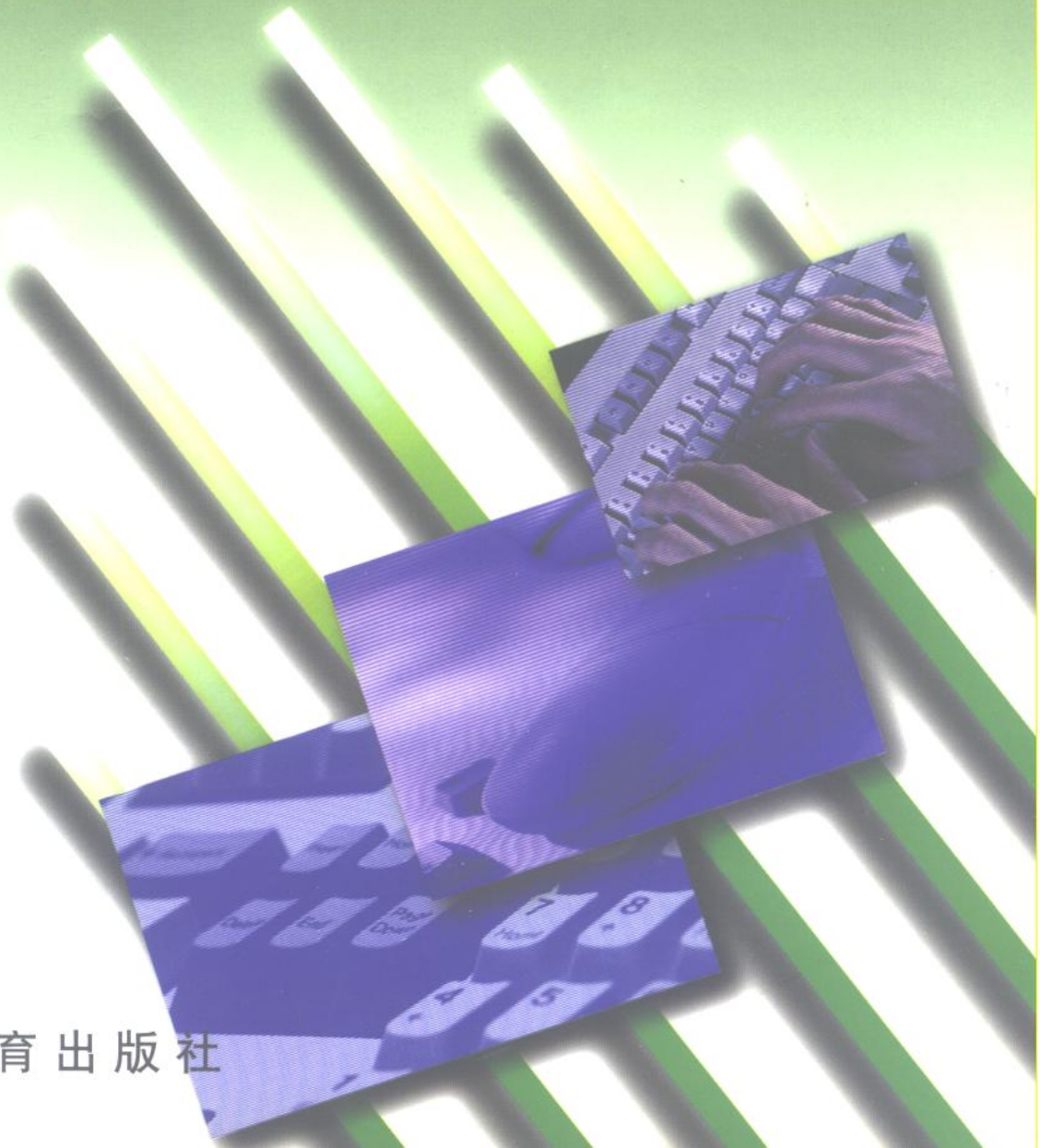
全国计算机等级考试

二级教程

——FORTRAN语言程序设计

教育部考试中心



高等教育出版社

全国计算机等级考试

二级教程

——FORTRAN 语言程序设计

主编　徐士良
编者　徐士良　杨明福　王卫兵

高等教育出版社

(京)112 号

内 容 提 要

本书是根据教育部考试中心制定的《全国计算机等级考试大纲》中对二级 FORTRAN 语言程序设计的要求而编写的。内容包括程序设计的基本概念、FORTRAN77 语言、简单数据结构以及常用算法。目的是为了帮助读者掌握考试大纲对二级 FORTRAN 语言程序设计的要求。

