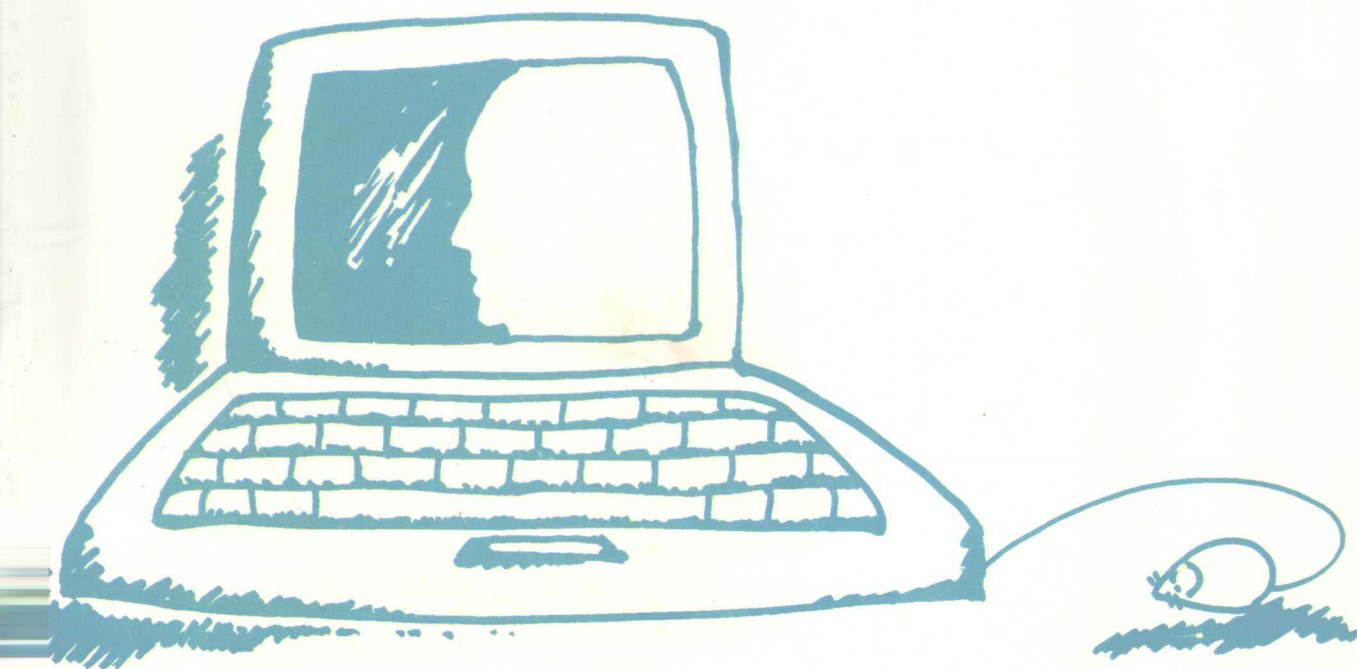


四川省计算机基础教育研究会推荐系列辅导教材

FORTRAN程序设计

自学指导与应试指南

王 舒 黄地龙 编著



电子科技大学出版社

中国工程计算机学会工程应用软件开发专业委员会

FORTRAN程序设计

自学指导与应试指南

王 明 李 强 编



机械工业出版社

四川省计算机基础教育研究会推荐系列辅导教材

FORTRAN 程序设计

自学指导与应试指南

向孟光

主 审

王 舒 黄地龙

编 著

内 容 提 要

本书是为读者准备参加计算机等级考试（二级 FORTRAN）而编写的指导书，也是学习 FORTRAN 程序设计及参加各类 FORTRAN 考试的辅导教材。本书共分三部分，第一部分简要叙述了考试大纲内容，对其中易混淆的问题做了分析和对比，为读者复习或学习 FORTRAN77 程序设计提供了内容提要。第二部分为各类试题集粹，其中包括试题类型分析和重点难点提示。收集的试题具有代表性，并按试题类型及知识结构的难易程度整理分类。全部试题均附有参考答案及解答说明。第三部分针对程序设计中常见错误作了重点讲解和分析。全书内容解析透彻、简练。

