第3章

银行常用程序设计语言

程序设计语言是用来编写计算机程序的语言。程序设计的好坏不仅影响到程序的使用是否方便，而且涉及程序员编写程序的质量。本章首先介绍结构化程序设计的基本原理，然后对银行常用的几种程序设计语言进行简要的介绍及比较分析。

3.1 结构化程序设计基本原理

结构化程序设计，是著名学者 E. W. 戴文斯特拉和 C. A. R. 霍尔于 20 世纪 60 年代后期提出的一种开创性的程序设计方法，它已成为当今计算机程序设计的重要基石和先进工具之一。结构化程序设计有以下基本特征。

(1) 程序设计＝算法＋数据结构＋程序设计方法学＋计算机语言，其核心和基础是算法。

(2) 从根本上讲，计算机程序是算法的具体表述形式之一，即用计算机语言描述的算法。

(3) 程序编码，即用计算机语言编写的程序，是算法构造(即算法设计)的最终表现形态。

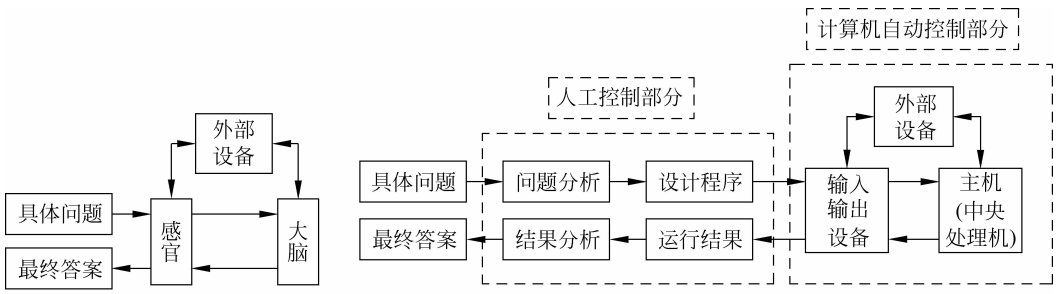
简而言之，离开了算法这个计算机程序设计的基础和关键，程序及其编码就成了无木之林、无源之水。

3.1.1 程序设计基本要素

1. 解题的基本模式

人与计算机解题的基本模式如图 3-1 所示，这表明计算机只是部分地取代了人脑解题的功能，因此应用计算机解决各种不同的实际问题，没有人的积极参与是不行的。从计算机科学来讲，真正客观地把“人”与“计算机”有机地统一起来，组成以人为主导、以计算机为主体的，各自发挥特长与效能的强大而完整的现代信息处理高级系统“人-计算机系统”，正是计算机程序设计(computer program design)。

结构化程序设计，总是从问题分析开始的，是借助计算机解决具体问题的基本过程。即根据给定问题的性质和要求，考虑计算机系统的性能和特点，采用计算机科学的方法和技术，实现“以人为主导、以计算机为主体”解决给定问题的辩证统一过程。通常可概略地分为



(a) 人解题的基本模式　　(b) 电子计算机解题的基本模式

图 3-1　人与计算机解题的基本模式

问题分析、算法设计、程序编码、分析调试、运行维护 5 个基本阶段。

2. 问题分析

问题分析阶段一般可进一步分解成如下环节：精确描述问题，识别数据输入，判定信息输出，正确设置变量，建立数学模型，决定处理方式，选择计算方法等。其中，建立数学模型与选择计算方法尤其重要。

1）精确描述问题

精确描述问题，是指必须准确无误地认清所给问题，并精密无差地明确解决它的系统目标。这一环节倘若有疏漏，则将“差之毫厘，谬以千里”。显然，在精确描述问题时，应力求考虑周密、分析详尽、描述精确，而避免草率行事、似是而非、描述失当。同时，还应当防止错误化、简单化、复杂化这 3 种不良倾向。

例 3-1　错误化的问题描述：如把求解一元二次方程错误地当成求解二元一次方程组或一元一次方程来处理，或者把建立职工工资表的问题混同于学生成绩表问题。

例 3-2　简单化的问题描述：如在运动员运动成绩名次处理中，最容易发生的不完备问题描述是忽视“同成绩者必须并列同一名次”的原则，而误将名次排列问题简单地等同于成绩排序问题。