本书内容精练,实例丰富,不仅是应试者必备的自学和辅导教材,也可作为一般院校 FORTRAN 语言程序设计课程的教材或自学参考书。

图书在版编目(CIP)数据

全国计算机等级考试二级教程:FORTRAN 语言程序设计/教育部考试中心编 .—北京:高等教育出版社,1998 (1999 重印)
ISBN 7-04-006903-2

Ⅰ.全… Ⅱ.教… Ⅲ.①电子计算机-工程技术人员-水平考试-教材 ②FORTRAN 语言-程序设计 Ⅳ.TP3

中国版本图书馆 CIP 数据核字(98)第 21296 号

*

高等教育出版社出版
北京沙滩后街 55 号
邮政编码:100009 传真:64014048 电话:64054588
高等教育出版社发行
国防工业出版社印刷厂印刷

*

开本 787×1092 1/16 印张 17.5 字数 430 000
1998 年 8 月第 1 版 1999 年 1 月第 3 次印刷
印数 19 167-29 176
定价 24.80 元

第二届全国计算机等级考试
委员会名单

主任委员： 杨芙清

副主任委员： （以姓氏笔画为序）

朱三元　杨学为　应书增　罗晓沛　谭浩强

委　　员： （以姓氏笔画为序）

王义和	王申康	边奠英	古天祥
齐治昌	仲萃豪	刘淦澄	刘瑞挺
李克洪	吴文虎	吴功宜	沈钧毅
杨　洪	杨明福	林卓然	施伯乐
钟津立	侯炳辉	俞瑞钊	张福炎
袁开榜	席先觉	唐兆亮	徐沪生
钱维民	潘桂明	鞠九滨	瞿　坦

秘 书 长 徐沪生(兼)

全国计算机等级考试
系列用书编审委员会名单

大力推行全国计算机等级考试 为发展知识经济、信息产业和培养计算机 专门人才作出贡献

中国科学院院士
北京大学计算机科学技术系主任
全国计算机等级考试委员会主任委员
杨芙清

当今，人类正在步入一个以智力资源的占有和配置，知识生产、分配和使用为最重要因素的知识经济时代，也就是小平同志提出的“科学技术是第一生产力”的时代。科教是经济发展的基础、知识是人类创新的源泉，基础研究的科学发现，应用研究的原理探索和开发研究的技术发明，三者之间的联系愈来愈紧密，转换周期日趋缩短。世界各国的竞争已成为以经济为基础，以科技特别是以高科技为先导的综合国力的竞争。

在高科技中，信息科学技术是知识高度密集、学科高度综合、具有科学与技术融合特征的学科，它直接渗透到经济、文化和社会和各个领域，迅速改变着人们的观念、生活和社会的结构，是当代发展知识经济的支撑之一。

在信息科学技术中，微电子是基础，计算机硬件及通信设施是载体，计算机软件是核心。软件是计算机的灵魂，没有软件就没有计算机的应用。软件产业已成为信息产业的核心和支柱。信息产业的发展，会大大提高我国的总体实力，增强我国在全球的竞争地位。

为了适应知识经济发展的需要，大力推动信息产业的发展，就需要在全民中普及计算机的基本知识，广开渠道，培养和造就一批又一批能熟练运用计算机和软件技术的各行各业的专门人才。

1994 年，国家教委推出了全国计算机等级考试，它是一种重视应试人员对计算机和软件的实际掌握能力的考试。它不限制报考人员的学历背景，任何年龄阶段的人员都可以报考。这就为培养各行业计算机的应用人才，开辟了一条广阔的道路。

1994 年是推出计算机等级考试的第一年，当年参加考试的有 1 万余人；到了 1998 年上半年，报考人员已达 38 万余人。截止至 1998 年上半年，等级考试共开考 7 次，考生人数累计共达 115 万人。其中，有 49 万 4 千人获得了各级计算机等级证书。