声 明

本书无四川省版权防盗标识，不得销售；版权所有，违者必究，举报有奖。举报电话：
(028) 6636481 6241146 3201496

FORTRAN 程序设计 自学指导与应试指南

向孟光 主审

王 舒 黄地龙 编著

出 版：电子科技大学出版社出版（成都建设北路二段四号，邮编：610054）

责任编辑：周 军

发 行：新华书店

印 刷：西南冶金地质印刷厂印刷

开 本：787×1092 1/16 印张 12 字数 285 千字

版 次：2000 年 3 月第一版

印 次：2000 年 3 月第一次印刷

书 号：ISBN 7—81065—358—X/TP·231

印 数：1—4000 册

定 价：13.80 元

序

随着科学技术的飞速发展,计算机已成为各行各业不可缺少的应用工具,计算机知识和应用能力已成为当代大学生培养质量高低的一个重要组成部分,是面向 21 世纪的高等教育培养高质量人才最突出需要加强的教学环节之一。目前在全国高校中普遍开展的计算机知识和应用能力等级考试制度正有效地推动这一目标的实现,四川省已成功地举行了十一次这样的考试正充分的证明了这一点,而已成立六年多的四川省计算机基础教育研究会为保证我省计算机知识和应用能力等级考试制度和方法的不断完善提供了有力的保证。

到目前为止,有关计算机应用等级考试的图书不少,对推动我省计算机知识和应用能力的普及起了积极的作用,但由于他们中的大多数是个别作者独立写作,难以反映众多学生对教材的需求。这一套计算机等级考试二级系列通用教材及配套复习资料和实验指导书一共十五种的主编和执笔者是来自省内许多大专院校,他们均是长期工作在该领域教学第一线,有丰富教育经验的优秀教师,他们中大多数是四川省计算机基础教育研究会成员单位的代表,他们的参与为本套教材的质量和水平提供了可行的保证,也使本套教材有十分广泛的代表性和符合四川省等级考试大纲的要求。电子科技大学出版社为这套系列教材的成功编写作了大量的组织工作。

综上所述,本系列教材可作为我省高校非计算机专业计算机等级考试(二级)的教材或参考书,我相信,这套系列教材的出版将进一步推动我省高校计算机基础教育质量的提高,推动计算机应用的进一步普及,对我国全民族现代化素质的进一步提高有所裨益。

四川省计算机等级考试委员会副主任

兰家隆 教授

2000 年元月 18 日于成都

前 言

人类已迈入 21 世纪, 科学技术突飞猛进, 知识经济蓬勃兴起。计算机在经济发展中的地位日益重要, 通晓计算机知识已经成为现代人必备的基本技能之一。计算机等级考试的设立旨在为衡量应试者计算机掌握程度提供一个统一、客观的测评标准。为帮助读者在短期内了解考试内容及要求, 熟悉考试形式, 顺利通过考试, 我们精心编写了这本计算机等级考试二级 FORTRAN 考试指南, 以满足读者的需要, 同时它也可以作为其他各种 FORTRAN 考试的复习材料。

本书分为三个部分。第一部分为 FORTRAN 考试大纲, 其中简述了大纲要求的全部内容, 并对其中易混淆的问题进行了对比。它不仅可以作为读者的复习提纲, 也是 FORTRAN 程序设计主要内容的浓缩, 为读者快速掌握 FORTRAN 程序设计提供方便。第二部分集粹了 FORTRAN 考试的各种题型和常用试题, 并按试题类型及知识结构的难易程度整理分类, 在每类试题前还给出了重点难点提示。所收集的试题均具有代表性, 并综合利用了程序设计的常见算法。因此, 本书也是学习 FORTRAN 程序设计的一本内容全面的习题集。为便于读者掌握做题能力并启发思路, 全部试题均给出了一个或多个参考答案, 并作了简要说明。第三部分收集了 FORTRAN 程序设计中的常见错误, 予以扼要分析, 指出错误原因, 归类提供纠错方法, 以提醒读者重视, 并通过正误对比使读者迅速发现问题症结所在, 以帮助读者正确掌握 FORTRAN 程序设计。

