DJS-6

计算机算法语言

《DJS-6 计算机算法语言》编写组编

国防工业出版社

DJS-6计算机算法语言

《DJS-6计算机算法语言》编写组 编

国防工业出版社

内 容 简 介

DJS-6 计算机算法语言是以 ALGOL-60为基础的编制计算机程序的语言。它自投入使用以来，经过了多次改进，本书所介绍的是在一九七三年十二月正式定稿的。

本书共分三章。第一章是绪论，举例说明了人与计算机之间的联系问题，并对程序设计语言作了简介；第二章详细介绍了DJS-6计算机算法语言；第三章介绍了 DJS-6计算机算法语言的使用及上机操作。本书内容比较通俗易懂，力图使初学者通过自学或听取简单的讲解，就能开始用该语言编制程序和上机算题，然后通过不断实践逐步掌握它。本书不仅可作为 DJS-6计算机算法语言的使用说明书，也可作为学习类似ALGOL-60的程序设计语言的参考书。

本书主要读者对象是DJS-6型电子计算机的广大用户的工作人员。也可供大专院校计算技术和计算数学专业的师生参考。

DJS-6计算机算法语言

《DJS-6计算机算法语言》编写组 编

国防工业出版社 出版

北京市书刊出版业营业许可证出字第 074 号

新华书店北京发行所发行　　各地新华书店经售

国防工业出版社印刷厂印装

787×1092 1/16　印张 8　178 千字

1975年11月第一版　1975年11月第一次印刷　印数：00,001—20,000册

统一书号：15034·1435　　定价：0.87元

（只限国内发行）

前　言

在毛主席无产阶级革命路线的指引下，在无产阶级文化大革命的推动下，我国电子计算机事业得到了迅速的发展。在我国国民经济的许多领域内，电子计算机越来越发挥其重要的作用。程序设计语言的不断改进，为电子计算机的推广使用提供了更加有利的条件。

DJS-6 计算机（以下简称 DJS-6 机）算法语言是以 ALGOL-60 为基础的语言，它在一九七〇年正式投入使用以来，在各级领导的关怀和支持下，通过广大用户、机器制造单位和原编制人员的共同努力，已经初步得到了推广使用，解决了许多实际问题。在使用该语言的过程中，其语言本身又得到了不断改进。本书所介绍的是在一九七三年十二月正式定稿的。

本书共分三章。第一章是绪论，举例说明了人与计算机之间的联系问题，并对国内外程序设计语言作了简介；第二章介绍了 DJS-6 机算法语言，一方面帮助读者熟悉 DJS-6 机算法语言，另一方面也可供学习类似于 ALGOL-60 的程序设计语言时参考。第三章介绍DJS-6 机算法语言的使用以及上机操作。

本书力求通俗易懂，以使初学者通过自学或听取简单的讲解就能编制程序和上机算题，然后，通过自己的实践便能逐步掌握它。

在本算法语言的编制、推广使用和改进过程中，得到了许多单位的大力支持和帮助，在此，谨致以谢意。

由于编者水平有限、经验不足，本书和本编译程序还可能存在不少缺点和错误，热忱欢迎广大读者和用户批评指正。

DJS-6计算机算法语言

《DJS-6计算机算法语言》编写组 编

国防工业出版社

内 容 简 介

DJS-6 计算机算法语言是以 ALGOL-60为基础的编制计算机程序的语言。它自投入使用以来，经过了多次改进，本书所介绍的是在一九七三年十二月正式定稿的。

本书共分三章。第一章是绪论，举例说明了人与计算机之间的联系问题，并对程序设计语言作了简介；第二章详细介绍了DJS-6计算机算法语言；第三章介绍了 DJS-6计算机算法语言的使用及上机操作。本书内容比较通俗易懂，力图使初学者通过自学或听取简单的讲解，就能开始用该语言编制程序和上机算题，然后通过不断实践逐步掌握它。本书不仅可作为 DJS-6计算机算法语言的使用说明书，也可作为学习类似ALGOL-60的程序设计语言的参考书。