例 3-3　复杂化的问题描述：如求一元二次方程实根问题，误被扩大为还要考虑求其虚根。

一般来说，成功的问题描述通常是给定问题领域的终端用户专家与计算机的应用开发专家互相取长补短、共同切磋、通力合作的产物。因此，双方应针对有关此问题的各种情况进行有效的交流，以便对给定问题尽可能考虑周详、表达准确、描述完整。显而易见，凡从事或参与计算机应用与开发者，应当自觉养成善于学习、勤于总结、擅长积累、长于合作、精于描述的良好风尚。

2）识别数据输入

识别数据输入，是指在解决给定问题所涉及的若干数据及其处理过程中，识别哪些数据是需要从外部提供（即输入）给计算机系统，并决定如何提供。计算机系统处理的数据，按其产生的方式，可分为输入性数据与非输入性数据（由计算机系统本身自动生成）。因此，正确辨别输入性数据与非输入性数据，对提高计算机解题的工作效率与适应能力意义重大。

例 3-4　求任意给定的一元二次方程 $ax^2+bx+c=0$ 的根。需要输入它的系数 a，b，c 的值（因为它们均为输入性数据），并以采用键盘人工输入的方式为宜。而其根 x_1 和 x_2 的

值就不需要采用输入方式来获取(因为它们都是非输入性数据),可用求根公式和韦达定理由计算机自动计算来求得。

例 3-5 建立职工工资表,应根据每个职工输入其基本数据:工号、姓名、各项应得工资基础数据(如基本工资、工龄工资、物价补贴、奖金……)、各项应扣工资基础数据(如房租费、水电费、互助储金、罚款……)等输入性数据,且以采用由磁盘文件或数据库文件输入的方式为妥。而非输入性数据如应得工资、应扣工资与实发工资可不用输入方式来获得,而用求和或求差的运算公式由计算机计算生成。

3) 判定信息输出

判定信息输出,是指在解决给定问题过程中,必须使终端用户得到所关心的信息。即把人们真正所需的信息从计算机系统内输出,并决定怎样输出。在这一阶段,应当正确判定为解决给定问题所必需的信息输出及其适度的输出总量,同时还应当防止信息输出量不足与过剩两种不良倾向。若信息输出量不足则不能完整地给出符合要求的正确答案,而信息输出量过剩则会造成计算机资源的严重浪费。

例 3-6 在输出职工个人的工资信息时,若只输出职工实发工资数而没有同时输出其各项应得与应扣工资等基础信息,则会因其信息输出量不足而使人们不知自己为什么会得此实发工资数,从而增加财务人员不必要的解释工作量,造成工资发放工作的被动。同样,若再多余地给出诸如婚否、年龄、性别、学历之类与工资发放无关的人事管理信息,则毫无意义。

4) 正确设置变量

正确设置变量,是指对所论问题的数据输入、加工处理、信息输出等主要操作中所涉及的数据(包括原始数据、中间结果、最终结果等),必须采用适当的变量形式来描述、处理和存储(如图 3-2 所示)。正确、合理地设置所需各变量,是计算机应用技术的基本功之一。

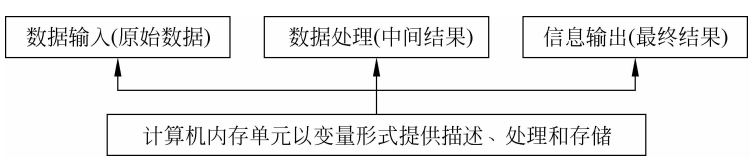


图 3-2 数据变量的设置

例 3-7 在求解一元二次方程 $ax^2+bx+c=0$ 的实根时,通常可选用变量 a,b,c,x_1,x_2,以分别表示所给一元二次方程的二次项系数、一次项系数、常数项、实根 1、实根 2。显然,如果只选用 a,b,c 这 3 个变量,则将不便于求根处理;而若再增加 x,x_3,x_4 等无关的变量,则既无必要又不经济。

5) 建立数学模型

建立数学模型,是指在对给定问题进行了科学的定性分析基础上,进一步抽象出对此问题的定量化数学描述形式。一般来说,通过数学模型来处理并解决给定问题,比直接对原始形态下的非数学模型进行处理更为深刻、简明、有效。因此,建立数学模型是问题分析阶段的核心工作,它是借助计算机解决给定问题的基本前提。