本书作者长期从事计算机教育, 具有丰富的 FORTRAN 教学经验。本书第一章和第二章由四川大学王舒编写; 第三部分由成都理工学院黄地龙编写, 四川大学向孟光主审。由于作者水平所限, 难免有不尽人意之处, 敬请读者批评指正。

编 者

2000 年 1 月

目 录

第一章 考试大纲及简述(FORTRAN 77 标准文本)	1
1.1 基本概念	1
1.2 基本语句与顺序结构程序设计	5
1.3 选择结构与循环结构程序设计	7
1.4 数组	9
1.5 字符信息处理	10
1.6 函数与子程序	11
1.7 文件	13
1.8 常用算法	15
第二章 试题集粹及分析	25
2.1 试题介绍	25
2.2 语法和概念题	26
2.3 阅读程序题	35
2.3.1 读程序并回答问题	35
2.3.2 程序填空	59
2.4 编程题	78
2.4.1 三种基本结构	78
2.4.2 数组	84
2.4.3 字符	91
2.4.4 函数与子程序	94
2.5 上机题	96
2.5.1 三种基本结构	96
2.5.2 数组	99
第三章 程序设计常见错误及分析	116
3.1 程序设计常见语法错误	116
3.1.1 FORTRAN 程序的书写格式与数据类型转换	116
3.1.2 选择结构程序	119
3.1.3 循环结构程序	123
3.1.4 数组应用中的常见错误	127
3.1.5 数据的输入/输出	130
3.1.6 输入/输出语句与文件的使用	134
3.1.7 函数与子程序	138
3.1.8 字符信息处理	150

3.2 程序设计中的逻辑错误	154
附录一 四川省普通高校非计算机专业计算机等级考试 FORTRAN 语言程序设计考试大纲 (FORTRAN 77 标准文本)	176
附录二 二级(FORTRAN)笔试试卷	178

第一章 考试大纲及简述

(FORTRAN77 标准文本)

本部分给出 1998 年四川省普通高校非计算机专业计算机等级考试考试大纲，并对大纲逐条进行了阐述，列出了大纲所有要求语句的一般形式和常见程序模式，对一些重点、难点和易混淆的问题进行了说明及比较。对大纲要求的算法，全部做了讲解，并予实例说明。本部分旨在让读者快速了解考试要求，明确学习 FORTRAN77 程序设计应该掌握的基本要点。为便于读者阅读和理解，原大纲内容全部用黑体字给出。

1.1 基本概念

一、源程序的结构和书写规则

FORTRAN 源程序的结构可简单归结为：程序分块、块中分行、行中分区。即一个源程序由一个或若干个程序块组成，这些程序块可以为主程序、函数子程序、子例行程序及数据块子程序。每个程序块都可为一个独立的编译单位。源程序中有且仅有一个主程序，它可调用各子程序，但不能被子程序所调用。因此，源程序总是从主程序开始执行。

程序块由注释行和语句行组成。注释行仅仅是为了人们阅读程序方便而加到程序中的注释，它不产生任何机器操作，但在打印程序清单时原样打印出来。在源程序中，注释行以“c”或“*”作为该行的第一个字符。语句行才是 FORTRAN 程序的实体。它可分为执行语句和非执行语句。执行语句使计算机在运行时产生某些操作，目标程序中对对应若干条机器指令。非执行语句将有关信息通知编译系统，以便在编译时作出相应处理，为执行语句的执行作准备。

语句行必须严格地按照一定的格式书写，其规定为：

1. 第 1~5 列为语句标号区。程序中只有要被别的语句引用的语句才需要标号。标号为无符号整数，其大小不代表程序执行的顺序。在标号区中的空格不起作用。
2. 第 6 列为续行区。FORTRAN 规定，一行只能写一个语句，如果一个语句太长，一行内写不完，可以续行。但必须在续行的第 6 列写一个非空格或非零的符号，作为续行标志。一个语句最多允许 19 个继续行。
3. 第 7~72 列为语句区。语句可从区间内的任何位置开始书写。
4. 第 73~80 列为注释区。注释区的作用与注释行相同。