本书主要读者对象是DJS-6型电子计算机的广大用户的工作人员。也可供大专院校计算技术和计算数学专业的师生参考。

DJS-6计算机算法语言

《DJS-6计算机算法语言》编写组 编

国防工业出版社 出版

北京市书刊出版业营业许可证出字第 074 号

新华书店北京发行所发行　　各地新华书店经售

国防工业出版社印刷厂印装

787×1092 1/16　印张 8　178 千字

1975年11月第一版　1975年11月第一次印刷　印数：00,001—20,000册

统一书号：15034·1435　　定价：0.87元

（只限国内发行）

前　言

在毛主席无产阶级革命路线的指引下，在无产阶级文化大革命的推动下，我国电子计算机事业得到了迅速的发展。在我国国民经济的许多领域内，电子计算机越来越发挥其重要的作用。程序设计语言的不断改进，为电子计算机的推广使用提供了更加有利的条件。

DJS-6 计算机（以下简称 DJS-6 机）算法语言是以 ALGOL-60 为基础的语言，它在一九七〇年正式投入使用以来，在各级领导的关怀和支持下，通过广大用户、机器制造单位和原编制人员的共同努力，已经初步得到了推广使用，解决了许多实际问题。在使用该语言的过程中，其语言本身又得到了不断改进。本书所介绍的是在一九七三年十二月正式定稿的。

本书共分三章。第一章是绪论，举例说明了人与计算机之间的联系问题，并对国内外程序设计语言作了简介；第二章介绍了 DJS-6 机算法语言，一方面帮助读者熟悉 DJS-6 机算法语言，另一方面也可供学习类似于 ALGOL-60 的程序设计语言时参考。第三章介绍DJS-6 机算法语言的使用以及上机操作。

本书力求通俗易懂，以使初学者通过自学或听取简单的讲解就能编制程序和上机算题，然后，通过自己的实践便能逐步掌握它。

在本算法语言的编制、推广使用和改进过程中，得到了许多单位的大力支持和帮助，在此，谨致以谢意。

由于编者水平有限、经验不足，本书和本编译程序还可能存在不少缺点和错误，热忱欢迎广大读者和用户批评指正。

目　录

第一章 绪 论

电子计算机作为当代科学成就之一，已经得到了迅速的发展和广泛的应用。从高层建筑的设计到宇宙空间的飞行，从资料检索到天气预报，从精密机床的自动化到大型炼钢炉的控制，从大量实验或测量数据的整理到国民经济计划的编制，从复杂的定理的证明到新产品的研制，不论在现代化工业、农业、商业、文化教育、医疗卫生，还是在现代化国防上，电子计算机都愈来愈发挥其重要的作用，成为非常有用的工具。

电子计算机之所以能完成如此复杂的任务，除了它本身具有高速度的运算能力、大容量的记忆功能和稳定、可靠等特点外，还在于它能执行人们事先以某种方式交给它的工作。为了清楚地说明人们如何把其意图告诉计算机即人机联系的这一问题，本章首先举例分析一下早期使用数字计算机解题的一般过程(着重介绍程序的编制)。然后，再介绍程序设计语言，最后对 JDS-6 机作一简介。

§1.1 解题的一般过程

首先是工农兵和科技人员提出在实际工作中遇到的问题，然后通过各种方法描述成数学问题，接着借助于数值计算方法使数学问题算术化，即把数学问题（例如：开方、三角函数、微分方程、积分等等）简化成适合计算机基本操作（例如：＋、－、×、÷）的近似计算公式，同时要求按近似计算公式所算得的结果，误差要足够小。

然后，人工编制反映计算先后次序的、由一连串旨在命令计算机做什么的“指令”组成的程序（通常称这种程序为代码程序或手编程序）；做好上机前的准备工作(例如把程序连同它需要的数据穿孔在纸带上)，拟定上机操作说明书。接着，根据操作说明书上机操作，例如：启动光电机，把纸带上的所有信息输入到计算机内存储器的一组或几组单元（地址）里，继之计算机执行程序，并借助打印机将程序的计算结果输出。最后计算机用户分析结果，并作必要的事后处理。