通常,有许多问题可以抽象描述为数学解析表达式,例如求解一元二次方程,解线性方程组,投入产出等。但是必须指出,也有相当多的问题未必总能抽象为相应的数学解析表达

式，例如职工工资表处理，银行业务处理，图书资料检索等。因此，数学解析表达式仅仅是数学模型的主要形式之一，而切不可误以为“数学模型就是数学解析表达式”。

应当指出，数值领域问题通常易于建立其数学模型（例如，数值积分、数值微分、线性规划、动态规划、线性方程、存储问题、排队问题等都有较成熟的数学模型），而非数值领域问题（例如排序、查找、分类、索引等）则常常难于建立其数学模型。因此实际应用中，如果所论问题已有其相应的较成熟的数学模型，则应优先考虑借用或改进现有数学模型；否则，就应当由设计者利用解析法，或数值法、直观法、试探法、类推法等，自行构造相应的数学模型。

由于描述同一问题的数学模型未必唯一，因而努力挑选出能更恰当地反映和更简捷地解决给定问题的优化数学模型至关重要。

例 3-8　在求解一元二次方程实根时，若已求得一个实根 $x1$，则另一实根可用如下两种数学模型之一求得：一是利用求根公式，二是利用韦达定理。显然，第二个数学模型比第一个数学模型更为简易、精确。

6）决定处理方式

决定处理方式，是指对给定问题究竟是采用人工处理方式，还是采用计算机处理方式，应视所论问题的实际情况而权衡利弊后慎重决定，因为并非总是以计算机处理方式为上策。计算机应用，实际是当今历史条件下一种新的社会经济活动，因而，应当用现代社会经济的一般尺度——社会效益与经济效益，来衡量和决定是否宜于应用计算机解决给定问题。通常，采用计算机处理方式的判定条件如下。

① 当人工处理方式所需时间和费用比计算机处理方式代价高时。

② 当解决所给定问题的数值计算或数据处理工作量非常大，以至于难以用人工处理方式来进行时。

③ 当同一性质的给定问题重复出现率高，要求多次做类似处理，而且每次解决有关问题的处理过程基本上大同小异（即只是某些数据有所不同而已）。

④ 当要求解决所给问题的响应速度快、信息可靠性好、数据精度高、自动化程度高时。

显然，若要解决的给定问题并不具备上述任何一个条件，则表明该问题的处理不必采取计算机处理方式，而应当考虑采用人工处理方式，并同时终止以下各阶段的计算机处理过程。

例 3-9　按照上述第①个条件，对某年某月某职工工资表的一次性特殊处理，不宜采用计算机处理方式，因为计算机仅一次性地处理这一特定工资表的代价高于人工处理同一问题的代价。但是，如果希望每年每月都能尽快建立某单位的当月职工工资表，则采用计算机处理方式，因为它符合上述第③和第④两个条件。

例 3-10　对任意给定的 $n(n\leqslant 10)$ 个实数，要找出其中最大者与最小者，自然用人工处理最为简便；但是对任意给定的 $n(n\geqslant 10^{10})$ 个实数，同样也要找出其中最大者与最小者，那就不是任何一个人毕生所能完成的（因为以每秒处理 1 个数据的速度估算，要处理完 10^{10} 个数据竟要 300 年以上），因此只能采用计算机处理方式。

7）选择计算方法

选择计算方法，是指必须精心挑选适合所论问题的最优计算方法，因为计算方法得当与否，其效果往往差别很大。一般来说，描述给定问题的数学模型不一定是便于计算机处理的数学表述形式，因而通常还必须精心选择能将指定数学模型转化为适于计算机处理的计算

方法(因为表述同一数学模型的计算方法常常不是唯一的)。必须强调指出,计算方法选择得当与否,不仅直接影响到解决所给问题的效率、精度和费用,而且关系到整个工作的成功与失败。

例 3-11 求一元五次方程 $x^5+7x-6=0$ 的实根(绝对误差精确到 0.000 001)的计算方法,可有如下两种迭代法,但其结果却截然不同。

先看计算方法 1:$x_2=\sqrt[5]{6-7\times x_1}$,其迭代计算过程及结果如下:

迭代次数	x_1	x_2	绝对误差$\lvert x_1-x_2\rvert$
1	0.000 000	1.430 970	1.430 970
2	1.430 970	− 1.320 610	2.751 580
⋮			
10	1.742 900	− 1.440 400	3.183 300
11	− 1.440 400	1.742 900	3.183 300

显然,计算方法 1 无论迭代计算多少次,永远不可能求得合乎要求的原方程的实根。

再看计算方法 2:$x_2=6/(x_1^4+7)$,其迭代计算过程及结果如下:

迭代次数	x_1	x_2	绝对误差$\lvert x_1-x_2\rvert$
1	0.000 000	0.857 143	0.857 143
2	0.857 143	0.795 780	0.061 363
3	0.795 780	0.810 699	0.014 919
⋮			
9	0.807 955	0.807 957	0.000 002
10	0.807 957	0.807 957	0.000 000

至此已经求得原方程的实根 $x\approx 0.807\,957$(其绝对误差不超过 0.000 001)。

比较计算方法 1 与计算方法 2 及其结果,可得出如下结论,对该数学模型 $x^5+7x-6=0$ 而言,计算方法 1 是发散的、失败的计算方法;而计算方法 2 则是收敛的、成功的计算方法。

由此可见,搞好问题分析诸环节的确事关重大,而那种轻视甚至忽视问题分析,对所论问题一知半解或者不甚了解,便贸然急于仓促编写程序的不良习惯,应当坚决戒除。

3.1.2 算法设计初步

继问题分析阶段之后,就进入应用计算机解决问题的关键性基本阶段——算法设计(algorithm design)。算法设计是计算机程序设计的核心,是计算机科学中最基本的重要概念之一。正确学习、理解、掌握这一基本概念,是非常重要的。

1. 系统与算法

系统一般分为可控系统(controllable system)与非控系统(uncontrollable system)。显然,与人类关系密切的不是非控系统(例如:彗星运动系统、雷电生成系统、台风形成系统、……),而是可控系统(例如:车队作战系统、电力调度系统、电器自控系统、……)。

可控系统又可分为施控子系统(controlling subsystem)与受控子系统(controlled subsystem)。可控系统的根本特性是通过其施控子系统对受控子系统的调控作用来对整个

系统实行控制。可控系统的行为方式，即系统运行全过程中施控子系统与受控子系统彼此响应、协同工作的行为方式，其根本模式可抽象为系统算法（system algorithm）。即可控系统为实现系统目标，要求其操作执行者对操作对象，遵照系统所认定的操作方式，遵从系统所设定的控制方式，并一步一步具体施行的有穷操作过程的描述。

因此，如下事实显然成立并且十分重要。

（1）系统目标不同，则其算法必不同。

（2）系统目标相同，而操作执行者不同，则其算法必不同。

（3）系统目标、操作执行者相同，而操作对象不同，则其算法必不同。

（4）系统目标、操作执行者、操作对象相同，而操作方式不同，则其算法必不同。

（5）系统目标、操作执行者、操作对象、操作方式相同，控制方式不同，则算法必不同。

当且仅当系统目标、操作执行者、操作对象、操作方式、控制方式都相同的算法，才是相同的算法。例如：

（1）提供淡水的水井、汲取卤水的卤井、抽取石油的油井虽然都是“井”，但由于其系统目标各异，故它们的勘测打井算法不同。

（2）有人驾驶侦察机与无人驾驶侦察机，其系统目标都是“侦察敌方，获取情报”，但由于其操作执行者各异，故它们的侦察飞行算法必不同。

（3）由某人等分一根长为35cm的线段，其系统目标、操作执行者、操作对象均无区别，若设定不同的操作方式（例如仅用圆规和直尺作图寻找线段中点的几何操作，采用对折等长线条决定线段中点的模拟操作，利用量具测算线段中点的测算操作等），则其等分线段算法必不同。

（4）四川都江堰水利工程系统的系统目标、操作执行者、操作对象、操作方式都不变，但若设定不同的控制方式（如正常水位时的灌溉水量调度控制方式，警戒水位时抗洪水量调度控制方式），则其河水调度算法必不同。

由此可见，系统算法是一切可控系统固有的重要本质特征与统一的基本行为模式，其系统目标、操作执行者、操作对象、操作方式、控制方式5大基本要素，对任何可控系统都是至关重要的，它们之间互相依存、相互协同、共同完成算法的自然联系，可概括成如下公式：

系统算法＝系统目标＋操作执行者＋操作对象＋操作方式＋控制方式

2. 计算机与算法

以人为操作执行主导者、以计算机为操作执行主体者的系统算法，称为计算机算法（computer algorithm），简称算法。但是，它绝不是天然自生的，而是由人在问题分析基础上专门为用计算机解决给定问题而设计出的一类特殊算法。所谓计算机算法，是指“人-计算机”系统为解决给定问题，需要“以人为主导，以计算机为主体”，对所论问题的数据，采取所设定的顺序结构、选择结构、循环结构、并行结构、子算法结构（必要时方可辅以捷径结构）及其结构化组合的控制方式，来组织和控制所认定的操作方式，并一步一步具体实施的有穷操作过程的描述。