二、常量、变量及其类型说明

FORTRAN 中的数据具有以下六种类型：

整型

实型
复型
双精度型
逻辑型
字符型

这些数据在程序中以常量或变量的形式出现。所谓常量是指在程序执行过程中其值始终不变的量。不同类型常量的表示形式为：

整型：±<整数>

实型：小数点形式：±<整数>.<整数>

指数形式： $\left\{ \begin{array}{l} \text{<整数>} \\ \text{<小数点形式实型数>} \end{array} \right\} E \pm \text{<整数>}$

双精度型：将实型数的小数点形式中的 E 改为 D 即可

复型：(<实型数>, <实型数>)

其中，前一个<实型数>表示复数的实部，后一个<实型数>表示复数的虚部。

逻辑型：.TRUE. 表示逻辑“真”

.FALSE. 表示逻辑“假”

字符型：用单引号(即撇号')括起来的若干个字符

变量是用来存放常量的，其值在程序执行过程中可变，程序中用变量名来识别。变量名的取名规则为：

1. 第一个字符必须是字母；
2. 在第一个字符后跟 0~5 个字母或数字。

变量的类型可以用下面三种方法加以识别：

1. 隐含规则(I-N 规则)：程序中的变量名，凡以字母 I, J, K, L, M, N 开头的，即为整型变量；以其他字母开头的为实型变量。

2. 类型说明语句：若程序中的变量所需的类型与隐含规则不符，可用类型说明语句加以指定，以改变隐含规则对变量类型的约束。类型说明语句的一般形式为：

<类型> <变量名>

其中，<类型>可为：

INTEGER	表示整型
REAL	表示实型
COMPLEX	表示复型
DOUBLE PRECISION	表示双精度型
LOGICAL	表示逻辑型
CHARACTER	表示字符型

3. 隐含说明语句 IMPLICIT：该语句将某一字母开头的变量指定为所需的类型。隐含说明语句的一般形式为：

IMPLICIT <类型>(<字母表>)[,]

其中，<类型>与类型说明语句相同；<字母表>中各字母之间用逗号分隔，也可出现

<字母 1>-<字母 2>

表示按字母顺序从字母 1 到字母 2 的所有字母。

确定变量类型的三种方法的优先级从高到低为：

类型说明语句——隐含说明语句——隐含规则

类型说明语句和隐含说明语句都是不可执行语句，应出现在本程序单元中所有可执行语句的前面，而隐含说明语句又应该在所有类型说明语句的前面。

三、各类表达式的求值规则及运算特点(算术表达式、关系表达式、逻辑表达式、字符表达式)

1. 算术表达式

FORTRAN 规定的算术运算符为：

＋ 表示“加”(或正号)

－ 表示“减”(或负号)

* 表示“乘”

/ 表示“除”

** 表示“乘方”

不同运算符运算的优先级从高到低为：

**——*, /——+, -

同一优先级的两个运算，按“先左后右”的原则运算，只有乘方运算按“先右后左”的顺序进行。例如，表达式 $2**3**2$ 的值是 512，而非 64。

算术表达式运算中结果类型的规定为：

(1)同类型的算术量之间的运算结果仍保持原类型；

(2)若参加运算的两个算术量为不同类型，则编译系统自动将其转换成同一类型后再运算。转换规律为：将低级类型转换成高级类型。类型的级别从高到低为：

复型——双精度型——实型——整型

2. 关系表达式

关系表达式的一般形式：

<算术表达式> <关系运算符> <算术表达式>

关系运算符为：

.LT. 表示“<”

.LE. 表示“<=”

.EQ. 表示“=”

.GT. 表示“>”

.GE. 表示“>=”

.NE. 表示“<>”

关系表达式的运算结果为逻辑真(.TRUE.)或逻辑假(.FALSE.)。

3. 逻辑表达式

基本逻辑表达式的一般形式为：

<关系表达式> <逻辑运算符> <关系表达式>

逻辑运算符为：

.AND. 表示“逻辑与”

.OR. 表示“逻辑或”

.NOT. 表示“逻辑非”

逻辑表达式的运算顺序为:

(1)先计算算术表达式的值;