综上所述，利用电子数字计算机解题的工作流程大致如下：

物理过程⟶数学描述⟶近似计算公式⟶编制计算程序⟶计算机自动计算（程序及数据代码输入⟶计算⟶计算结果输出）⟶分析结果。

§1.2 程序的编制

下面我们通过一个例子，着重对上述解题过程中的编制程序这个步骤作具体的说明。

例如，甲、乙两个物体从同一地点分别以 a 米/秒、b 米/秒的速度，一个向东，一个向南同时运动，问 t 秒后甲、乙两个物体相距多远（米）？

根据上述物理过程，令 l 是甲、乙两个物体在 t 秒后相距的距离(米)，则有下面的数学表达式：

$$l = \sqrt{x} \times t \qquad (1)$$

$$x=a^2+b^2 \tag{2}$$

采用数值计算方法之一，即牛顿迭代法求$\sqrt{x}$的近似值：

$$y_{n+1}=\frac{1}{2}\left(y_n+\frac{x}{y_n}\right) \tag{3}$$

$$y_0=\frac{1}{2}+\frac{x}{2} \tag{4}$$

上述就是解题过程的前三步。接着就是第四步：编制程序。下面我们就着重介绍一下程序的编制。

为了算某一问题，在没有出现程序设计语言之前，进行手编程序的程序设计员务必先要了解计算机的某些性能。在熟悉机器每条指令功能的基础上，才能编出正确的、好的程序来。

计算机的指令系统是指各种指令的全体。一般来讲，不同的计算机有着不同的指令系统，并且指令形式也不尽一样，但其基本指令都应包含指令操作码和地址码。因此，有下面的形式：

操作码θ	地址码D

其中操作码θ表示本指令执行哪种运算（如+、−、×、÷和逻辑运算等等）；地址码D指示操作单元号或用以存放结果的单元号。例如，在DJS-6机指令系统中，操作码θ＝04，表示二个浮点数相加，即电子计算机的B寄存器的内容加上D的内容●；θ＝06，表示二个浮点数相乘等等。

人工编制程序一般分三个阶段，这就是：绘制框图、编制符号程序、代真。

（一） 绘制框图

框图用于描述计算的进程，尽管它很粗糙，但对程序员还是有帮助的。图1.1给出了例子的算式（1）～（4）的计算框图。其中

ε——要求计算结果的精确度；

→——箭号，指示执行方向；

:=——赋值号，表示把其右端的结果（值）赋给左端的变量；

Ω——常用停机符号。

图1.1所示框图的计算过程是：从框1开始，首先计算a^2+b^2，然后把结果赋给变量x。框2是计算$\sqrt{x}$的初始近似值$\frac{1}{2}+\frac{x}{2}$，并把结果赋给变量y_{n+1}。框3～框5完成用迭代法计算$\sqrt{x}$的近似值。框