事实上，人的主导作用主要体现在：首先是算法设计，即如何针对计算机的特点和要求，来科学地设计解决给定问题的算法；其次是程序编码（或称编写程序，简称编程），即如何按照给定计算机语言的要求和特点，正确地把所设计的算法编写（翻译）成该计算机语言下的程序；最后是结果分析，即如何依据客观规律与问题要求，慎重地甄别计算机执行给定

程序后所得执行结果的正确性、科学性与合理性。

而计算机的主体作用主要体现在：一旦人设计好解决给定问题的算法，编写好该算法所决定的计算机语言程序，并把该程序交给计算机执行之后，计算机将精确、及时、有效地自动执行该程序(必要时可由人辅以适当的配合操作)，并输出人们所希望的执行结果。显然，借助计算机来解决给定的问题，其关键点、重点、难点都是算法设计，而绝不是人们常常误认为的程序编码。因此循着“对→好→巧→妙→绝”的算法设计发展轨迹，努力探索和追求解决同一问题的最优算法，是算法设计人员应有的宝贵品质和良好习惯。

例 3-12 一个最简单的算法描述。

算法		注释
第 1 步	开始	{算法自此开始}
第 2 步	行输出“您好!”	{在某行原样输出该字符串,且每个汉字占两格}
第 3 步	结束	{算法到此结束}

3. 算法的系统目标

算法的系统目标是解决给定问题。对于“人-计算机”系统而言，即在解决给定问题全过程中，算法必须同时为人和计算机两方面都提供及时、可靠、方便、友好的“人-机”工作界面，使人和计算机能得以充分发挥各自特长与效能。算法之所以特别重要和有用，是因为算法被执行(即其操作被一步一步具体实行)后所输出的信息——执行结果，不仅是算法的系统目标的具体实现，也是人们所关心结果的具体兑现。因此，凡能正确实现系统目标的算法，即能向人们提供正确、及时、清晰直观、简明易懂、信息完备的执行结果的算法，都是应当提倡的良好算法；反之，则是应该防止的病态算法甚至错误算法。

例 3-13 试设计并比较“求最小 1,2,3 位数的算术平均数”的良态算法和病态算法。

问题分析：显然，最小 1,2,3 位数必为 1,10,100。若设 x 为用于描述所求算术平均数的变量，并考虑到为人和计算机提供良好界面，则良态算法可用自然语言描述如下：

算法		注释(良态算法)
第 1 步	开始	{算法自此开始}
第 2 步	x←(1 + 10 + 100)/3	{求算术平均值 x 的赋值操作}
第 3 步	行输出“最小 1,2,3 位数的算术平均数为”	{在某行展示最终结果的性质}
第 4 步	行输出 x	{在下一行展示所得算术平均值}
第 5 步	结束	{算法到此结束}

该算法中，赋值操作“x←(1＋10＋100)/3”的算法意义是，把(1＋10＋100)/3 的值存放到变量 x 中。因此，良态算法的执行结果是：最小 1、2、3 位数的算术平均数为␣37.0(其中，“␣”表示一个空格)。

但是，如果设计时考虑不周、处理不当，而误失上述第 3 步操作，便只能得到病态算法。因为它只能输出一个孤零零的数 37.0，而这常会使人(包括算法设计者和算法使用者)迷惑不解——产生诸如“这个 37.0 到底是何物？来自何方?”之类的疑问。自然，如果连良态算法中的第 4 步也误被丢弃，则所得的算法就是全然无用的“病态算法”了。

4. 算法的操作执行者与操作对象

认清算法的操作执行者及其操作对象的性质、特点与效能，是算法设计顺利进行并取得成功的基本保证。

(1) 算法的操作执行者

算法的操作执行者,其主导是人,其主体是计算机。既然其主导者是人而不是计算机,故在算法设计中"以人昏昏,使计算机昭昭"是不行的;既然其主体是计算机而不是人,故在设计算法中,既要充分发挥人的主导作用,又绝不能"越俎代庖"对计算机取而代之。