(2)再计算关系表达式的值;

(3)最后进行逻辑运算。逻辑运算的次序为: 先.NOT., .AND.次之, 最后.OR.。如果有括号, 则先进行括号内的运算。

4. 字符表达式

字符表达式放在字符信息处理部分叙述。

四、结构化程序与程序的三种基本结构

结构化程序的三种基本结构为: 顺序结构、选择结构及循环结构。每一基本结构具有以下共同特点:

1. 只有一个入口;
2. 只有一个出口;
3. 结构内的每一条语句都有机会被执行到;
4. 结构内不存在无限循环。

五、计算机算法概念与算法的表示(流程图)

计算机算法即是利用计算机解题的方法和步骤.它具有如下特点:

1. 算法的操作步骤是有限的;
2. 其中每一个步骤都应当是确定的;
3. 每一个步骤都应当能有效的执行, 并得到确定的结果;
4. 有零个或多个输入;
5. 有一个或多个输出。

流程图是用以表示算法的一种框图, 它由许多不同形状的符号(框)组成。图 1 列出的各种符号是美国国家标准化协会 ANSI 规定的一些常用符号。

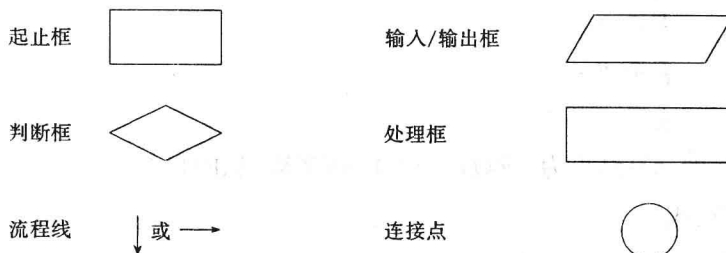


图 1 流程图的常用符号

用流程图表示的结构化程序设计的三种基本结构如图 2、图 3、图 4 所示.图中 A、B 可

为空块，也可为三种基本结构中的任何一种。

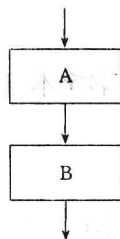
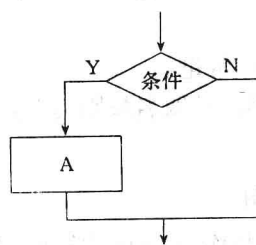
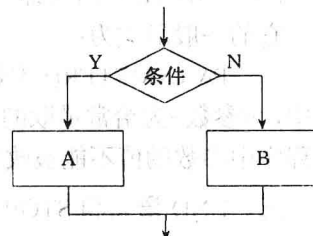


图 2 顺序结构

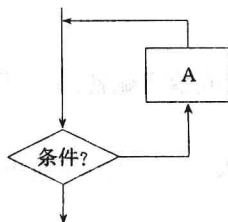


单边选择

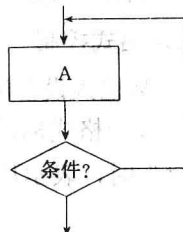


双边选择

图 3 选择结构



当型循环



直到型循环

图 4 循环结构

1.2 基本语句与顺序结构程序设计

一、赋值语句

赋值语句的作用，就是把表达式的值计算出来，送到指定变量的内存单元中。其一般形式为：

$\langle \text{变量} \rangle = \langle \text{表达式} \rangle$

当算术表达式运算结果的类型与左边变量的类型不一致时，计算机将把结果类型转变成变量类型后再赋值。

二、DATA 语句和 PARAMETER 语句

DATA 语句即赋初值语句，通过该语句可给变量和数组赋初值。其一般形式为：

DATA $\langle \text{变元表} \rangle / \langle \text{常量表} \rangle /$

其中， $\langle \text{变元表} \rangle$ 中可包括简单变量，下标变量，数组名及隐 DO 表。常量表中可出现： $\langle \text{数字 1} \rangle * \langle \text{数字 2} \rangle$ ，表示有数字 1 个数字 2。如：

DATA M, N/2*0/

表示将变量 M, N 的初值都赋为 0。

PARAMETER 语句即参数语句, 其作用是为本程序单元定义一个符号, 来表示某个常量。它的一般形式为:

PARAMETER(<参数>=<常量>, ...)