事实说明，鼓励社会各阶层的人士通过多种途径掌握计算机应用技术，并运用等级考试对他们的才干予以认真的、有权威性的认证，是一种较好的人才培养的有效途径，是比较符合我国具体情况的。等级考试也为用人部门录用和考核人员提供了一种评测手段。从有关公司对等级考试所作的社会抽样调查结果来看，不论是管理人员还是应试人员，对该项考试的内容和形式都给予了充分肯定的评价。

计算机等级考试所取得的良好效果，也同全国各有关单位专家们在等级考试的大纲编写、试

题设计、阅卷评分及效果分析等等多项工作中，付出的大量心血和辛勤的劳动密切相关，他们为这项工作的顺利开展作出了重要的贡献。

计算机与软件技术是一项日新月异的高新技术。计算机等级考试大纲有必要根据计算机与软件技术在近年的新发展，进行适当的修正，从而使等级考试更能反映当前计算机与软件技术的应用实际，使培养计算机应用人才的基础工作更健康地向前发展。

从面临知识经济的机遇与挑战这样一个社会大环境的背景下，考察全国计算机等级考试，就会看到，这一举措是符合知识经济和发展信息产业的方向的，是值得大力推行的。

我们相信，在21世纪知识经济和加快发展信息产业的形势下，在教育部考试中心的精心组织领导下，在全国各有关专家们的大力配合下，全国计算机等级考试一定会以更新的面貌出现，从而为我国培养计算机应用专门人才的宏大事业做出更多的贡献。

前　言

本书是全国计算机等级考试教程系列丛书之一。本书根据《全国计算机等级考试大纲》中关于二级 FORTRAN 语言程序设计的要求组织编写。

全书共分十二章。除了最后一章是介绍上机考试环境外，其余各章的内容都是结合考试大纲的要求而组织的。主要内容包括：程序设计的概念、FORTRAN77 概述、数据类型及其运算、最基本的语句、选择结构程序设计、循环结构程序设计、数组、函数与子程序、数据联系、字符处理、文件、上机指导。除最后一章外，每章后面都附有习题，习题的形式与等级考试中的笔试题完全相同。

根据考试大纲的要求，二级考试中有五种计算机语言，对于每一种计算机语言都包括对程序设计的概念、简单数据结构与常用算法的要求。在本书中，除了第一章专门介绍程序设计的概念外，简单数据结构与常用算法的内容是放在各章中以程序举例的形式来介绍的。

本书的第一章至第十一章由徐士良编写，第十二章由杨明福与王卫兵编写。

本书的特点是内容精练，语言通俗，实例丰富，紧扣考试大纲。它不仅可以满足参加全国计算机等级考试二级 FORTRAN 语言程序设计的需要，而且也可以作为一般 FORTRAN 语言程序设计课程的教材或自学参考书。

本书由谭浩强教授主审。由于时间仓促以及作者水平有限，书中难免有不妥或错误之处，恳请读者批评指正。

作　者

1998 年 4 月

目　录

第一章　程序设计概念

第二章　FORTRAN77 概述

第三章　数据类型及其运算

第四章　最基本的语句

第五章 选择结构程序设计

第六章 循环结构程序设计

第七章 数 组

第八章 函数与子程序

第九章 数据联系

第十章 字符处理

第十一章 文　件

第十二章 上机指导

第一章　程序设计概念

1.1　程序设计的基本过程

什么是程序设计？对于初学计算机的人员来说，往往把程序设计理解为简单的编制一个程序。其实这是不对的，至少是不全面的。实际上，程序设计包括多方面的内容，而编制程序只是其中的一个方面。有人将程序设计描述成如下公式：

程序设计＝算法＋数据结构＋方法＋工具

由这个公式可以看出，在整个程序设计的过程中，要涉及到算法的设计、数据结构的设计、方法的设计和设计工具的选择等几个方面。一般来说，程序设计的过程可以分为以下五个基本步骤：

(1) 问题的分析；

(2) 算法的设计；

(3) 流程的描述；

(4) 调试与运行。

下面分别对这五个步骤作简要的介绍。

1.1.1　问题分析

问题分析是进行程序设计的基础。如果在没有把所要解决的问题分析清楚之前就想着手编制程序，是很难得到预想结果的，这只能起到事倍功半的效果。根据所要解决的问题性质与类型，需要分析的内容可能是不同的，但作为最基本的分析内容主要有以下几个方面。