(2) 算法操作对象

既然算法的系统目标是解决给定问题,算法的操作执行者是"人-计算机",因此算法的操作对象必然是该问题所涉及的数据,在计算机科学中,脱离数据的算法与脱离算法的数据都是同样不可思议的。

计算机的数据,总是属于一定的数据类型,各种类型数据总是以一定的数据形态(即常数、变量、函数、表达式)出现在算法(或程序)中。每一种数据类型,决定本类数据的取值方式与运算方式;每一种数据形态,则具有自己的结构方式与使用方式。属于同一数据类型的不同形态数据(常数、变量、函数、表达式),统称为同型量(或同型数据)。不同计算机语言的数据、数据类型、数据形态各有所不同,但其结构本质与构造实质则都是同构化的。因此,通过对它们共同的典型代表(如表 3-1 所示)与数据形态的学习,必将有利于人们学习并掌握各种计算机语言中的数据类型。为了增强算法(或程序)的可移植性,建议在算法(或程序)设计中尽可能优先使用最基础的数据类型,即整型、实型、字符(串)型、逻辑型、数组型、记录型、文件型、数据库等,特别是前 4 项所构成的标准(内存数据)类型。

表 3-1　同构化数据类型的简要划分示意

<table>
<tr><td rowspan="15">数据类型</td><td rowspan="11">内存数据类型</td><td rowspan="7">基本类型</td><td rowspan="5">低级类型</td><td rowspan="4">标准类型</td><td rowspan="2">数值型</td><td>整型</td></tr>
<tr><td>实型</td></tr>
<tr><td colspan="2">字符(串)型</td></tr>
<tr><td colspan="2">逻辑型</td></tr>
<tr><td colspan="3">其他非标准类型</td></tr>
<tr><td rowspan="2">中级类型</td><td colspan="3">数组型</td></tr>
<tr><td colspan="3">其他非数组型</td></tr>
<tr><td rowspan="4">扩展类型</td><td rowspan="3">高级类型</td><td colspan="3">内存记录型</td></tr>
<tr><td colspan="3">指针型</td></tr>
<tr><td colspan="3">其他型</td></tr>
<tr><td colspan="4">超级类型(发展中各内存数据类型)</td></tr>
<tr><td rowspan="4">外存数据类型</td><td rowspan="2">基本类型</td><td>低级类型</td><td colspan="3">字符、基本项、组合项</td></tr>
<tr><td rowspan="2">中级类型</td><td colspan="3">外存记录</td></tr>
<tr><td rowspan="2">扩展类型</td><td colspan="3">文件</td></tr>
<tr><td rowspan="2">高级类型</td><td colspan="3">数据库</td></tr>
<tr><td colspan="3">其他</td></tr>
<tr><td colspan="4">超级类型(发展中各外存数据类型)</td></tr>
</table>

5. 算法的操作方式与控制方式

算法的操作方式与控制方式,是算法最富有生气、最具有活力、最反映才智的关键基本要素。前者,其形态往往变化不大,并且其构造较简单而具体,有如构成算法的躯体;后者,其形式常常变化多端,而且其构造较复杂而抽象,恰似主宰算法的灵魂。不同计算机语言的操作方式与控制方式各有不同,但其结构本质与构造实质则是同构化的。人们通过对同构

化操作方式与控制方式的学习,将对算法有更深的认识。

(1) 算法的职能操作与操作形式

数据运算与职能操作,是构成算法所认定的同构化操作方式的基础,是用于解决所论问题操作方式的描述手段。当然,不同计算机语言的数据运算与职能操作各有特色,但是无论何种计算机语言都无一例外地以这样或那样的表现形态,至少设置有如表 3-2 所示的同构化的基本数据运算与基本职能操作。它们之间的关系是,前者是后者的素材,后者是前者的应用。实际上,数据及其运算不能独立于职能操作与控制结构之外而存在,它只有在职能操作(例如赋值、输出)与控制结构之中,才能发挥其效能;职能操作也不能独立于控制结构之外而独立存在,它唯有在控制结构之中,才能产生其效力。