其中, <参数>为给常量取的名字, 其取名规则与变量名相同, 名字本身也遵循 I-N 规则。在程序中参数的值不能被改变。

三、END 语句和 STOP 语句

END 语句的作用有二: 一是作为一个程序单元的结束标志; 二是结束本程序单元的运行。在主程序中, END 语句使程序停止运行, 控制回到操作命令状态。在子程序中使流程返回调用程序, 接着执行调用程序中的其他语句。END 语句的一般形式为:

END

STOP 语句表示停止程序的执行, 一个程序单元中可以有多个。其一般形式为:

STOP [n]

其中, n 为一个不超过 5 位的无符号整数或一个字符串。END 语句之前的 STOP 语句可以不写。

四、输入/输出语句及格式控制

输入语句的作用, 是从输入设备读取数据, 为变量或数组赋初值。其一般形式为:

READ(<通道号>, <格式标识>)<输入表>

其中, 当<通道号>为“*”时, 表示从键盘输入; <输入表>中可出现简单变量、下标变量、数组名及隐 DO 表。

输出语句的作用, 是将运算或处理的结果通过输出设备显示(或打印)出来。其一般形式为:

WRITE(<通道号>, <格式标识>)<输出表>

其中, 当<通道号>为“*”时, 表示输出显示在屏幕上; <输出表>中可出现表达式, 这时计算机先运算出表达式的值, 再将结果值输出。

1. 表控格式输入/输出语句

表控格式系指某些约定的格式。当采用表控格式输入/输出时, 输入/输出语句中的<格式标识>应为“*”, 即:

READ(<通道号>, *)<输入表>

WRITE(<通道号>, *)<输出表>

表控格式输入语句要求用户在输入数据时, 各数据之间用逗号或空格分隔, 且输入数据的个数, 类型应与 READ 语句的输入表中变元的个数、类型一致。如果输入的数据个数少于变元的个数, 则系统将等待用户继续输入, 直到输入完全部数据为止; 如果在一行上输入的数据个数多余变元的个数, 则后面多余的数据不起作用。当输入数据的类型与变元不一致时, 一般按出错处理。但如果输入数据为整型, 而变元为实型, 则许多系统先自动将整型数据转换成实型数据后, 再输入给实型变元。

2. FORMAT 格式输入/输出语句

格式语句的作用是为格式输入/输出语句提供指定的格式, 当采用 FORMAT 格式输入/输出时, 输入/输出语句中的<格式标识>应为 FORMAT 语句的语句标号。

FORMAT 语句的一般形式为:

FORMAT(<格式说明>)

其中, <格式说明>是通过格式描述符来表示的。各种格式描述符的一般形式及作用如下:

Iw	说明整型数据的格式
Fw.d	说明小数点形式实型数据的格式
Ew.d	说明指数形式实型数据的格式
Lw	说明逻辑型数据的格式
Aw 或 A	说明字符型数据的格式
Dw.d	说明双精度型数据的格式

其中, w 表示字段宽度; d 表示小数点后的位数。

格式语句前必须有语句标号, 以供格式输入/输出语句引用。同一格式语句可被多条输入/输出语句引用。

输出语句所引用的格式语句的格式说明一般以 1X(即一个空格)开头, 表示先走纸一行, 然后开始该行的输出。

1.3 选择结构与循环结构程序设计

一、有关控制语句(无条件 GOTO 语句、逻辑 IF 语句、块 IF 语句、DO 语句)

无条件 GOTO 语句的一般形式为:

GOTO <语句号>

逻辑 IF 语句的一般形式为:

IF(<逻辑表达式>)<可执行语句>

其中, <可执行语句>可以是除 DO 语句、块 IF 语句、或另一逻辑 IF 语句以外的任何一条可执行语句。

块 IF 语句的一般形式为:

```
IF(<逻辑表达式>)THEN
ELSE IF(<逻辑表达式>)THEN
ELSE
END IF
```

DO 语句的一般形式为:

DO <语句号> <循环变量>=<初值>, <终值>, <步长值>

其中, <循环变量>可以是整型或实型变量; <初值>、<终值>和<步长值>可以是整型或实型表达式。<步长值>为 1 时, 可省略不写。

二、选择结构的组成与分支程序设计

选择结构的程序一般采用块 IF 语句来实现, 其形式为:

```
IF(<条件 1>)THEN
:
:
ELSE IF(<条件 2>)THEN
```

```

:
:
ELSE
:
:
END IF

```

在一个块 IF 中，IF 语句和 END IF 语句必须配对出现；ELSE IF 语句可视情况不用或用一条，甚至多条；ELSE 语句也可视情况不用或用一条。

如果在选择结构中 ELSE IF 块和 ELSE 块为空块，而 THEN 块只有一个语句，则可用逻辑 IF 代替块 IF。

三、循环结构的组成与循环程序设计(DO 循环，当型循环，直到型循环)

1. DO 循环的典型结构

DO <语句号> <循环变量>=<初值>, <终值>, <步长值>

```

:
: } (循环体)
:
:

```

<语句号> CONTINUE

2. 用块 IF 语句和 GOTO 语句实现当型循环

<语句号> IF(<条件>)THEN

```

:
: } (循环体)
:
:

```

GOTO <语句号>

END IF

3. 用逻辑 IF 和 GOTO 语句实现直到型循环

<语句号> (循环体的第一个语句)

```

:
: } (循环体其他语句)
:
:

```

IF (<条件>) GOTO <语句号>

在进行循环结构程序设计时，如果循环次数已知或可通过初值、终值、步长值换算出来，则一般采用 DO 循环来实现；当循环次数无法确定时，采用当型循环或直到型循环来实现。通常情况下，当型循环和直到型循环是可以相互转换的。但是，如果执行程序时，这两种循环的条件第一次就不成立，那么当型循环的循环体就没有被执行，而直到型循环的循环体却已经被执行了一次。

四、选择结构及循环结构的嵌套

选择结构和循环结构都可进行嵌套或互相嵌套，但被嵌套部分都必须完整地包括在某

一基本块中。

1.4 数组

一、数组概念及数组的说明

一组具有相同的数据类型并按一定顺序排列的数据称为数组。程序中使用数组时，必须先进行说明。说明数组的方式有如下几种：

1. 用数组说明语句

其一般形式为：

DIMENSION <数组名>(<下界>: <上界>,)[,]

数组为几维就需要几组上下界。当<下界>为1时，语句中“<下界>:”可省略不写。

2. 用类型说明语句

其一般形式为：

<类型> <数组名>(<下界>: <上界>,)[,]

3. 用数组说明语句与类型说明语句配合使用

当采用数组说明语句说明数组时，如果数组的类型与隐含规则不符，可采用该方式。

其一般形式为：

DIMENSION <数组名>(<下界>: <上界>,)[,]

<类型> <数组名>

二、数组元素在内存中的存储结构

数组元素在内存中是按第一个下标变化最快，其余下标变化趋势依次递减的顺序存放的。因此，对二维数组即按列存放。

三、数组和数组元素的输入/输出(包括隐 DO 表的输入/输出)

对数组和数组元素的输入/输出，可直接在输入/输出语句的输入/输出表中给出数组名，数组元素或隐 DO 表即可。当输入/输出表中有数组名时，表示按数组元素在内存中的存储顺序输入/输出整个数组。隐 DO 表则可用于按所需的顺序输入/输出数组的部分或全部元素。

隐 DO 表是用循环的方式来表示变元。其一般形式为：

(<变元表>, <循环变量>=<初值>, <终值>, <步长值>)

其中，<变元表>可包括简单变量，下标变量，甚至另一隐 DO 表；<循环变量>、<初值>、<终值>、<步长值>应为整型。实际上与 WRITE 语句连用时，隐 DO 表的变元表中也可包括常量。

由于二维数组在内存是按列存放的，如果需要按行的顺序输入/输出，则可在输入/输出表中采用隐 DO 表，例如具有 10 行 10 列的数组 A 的隐 DO 表的形式为：

((A(I, J), J=1, 10), I=1, 10)