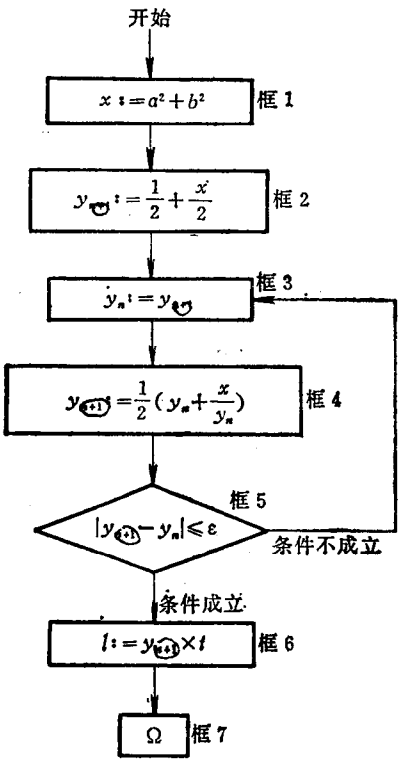

图1.1 例子的计算框图

● 作为运算器（参见§1.4）的一个非核心部件的B寄存器，主要用来存放参加运算的数码和存放运算的结果。

5 说明当新近的两次迭代结果的绝对误差$\leqslant\varepsilon$时，迭代过程业已结束，$\sqrt{x}$的近似值在y_{n+1}之中。否则，即条件不成立，转向框 3，再迭代，直到框 5 的条件成立为止。若框 5 的条件成立，就转向框 6，算出欲求的 l 值，最后到停机。

（二） 编制符号程序

人们要想一下子编出机器语言的程序，往往是困难的。通常的方法是先编制符号程序，其基本思想是要使机器指令形式符号化。例如，用记忆码来代替操作码，用量(例如常量、变量和符号地址等）来代替地址码等等。根据这一思想，不难编出求解上例的DJS-6 机的计算程序（参见表 1.1，其中记忆码取自 DJS-6 机的指令系统)。

表1.1 例子的符号程序

	指令存放单元	符号指令		说明
框 1	$k+0$	→B	a	把 a 送至B寄存器
	$k+1$	×	a	B的内容乘 a，此时B有值a^2
	$k+2$	B→	T_1	把B的内容(a^2)作为临时结果存到工作单元T_1
	$k+3$	→B	b	B有值 b
	$k+4$	×	b	B有值b^2
	$k+5$	+	T_1	B的内容是 a^2+b^2
	$k+6$	B→	x	把B的内容送至x，则$x=a^2+b^2$
框 2	$k+7$	×	0.5	B的内容是$\frac{x}{2}$，即$\frac{a^2+b^2}{2}$
	$k+10$	+	0.5	B有值$\frac{1}{2}+\frac{x}{2}$
	$k+11$	B→	y_{n+1}	把B的内容送至 y_{n+1}，则 $y_{n+1}=\frac{1}{2}+\frac{x}{2}$
框 3	$k+12$	→B	y_{n+1} ←	以下是牛顿迭代程序
	$k+13$	B→	y_n	
框 4	$k+14$	÷	x	
	$k+15$	+	y_n	
	$k+16$	×	0.5	
	$k+17$	B→	y_{n+1}	
框 5	$k+20$	−	y_n	
	$k+21$	|−|	ε	
	$k+22$	⇈	$k+12$	当$\lvert y_{n+1}-y_n\rvert>\varepsilon$时，才转向$k+12$
框 6	$k+23$	→B	y_{n+1}	B的内容是$\sqrt{x}$的近似值
	$k+24$	×	t	
	$k+25$	B→	l	最后把计算结果送至变量 l
框 7	$k+26$	Ω		停机，计算终止

程序从 $k+0$ 开始执行，然后执行 $k+1$，依次执行下去，直到 $k+22$ 为止。$k+22$ 是一条件转移指令，它不改变当前B寄存器的内容，而只是对B的内容作判别。当B的内容大于（机器）零时，它才按地址 $k+12$ 转移，否则执行下一条指令，即 $k+23$。指令 $k+21$ 的功能是二个浮点数的绝对值相减，即B的绝对值减去$|\varepsilon|$。指令 $k+14$ 是一反除指令，即 x 除以当前B的值（除指令是当前B的内容除以 x)。

（三） 代真

代真是一机械过程，其主要内容是把符号程序代真成机器能执行的代码程序。在 DJS-6 机的指令系统中，上述的记忆码对应的操作码如表 1.2 所示。

代真的过程是先对所有的量分配存储单元。例如，上述例子中的存储单元作如下分配：

$k=1000$		1030	1/2
400	a	402	b
404	t	406	ε
410	x	412	y_n
414	y_{n+1}	416	l
420	T_1		