表 3-2 算法的同构化数据运算与职能操作的划分示意

<table>
<tr><td rowspan="8">操作方式</td><td rowspan="6">数据运算</td><td rowspan="5">基本数据运算</td><td rowspan="2">数值运算</td><td>整型运算:+、-、*、DIV、MOD(加、减、乘、取商、取余)</td></tr>
<tr><td>实型运算:+、-、*、/、**(加、减、乘、除、乘方)</td></tr>
<tr><td colspan="2">字符串运算:+(连接)</td></tr>
<tr><td colspan="2">逻辑运算:AND(与)、OR(或)、NOT(非)</td></tr>
<tr><td colspan="2">比较运算:>、>=(≥)、<、<=(≤)、=、<>(≠)</td></tr>
<tr><td colspan="3">扩展数据运算(略)</td></tr>
<tr><td rowspan="4">职能操作</td><td rowspan="3">基本职能操作</td><td colspan="2">输入</td></tr>
<tr><td colspan="2">赋值</td></tr>
<tr><td colspan="2">输出</td></tr>
<tr><td colspan="3">扩展职能操作(略)</td></tr>
</table>

所谓算法的职能操作,是指从外部环境向计算机系统提供数据的输入类操作、计算机系统内部加工数据(包括常数、变量、函数、表达式)的处理类操作、计算机系统向外部环境发送信息的输出类操作的总称。所谓算法的基本职能操作,则是指任何计算机语言都必须提供的这 3 类不同性质职能操作中最简单、最基本、最典型代表——输入操作、赋值操作与输出操作。学习并掌握好这 3 大基本职能操作,是算法设计的基本条件。

例 3-14 设计一个算法,求 s=1×2+2×3+3×4+…+9×10 的值。

问题分析:本题实际上蕴含了两个内容,即顺次产生自然数 k(k=1,2,…,9);同时,依次将所产生出的 k×(k+1)的积逐个累加。由于判定是否已生成所需各自然数 k 的工作可先可后,故解决此求和问题,可用如下两个算法之一来实现(其程序流程图如图 3-3 所示)。

```
算法 1                                    注释(求和算法 1)
第 1 步  开始                             {算法自此开始}
第 2 步  s←0;k←1                          {给累加变量 s、计数变量 k 赋初值}
第 3 步  当 k<10 时,顺次执行第 4 步;反之  {当条件成立时,就反复执行
         (即当 k≥10 时),则转到第 7 步    第 4、5 步}
第 4 步  s←s+k×(k+1)                      {累加求和}
第 5 步  k←k+1                            {累计产生自然数}
第 6 步  返回第 3 步                      {循环执行第 3 步}
第 7 步  行输出"s=",s                     {输出所求之和}
第 8 步  结束                             {算法到此结束}

算法 2                                    注释(求和算法 2)
第 1 步  开始                             {算法自此开始}
```

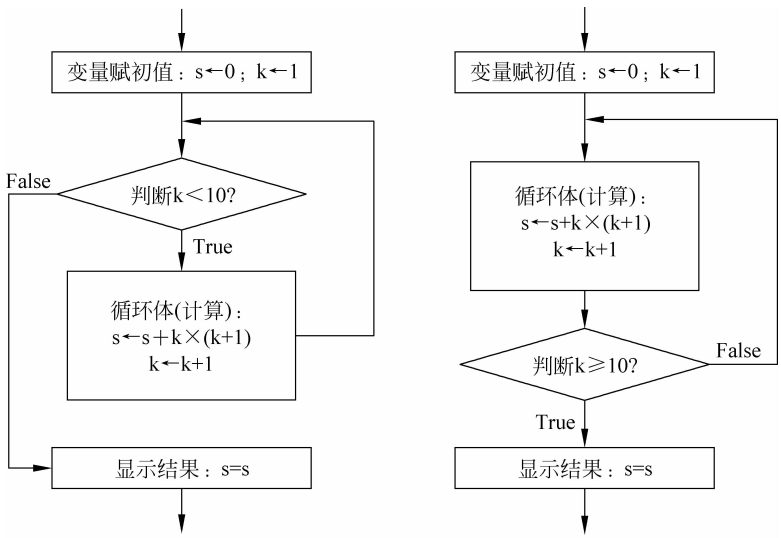


图 3-3　求和算法的程序流程图

第 2 步	s←0;k←1	{给累加变量 s、计数变量 k 赋初值}
第 3 步	s←s＋k×(k＋1)	{累加求和}
第 4 步	k←k＋1	{累计产生自然数}
第 5 步	直到 k≥10 时,顺次执行第 6 步;反之(即 k＜10 时),则返回第 3 步	{直到条件成立时,才不反复执行第 3、4 步}
第 6 步	行输出 s＝s	{输出所求之和}
第 7 步	结束	{算法到此结束}

