语言的出现,大大推动了计算机应用的迅速普及和推广,从而也促进了软件技术的迅猛发展。廿多年来,适用于各个不同领域的各种程序设计语言的不断涌现,犹如雨后春笋,数量已达数百种之多。在科学与工程计算方面,使用比较广泛的有 FORTRAN 语言、BASIC 语言、ALGOL 语言、PASCAL 语言和 PL/1 语言等。

但是,不管语言种类如何繁多,其组成原理和实现方法却大致都是一样的。一般地说,语言是一种用于通讯的符号系统。而程序设计语言则是一种用于描述算法,在人与计算机或人与人之间进行算法思想通讯的符号系统。因此,它也和其它语言一样:

(1) 它必须有一套规定好的基本符号。

(2) 必须有一套由上述基本符号构成各类语句和程序的基本语法规则。

(3) 对语言中的每一种合法语句或句子成分,都必须有一个确切的语言含义(即表达一组确定的操作命令)。

从而保证该语言能确切无误地表达信息,完成人与机器或人与人之间的信息通讯任务。

二、源程序在计算机上的执行过程

为区别于机器语言的手编程序,通常把用汇编语言或高级语言等编写的程序称为"源程序"。源程序在计算机上执行时,一般都要首先经过一个相应的语言翻译系统的加工处理,然后才能执行。

语言翻译系统加工处理源程序的方式通常有编译方式和解释方式两种。

1. 编译方式

按编译方式工作时,源程序在计算机上执行一般需要经过"编译"、"连接"和"运行"三个步骤:

(1) 源程序编译

这步工作是由计算机上一个叫做"编译程序(Compiler)"的特定语言翻译程序来完成的,它所完成的工作,就是把源程序翻译成为一个功能上等价的目标语言程序(一般地说,它也就是实际计算机的机器语言程序),称为"目的程序"或"目的模块"。

(2) 目的模块连接

一般,一个程序可以由几个目的模块组成。这些目的模块可以是一次编译产生的,也可以是分几次编译产生的。另外,目的模块通常也需要和软件系统库中的一些系统模块组合在一起,才能形成真正可在计算机上执行的程序。而这些工作都是由计算机上一个叫做"连接程序(Linker)"的程序来完成的。目的模块经连接程序连接装配以后所形成的程序,就叫做"可执行程序"。

(3) 程序运行

这步工作才是由计算机执行真正用户所需要的计算程序(即源程序编译、连接后产生的可执行程序)。它输入必要的初始数据,并按照用户所规定的算法步骤,计算出所需的结果。

编译方式的主要优点是它能够产生具有较高执行效率的目标程序,而且一旦调试成