1. 问题的性质

人们所要解决的问题是各种各样的，而对于不同性质的问题，所用的方法、工具以及输入输出的形式一般也是不同的。例如，你应当明确所要解决的问题是属于数值型还是属于非数值型的问题；如果是数值型的问题，那么要求确定一个合理的精度要求。不管是数值型问题还是非数值型问题，都需要明确最终的结果是什么，如对于一元二次方程求根的问题，需要明确是只需要实根还是实根与复根都需要。

2. 输入/输出数据

数据处理是计算机应用中最广泛的领域。在用计算机解决问题时，一般总要有一些输入数据，计算的结果也要以某种方式进行输出。因此，在进行程序设计的过程中，需要考虑对输入/输出数据的处理。一般来说，对于输入/输出数据主要应考虑以下几个方面：

(1) 数据的类型是什么？如整型、实型、双精度型、字符型等。

(2) 在何设备上进行输入或输出？如显示或打印。

(3) 采用什么样的格式进行数据的输入或输出?

3. 数学模型或常用的方法

对于数值型问题,一般要考虑数学模型的设计,或者要对常用的一些方法进行分析和比较,从而根据问题的性质选择一种合理的方案。对于非数值型的问题,通常也需要从众多的方法中选择一种合适的方法。

1.1.2　算法的设计

算法是指为在有限步内解决一个具体问题而规定的意义明确的解题步骤的有限集合。概括地说,算法是指解题方案的准确而完整的描述。从程序来说,也可以说算法是一个有限条指令的集合,这些指令确定了解决某一特定类型问题的运算序列。

算法不同于一般的计算公式。作为算法应具有以下一些基本特征。

(1) 可行性

算法的可行性包括两个方面:一是算法中的每一个步骤必须是能实现的,例如,在算法中,不允许出现分母为零的情况,在实数范围内不能求一个负数的平方根等;二是算法执行的结果要能达到预期的目的。

(2) 确定性

算法的确定性是指算法中的每一个步骤都必须是有明确定义的,不允许有模棱两可的解释,也不允许有多义性。这一性质也反映了算法与数学公式的明显差异。在解决实际问题时,可能会出现这样的情况:针对某种特殊问题,数学公式是正确的,但是按此数学公式设计的计算过程可能会使计算机无所适从。这是因为根据数学公式设计的计算过程只考虑了正常使用的情况,而当出现异常情况时,此计算过程就不能适应了。

(3) 有穷性

算法的有穷性是指算法必须能在有限的时间内做完,即算法必须能在执行有限个步骤之后终止。数学中的无穷级数,在实际计算时只能取有限项,即计算无穷级数数值的过程只能是有穷的。因此,一个数的无穷级数表示只是一个计算公式,而根据精度要求确定的计算过程才是有穷的算法。

(4) 有足够的原始数据

一个算法是否有效,还取决于为算法所提供的数据是否足够。例如,对于指令"如果小明是学生,则输出字母 Y,否则输出字母 N"。当算法执行过程中提供了小明不是学生的情报时,执行的结果将输出字母 N;当提供的情报中只有部分学生的名单,且小明恰在其中,执行的结果将输出字母 Y。但如果在提供的部分学生的名单中,找不到小明的名字,则在执行算法过程中无法服从此指令,因为在部分学生的名单中找不到小明的名字,既不能说明小明是学生,也不能说明小明不是学生。

综上所述,所谓算法是一个过程,这一过程有一组严谨地定义运算顺序的规则,并且每一个规则都是有效的且是明确的,此顺序将在有限的次数下终止。

1.1.3　流程的描述

程序设计的过程,实际上就是确定解决问题的详细步骤,而这个步骤通常称为流程。在程序

设计过程中,根据不同的设计阶段以及具体要求,可以使用不同的描述流程的工具。常用的流程描述工具有以下几种。

1. 自然语言

自然语言是人们在日常生活、工作、学习中通用的语言,一般不需专门的学习和训练就能理解用这种语言所表达的意思。但用自然语言描述程序(或算法)的流程时,一般要求直接而简练,尽量减少语言上的修饰。例如,为了描述计算并输出 $z=y/x$ 的流程,可以用自然语言描述如下:

(1) 输入 x,y。

(2) 判断 x 是否为 0:

若 $x=0$,则输出错误信息;

否则计算 $y/x \rightarrow z$,且输出 z。

2. 算法描述语言

为了说明程序的流程,还可以使用专门规定的某种语言来描述,这种语言通常称为算法描述语言。算法描述语言一般介于自然语言与程序设计语言之间,它具有自然语言灵活的特点,同时又接近于程序设计语言的描述。但必须指出,用算法描述语言所描述的流程,一般不能直接作为程序来使用,最后还需转换成用某种程序设计语言所描述的程序。算法描述语言与程序设计语言最大的区别就在于,算法描述语言比较自由,不象程序设计语言那样受语法的约束,只要描述得人们能理解就行,而不必考虑计算机处理时所要遵循的规定或其它一些细节。

在程序设计过程中,一般不可能在一开始就用某种程序设计语言编制计算机程序,而是先用某种简单、直观、灵活的描述工具来描述处理问题的流程。当方案确定以后,再将这样的流程转换成计算机程序,这种转换往往是机械的,已经不涉及功能的重新设计或控制流程的变化,而只需考虑程序设计语言所规定的语法要求以及一些细节问题。

例如,对于计算 $z=y/x$ 这个例子,其处理的流程可以描述如下:

```
INPUT x,y
IF (x=0) THEN
  OUTPUT "ERROR"
ELSE
  { z=y/x
    OUTPUT z
  }
```

对于这样的描述,只要懂一些英语,是不难理解的。有时为了更简练,在算法描述语言中还可以用一些自然语言。例如,上面的描述可以改为

```
输入 x,y
IF (x=0) THEN
  输出错误信息
ELSE
  { z=y/x
    输出 z
  }
```

3. 流程图

人们在程序设计的实践过程中，还总结出一套用图形来描述问题的处理过程，使流程更直观，易被一般人所接受。用图形描述处理流程的工具称为流程图。目前用得比较普遍的是 N-S 图，有时也称为结构化流程图。本书及其多数教材中都使用结构化流程图，在本书的 1.2.1 节中将介绍这种流程图。为了便于比较，我们先给出用结构化流程图描述计算 $z=y/x$ 的流程如图 1.1 所示。

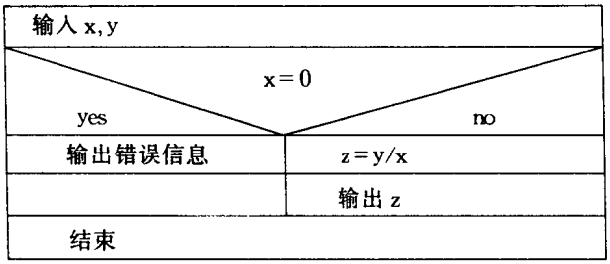


图 1.1

4. 编程

用某种程序设计语言编写的程序本质上也是问题处理方案的描述，并且是最终的描述。但在一般的程序设计过程中，不提倡一开始就编写程序，特别是对于大型的程序。程序是程序设计的最终产品，需要经过每一步的细致加工才能得到，如果企图一开始就编写出程序，往往会适得其反，达不到预想的结果。

下面是用 FORTRAN 语言编写的计算 $z=y/x$ 的程序：

```
      WRITE(*,100)
100   FORMAT(1X,'INPUT X,Y:')
      READ(*,*)  X,Y
      IF  (X.EQ.0.0)  THEN
        WRITE(*,200)
      ELSE
        Z=Y/X
        WRITE(*,300)  Z
      END IF
200   FORMAT(1X,'ERROR ! X=0')
300   FORMAT(1X,'Z=',E15.6)
      END
```

1.1.4　调试与运行

最后编写出的程序还需要进行测试和调试，只有经过调试后的程序才能正式运行。

所谓测试，是指通过一些典型例子，尽可能多地发现程序中的错误。因此，测试的目的是为了发现程序中的错误，而不是为了证明程序正确。

所谓调试，是指找出程序中错误的具体位置，并改正错误。

由此可以看出，测试与调试往往是交替进行的，通过测试发现程序中的错误，通过调试进一步找出错误的位置并改正错误。这个过程需要重复多次。

1.2 程序设计的基本方法

1.2.1 模块化设计

模块化设计是指把一个大程序按人们能理解的大小规模进行分解。由于经过分解后的各模块比较小，因此容易实现，也容易调试。

在进行模块化程序设计时，应重点考虑以下两个问题：

按什么原则划分模块？

如何组织好各模块之间的联系？

(1) 按功能划分模块

划分模块的基本原则是使每个模块都易于理解。而按照人类思维的特点，按功能来划分模块最为自然。按功能划分模块时，要求各模块的功能尽量单一，各模块之间的联系尽量少。这样的模块其可读性和可理解性都比较好；各模块间的接口关系比较简单；当要修改某一功能时只涉及一个模块；其它应用程序可以充分利用已有的一些模块。