表1.2 记忆码与操作码的对照表

记忆码	操作码
B→	02
→B	03
+	04
−	05
×	06
|−|	15
比	17
甲	63
Ω	77

最后，把符号程序代真成代码程序。例子的代码程序如表1.3所示（机器指令中多余的零的含义请参考DJS-6机指令系统）。

表1.3 例子的代码程序

指令存放单元	机器指令	说明	指令存放单元	机器指令	说明
001000	003 000400	启动	001020	005 000412	框5
001001	006 000400		001021	015 000406	
001002	002 000420		001022	063 001012	
001003	003 000402	框1，计算x	001023	003 000414	框6，计算l
001004	006 000402		001024	006 000404	
001005	004 000420		001025	002 000416	
001006	002 000410		001026	077 000000	框7，计算终止
001007	006 001030	框2，以下是计算初值，循环迭代求$\sqrt{x}$的近似值	001027	000 000000	
001010	004 001030		001030	100 000000	常数$\frac{1}{2}$（指令形式表示）DJS-6机要求浮点数从偶地址开始存放，占二个单元（共48位）
001011	002 000414		001031	000 000000	
001012	003 000414	框3			
001013	002 000412				
001014	017 000410	框4			
001015	004 000412				
001016	006 001030				
001017	002 000414				

总观上述人工编制代码程序过程，显然存在下面几个主要问题：

1）程序完全依赖于机器，以致于甲机上的程序绝对不能在乙机上执行，反之亦然。

2）代码程序的编制既繁琐又容易出错。出了错还难发现，不仅使解题周期大大延长，而且使这种工作往往只有少数专业人员才能担任。

3）程序不易阅读（参见表1.3的程序）。这就不利于各种算法的交流。

4）对程序的修改也不方便，例如，要在程序的某处插入一段程序就非常麻烦。

由于这些问题的存在，使落后的手编程序与向高速、大容量方向发展的先进的计算机之间的矛盾更加突出了。

为了克服手编程序的缺点，曾经想了某些办法，如配备程序库，即：把一些标准程序、标准子程序收集在一起，遇到同类问题，只要抄过来用，或者从磁带中调出来用就行了，不需要重新再编。但是，这些办法都没有从根本上解决问题，它远远适应不了各种应用领域日益增加的、迫切希望使用电子计算机的要求。

能比较彻底解决问题的办法，就是要把电子计算机从计算专业人员的手中解放出来，使应用领域的广大工农兵和科技人员，能够直接掌握使用电子计算机。程序设计语言能够比较满意地承担这个任务。它作为人-机之间相互连系的纽带与桥梁，在过去的二十多年来取得了喜人的进展。

§1.3 程序设计语言

下面我们对程序设计语言作一简单介绍。

程序设计语言就是用来编制程序的某些约定语言。它通过一系列元语言公式和其他规定，在相当程度上明确地规定了程序的结构。通常把利用程序设计语言编制算题的程序称为“源程序”。采用程序设计语言来编制算题的源程序后，电子数字计算机还是不能直接用源程序来计算。因为计算机是一种逻辑电子装置，它只能接受用二进制数表示的指令和数所构成的程序。因此，还必须把源程序“编译”为计算机能执行的程序，通称之为目标程序。不过这个编译工作不是由人来完成，而是由计算机自动来完成。为了使计算机能够完成这一编译工作，人们必须预先给计算机配备一个与程序语言紧密相关的“编译程序”。这种预先由人工编制的编译程序存储在机器的内存储器里，它是计算机必不可少的组成部分。当计算机出厂交付用户使用时，必须同时交付该机器应用某些程序设计语言时所需的编译程序。这种编译程序可以记录在磁带上，或打在纸带上，连同机器一起卖给用户。国外把编译程序这一类不是由电子器件等构成的计算机的设备称之为“软件”(software) 或“软设备”，用电子器件等构成的计算机本体，则相对地称为“硬件”(hardware) 或“硬设备”。事实表明，软件的功能与质量在很大程度上左右着整个计算机系统的功能。据资料统计，如果软件设计或使用得不好，只能发挥出计算机系统效能的百分之三十，在最佳条件下，则可达到百分之七十至百分之八十。