有关说明如下。

输入操作是指由人向计算机直接提供所需数据并用相应变量存放之的职能操作。

赋值操作的功能是把＝右端的同种数据类型的值赋给它左端的变量,即在计算机加工、处理数据和信息的全过程中,最常见的职能操作是把已获得的某型量(即某种类型的常数、变量、函数、表达式)的值,用一个同型变量存放它——赋值给该变量以备后用。赋值操作是计算机科学中最简单、最重要、最基础的职能操作。

通过赋值操作,人们更能清楚地看到,计算机科学中的变量与数学中的变量的确“既有密切联系,又有本质区别”。即计算机科学中的变量,通俗地说,具有“新的不来,旧的不去;新的一来,旧的必失”的特性。换言之,一个变量一经获得其值(无论是由输入操作获取,还是由赋值操作获取的),则可自行保持该值不变(而根本不管它在算法或程序中是否被使用过多次),直到用新的值来取代它(使该变量另获得其新值)时为止。

输出操作是指,由计算机在打印机上自上而下、从左向右、逐字逐行地,或者在显示屏幕(或 x-y 平面绘图仪等)上可上可下、能左能右、逐点逐字地直观地发送并展示人所需信息的职能操作。

(2) 算法的流程控制与控制结构

流程控制与控制结构,是构成算法所设定的同构化控制方法的基础,是用于解决所论问题的同构化控制方式的描述手段。

① 算法流程控制

所谓流程,是指算法被执行时其(职能)操作实际执行路径(或流向)。一般来说,一个算法可能的执行路径通常都是非唯一的,而且正因为算法执行路径的多样性,才使算法能对所论问题的各种可能情形,都自有其相应的算法解题功能与之相适应。因而不同情况下的实际执行路径应当不尽相同、互不相混。这种以控制算法各流程来适应并解决所论问题的各种情况的执行路径控制方式,称为算法流程控制。

② 算法基本结构

虽然算法形式千姿百态,算法流程纷繁万状,算法构造千变万化,但是就其流程控制模式及其组成形式与构造而言,可以归结成顺序控制结构、选择控制结构、循环控制结构、并行控制结构和子算法控制结构5种。鉴于顺序、选择、循环结构是所有控制模式与控制结构的基础和核心,因此这三者统称为算法基本结构。

它们共同的本质属性是:有且仅有一个入口;有且仅有一个出口;无死块(或死程序段),即永远也无法执行到的操作块(或程序段);无死循环块,即一经开始便永无休止地循环执行下去的操作块。必须强调指出,这4个本质属性缺一不可,否则将破坏算法的结构化,而使之成为非结构化病态算法。

- 顺序结构:这是一种最简单而基础的算法基本结构,是任何从最简单到最复杂的算法都离不开的基本结构。其特点是:在本结构中,各操作按其出现的先后,依次顺序执行。
- 选择结构:这是一种最常用而重要的算法基本结构,是解决任何一个稍有复杂性问题的算法所必不可少的基本结构。其特点是:在本结构中,根据所给定的选择条件为真(即成立)与否,而从各可能的不同分支中选择执行其某一分支的相应操作,它显然有"多者必择其一,且仅择其一"的特性。

例 3-15 如解决"将任给两实数按从大到小排序"的算法就是含有选择结构(即该算法的第3步操作)的最简单算法示例。

问题分析:自然,只需按所给两实数a和b的大小性质来安排其先后排列顺序即可实现排序。同时,a>b或者a<b均可选作两数大小性质的判据。于是,由判据选取的不同可得如下能解决同一问题的两个不同算法:

```
算法                                  注释(算法1)
第1步  开始                           {算法自此开始}
第2步  输入a,b                        {输入任给两实数}
第3步  如果a>b,则输出a,b;            {a>b成立吗?若成立,则先a后b输出之;
       否则,输出b,a                  否则,先b后a输出之}
第4步  结束                           {算法到此结束}
```

显然,若把上述算法中第3步改为如下所示,便可得解决同一问题的另一不同算法:

```
第3步  如果a<b,则输出b,a;            {a<b成立吗?若成立,则先b后a输出之;
       否则,输出a,b                  否则,先a后b输出之}
```

- 循环结构:这也是一种最常用且重要的算法基本结构,是解决绝大多数实际问题的算法所必需的基本结构。其特点是:在本结构中,根据所给定的循环条件为真(即成立)与否,来决定如何重复地循环执行同一组操作。例3-14是含有循环结构(它的第3～5步)的典型算法示例。