(2) 按层次组织模块

在按层次组织模块时，一般上层模块只指出“做什么”，只有在最底层的模块才精确地描述“怎么做”。例如，在图 1.2 所示的层次结构中，主模块只需要指出总任务就可以了，而模块 1、模块 2 与模块 3 分别指出各自的子任务，模块 4、模块 5 与模块 6 才精确描述“怎么做”。

1.2.2 自顶向下、逐步细化的设计过程

自顶向下、逐步细化的设计过程，包括以下两个方面：

将一个复杂问题的求解过程分解和细化成由若干模块组成的层次结构；

将一个模块的功能逐步分解、细化为一系列的处理步骤，直到分解为某种程序设计语言的语句或某种机器指令。

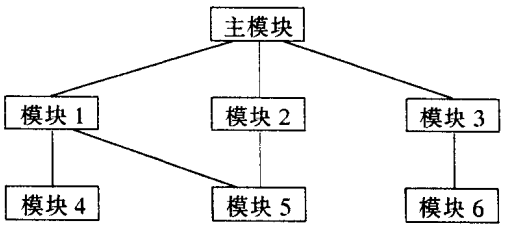


图 1.2

自顶向下、逐步细化的设计过程具有以下两个优点：

(1) 符合人们解决复杂问题的普遍规律,可以显著提高程序设计的效率;

(2) 用先全局后局部、先整体后细节、先抽象后具体的方法设计出的程序具有清晰的层次结构,容易阅读和理解。

图 1.3 表示了计算并打印输出班上某门课程平均分的细化过程。

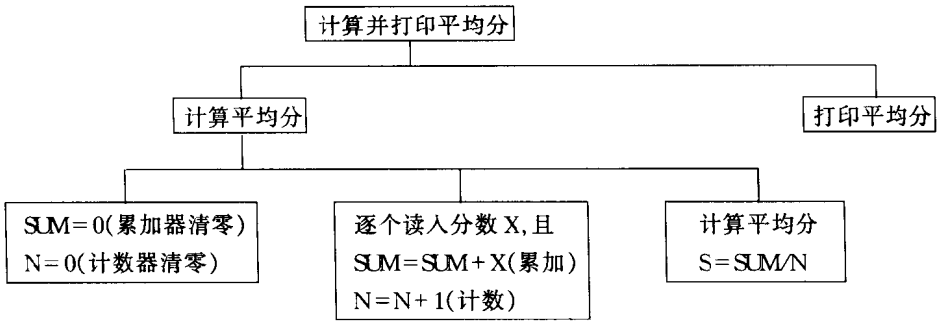


图 1.3

1.2.3　结构化设计

结构化程序设计要求把程序的结构限制为顺序、选择和循环三种基本结构,以便提高程序的可读性。这种结构化程序具有以下两个特点:

(1) 以控制结构为单位,只有一个入口和一个出口,各单位之间的接口比较简单,每个单位也容易被人们所理解。

(2) 使人们直接从程序文本方便、正确地理解程序的功能。

下面简要介绍三种基本控制结构的形式。

1. 顺序结构

顺序结构的流程图如图 1.4 所示。在图 1.4 中,块 S_1、S_2、S_3 是按顺序执行的。

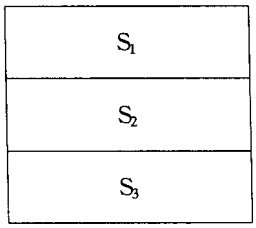


图 1.4

2. 选择结构

选择结构分为两路分支选择结构和多路分支选择结构,其中多路分支选择结构又称为分情形选择结构。

(1) 两路分支结构

两路分支选择结构的流程图如图 1.5 所示。