软件的种类繁多，它主要包括程序设计语言、系统软件与应用软件。下面我们仅对程序设计语言中的某些通用的较为重要的语言，作一简单的介绍：

1.3.1 面向机器、过程和问题的程序设计语言

(一) 面向机器的程序设计语言

这种语言是围绕特定的计算机或计算机族而设计的语言，其目的是使程序设计员摆脱计算机所特有的一些细节，而去专心考虑程序之间的内在联系。这类语言的典型代表是汇编语言，又称符号机器语言。它的源程序是§1.2中介绍的符号程序这一级。

这样一来，手编程序中大量机械性劳动（如代真工作）将由计算机来完成，从而大大提高了编制程序的效率。其具体好处表现在以下几个方面：第一，程序设计员无需死记硬背特定计算机的指令系统（对由数百条指令组成的指令系统，这种记忆更难）；第二，省掉了存储单元分配这一环；第三，不要熟记诸如量的对应地址码；第四，程序设计员无需再做数的转换（例如将十进制数转换成二进制数）工作。

因此，尽管面向机器语言的自动化程度不很高，但是由于它具有与机器语言同样的灵活性和效率，而且又比机器语言易于编写程序，所以至今它仍然是一种重要的程序设计语言。尤其是对一些计算量较大或经常使用的问题程序，很多还是用汇编语言编写的。

为了克服面向机器的程序设计语言只适应特定计算机，且自动化程度仍不很高的缺点，

发展了面向过程的程序设计语言。

（二） 面向过程的程序设计语言

这种面向过程的程序设计语言是独立于计算机的。用它所编制的源程序是§1.2中介绍的类似框图这一级，这种语言称之为算法语言。这种算法语言的主要目的是使程序设计人员不必通晓机器的内部逻辑结构，从而更集中精力去推敲解题算法的逻辑和计算过程的描述，至使编出来的语言程序短、占内存储器单元少、计算速度快等。

例如，用DJS-6机ALGOL算法语言可以很简单地编制前述例子的源程序如下：

```
'BEGIN''REAL' a, b, t, E, x, y0, y, l;
        READR(a, b, t, E);
        x:=a↑2+b↑2;
        y0:=0.5+0.5×x;
        'FOR'y:=0.5×(y0+x/y0)'WHILE'ABS(y-y0)'
        'GR'E'DO'y0:=y;
        l:=y×t;
        OUTPUTR(l)
'END'
```

这里用E代替ε。

操作人员只需把这个源程序中的各种符号（关于各种符号的定义和用法将在第二章作详细介绍）穿在纸带上，由输入装置送入计算机，计算机中的编译程序（事先已装在计算机里）便把源程序编译成目标程序，计算机按目标程序进行计算，便可算出计算结果。这样一来，解题工作过程便可以用下述流程表示：

物理过程——→数学问题——→近似计算公式——→用程序设计语言来编制源程序——→计算机自动计算（源程序代码输入——→由编译程序把源程序编译成目标程序——→计算——→计算结果输出）——→分析结果。

目前国际上公认的比较流行的典型的面向过程的程序设计语言是FORTRAN（FORmula TRANslation）、ALGOL(ALGebraic Oriented Language)和COBOL(COmmon Business Oriented Language)。在我国，程序设计语言的研究和使用也正迅速发展和推广。大多数国产电子计算机上所使用的是类似于ALGOL的算法语言（如本书所介绍的DJS-6机算法语言），个别也有类似于FORTRAN的语言。中国科学院计算技术研究所所编制的BCY算法语言也是以ALGOL-60为蓝本的。

下面对几种典型的面向过程的程序设计语言作一简单的介绍：

FORTRAN是科技计算方面的重要语言，也是目前国际上在同类语言中应用最广泛的语言。它用于描述数值计算过程，其基本成分是普通的算术表达式。算术表达式是由数、变量、函数、运算符以及括号组成的。从算术表达式又构成语言中的基本语句，称为赋值语句。为了使计算过程有选择或循环迭代的功能，语言中加入某些控制语句，如转语句、条件语句和循环语句等。因为这些语句要涉及其他语句，所以语句前可带标号。另外，还引入一些非执行语句，表明量的性质、量之间的关系等，如维数语句、等价语句和公用语句等。一系列语句组成程序段，而FORTRAN程序是由一个或若干个程序段组成的，其

中有一主程序段，而其余的是函数段或过程段。程序由主程序段第一个语句开始执行，在执行当中，它可以访问其他辅程序段。

ALGOL 算法语言系指 ALGOL-60，它自 60 年开始，成为国际语言。ALGOL 语言类似 FORTRAN 语言，也是描述数值计算过程的。两者相比，ALGOL 语言有如下主要特点：第一，用巴科斯范式定义的语言，ALGOL 是一个重要的先例；第二，ALGOL 程序是嵌套结构的，而 FORTRAN 是程序块结构的；第三，ALGOL 中引进递归过程，动态存储地址分配、分程序结构、固有变量和固有数组等概念；对标识符加入完备的说明，因此语法检查较为全面。

ALGOL 语言虽不如 FORTRAN 语言使用广泛（就国际上而言），但作为国际通用语言的 ALGOL 已在下面两个方面产生了深远的影响：一是作为一个模型对所有后来的语言的发展产生了影响。例如由美国 IBM（国际商业机器公司）发起的语言 PL/1，它所依据的模型就是 ALGOL 语言；二是引起了对编译程序技术的更多的理论性研究，并产生了发展新的编译程序的“生产线手段”的可能性。ALGOL 语言的严重缺陷是全然没有输入与输出。

COBOL 是为描述商业数据处理问题而设计的语言。由于该语言的统一标准化程度较好，因此，COBOL 程序可以无需改变地从一个系统转到另一个系统。这一点，现今的 FORTRAN 语言还未能真正做到。

（三）面向问题的程序设计语言

这种语言不仅使人摆脱机器的逻辑结构，而且使人不必关心问题的解法和计算过程的描述。这样，人们只要指明输入数据、所要完成的操作以及输出形式，就能得到所要的结果。这类程序设计语言的典型代表是报表语言和判定表语言。

1.3.2 汇集型语言和可扩充语言

随着电子计算机本身的发展及其应用领域的日益扩大，随着许多边缘学科对电子计算机客观要求，专用程序设计语言愈来愈多，同时也促使人们去建立多用途的程序设计语言。例如工程技术人员除了进行科技计算外，还遇到了数据的分类、编辑、化简等数据处理问题；反之，在商业计算中，也要用到统计预测、线性规划之类的科技计算。另外，计算机的发展引进了并行、重迭、中断、分时一些新技术，并增加显示、照象等外围装置。这一切都要求扩充程序设计语言的能力，以适应来自各方面的要求。为了达到这一目的，一般有两种办法：一种办法是吸取各种程序设计语言的特点，使之汇集在同一语言中，这就是所谓“汇集型语言”，它的典型代表是PL/1；另一种办法，就是建立一个基础语言，并提供一种扩充手段。基础语言是所有语言的核心部分，所以又称为核心语言；而扩充手段的目的，是为了使用户能够按照自己的需要，将语言在语法、数据结构、运算和控制等方面进行扩充。例如，一般的可扩充语言有许多原始的运算符和手段，用户可以借助于它们来定义新的运算符。甚至可以重新定义或扩充老的运算符的含义、修改语言的语法等等（这些是汇集型语言办不到的）。用这种办法设计的语言，称之为“可扩充语言”，它的典型代表是ECL。下面分别介绍一下PL/1 和ECL。

汇集型语言 PL/1 (Programming Language/one)，最初是为IBM360 机设计的程序设计语言。PL/1 吸收了 FORTRAN、ALGOL 和 COBOL 各种语言的优点。它是兼用