计算机学术报告会
资料汇编

第二册　程序系统

国防科委情报资料研究所

一九七二年十二月

出 版 说 明

根据各单位的反映和要求，现将一九七二年十一月国防科委司令部主办的计算机学术报告会的有关资料铅印出版。

全部资料共分三册。第一册 主机系统；第二册 程序系统；第三册 宏模组件计算机和HP-35型计算器解剖(阶段)报告。

由于出版时间仓促加上我们业务水平有限，材料内容可能有不妥之处，希望读者发现后及时指正。

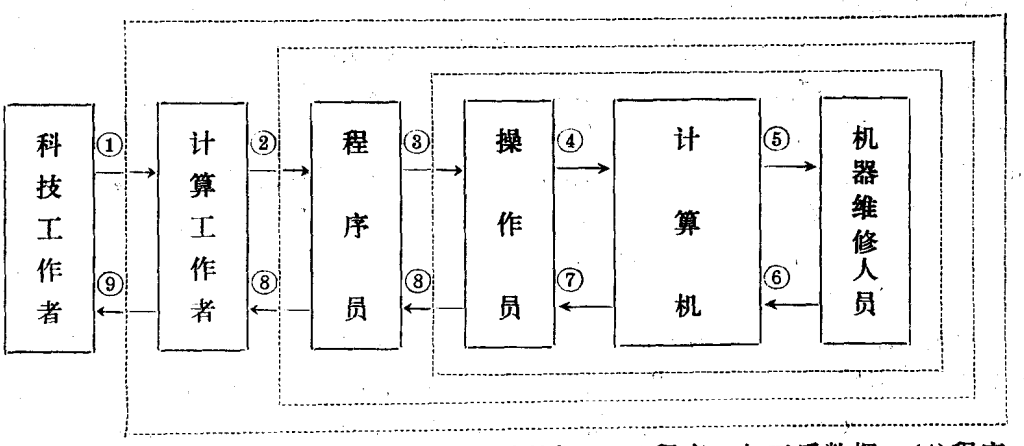
一九七二年十二月

目　　录

程 序 系 统 简 介

§1 引 言

为了便于了解程序系统发展概况及其有关内容，首先让我们分析一下早期利用计算机解题的工作流程。该过程概况分为如下五个环节：提出问题、确定算法、编写程序、上机计算、分析结果。图示如下：



其中，(1)问题、原始数据 (2)算法、原始数据 (3)程序、加工后数据 (4)程序、数据、操作说明书 (5)故障 (6)修复 (7)计算结果 (8)加工结果 (9)最后结果

图 1 解题工作流程

科技工作者提出问题，经计算工作者确定算法、并用适当方式描述出来，交程序员编写程序、加工原始数据、拟定操作说明书，再由操作员上机操作，经计算机运行输出计算结果，最后将计算结果核实、整理交付用户。上述过程未必能轻而易举地完成，每个环节都有自己的难点和烦琐之处，即使专业人员亦难免出现差错，以致有关环节往往还要经过多次修改、调整最后才能得到正确结果。

总观上述流程，其缺点就是：人工干预频繁，而且很多环节工作还必须由专业人员承担，因而不能充分发挥机器效率，使用范围不广泛。为了克服这种缺点，让我们从"系统理论"的观点分析一下图 1 所示的流程，它实质上是一个信息转换过程，每一环节都从上一环节取得信息，并将它加工成为下一环节可以接收的信息形式(图中标明信息的转换)。只不过有些转换是人承担的，有些是机器承担的。然而，由谁承担并非是一成不变的，人承担的部分有可能用机器代替，而机器承担的部分也可以由人去完成。另外，一个单独部分可以分化为若干部分，反之，多个部分也可合并成一个单独部分。如果我们把图中虚线框内的各部分合并成一个单独部分，那么愈是外框的信息形式，就愈便于用户熟悉、掌握。这就是说，要想得到问题的计算结果，外框只要求用户提出问题，中框要描述解题算法，而内框就要编出程序。然而，解决这个问题的关键是：如何组织框内各部分，又如何让机器完成人所承担的信息转

换工作呢？由于计算机能够完成复杂的逻辑运算，人们就事先编好一些程序，利用计算机自身的特点来完成这种转换工作。

所谓程序系统，就是一些事先编好的程序，它不同于普通的算题程序，它是计算机的必要组成部分，其着眼点是：利用计算机本身的逻辑功能，合理地组织整个解题流程，简化或代替各环节中人所承担的工作，从而达到充分发挥机器效率，便于使用者熟悉，掌握计算机之目的。

随着计算机在制造和使用方面的发展．程序系统也迅速发展起来；反过来，程序系统的发展又为计算机的设计提供了新的思想，又为计算机的应用开辟了新的领域。下面我们简单介绍一下，程序系统发展概况及其有关内容。

§2 程序系统发展概况及其有关内容

促进程序系统发展的有三个因素：第一，应用领域的扩大提出建立程序系统的客观要求（必要性）；第二，计算机的发展提供了建立程序系统的物质前提（可能性）；第三，程序系统自身的发展充实了建立程序系统所用的技术（现实性）。显然，对程序系统发展来说，这三个方面是互相联系，又是互相影响的，但是，它们对程序系统发展所起的作用又是有所差异的。本节介绍如下四个方面的内容：早期程序自动化方法和系统，程序设计语言概述，计算机程序系统概述，建立程序系统的有关问题和工具。前两方面内容侧重谈应用领域对程序系统的影响，第三方面内容侧重谈计算机发展对程序系统的影响，最后谈一谈建立程序系统工具的一些问题。

1． 早期程序自动化方法和系统

计算机不断发展，应用领域日益广泛，引出一个尖锐的矛盾，就是日益感到从事计算工作的专业人员不足。一方面，应用领域有大量的问题期待人们利用计算机去加以解决；另一方面，又只有少数计算工作者才能驾驭、使用计算机。最初一个时期，程序设计工作者最为关心的问题就是摆脱手工程序设计，改进自己的工作方法，提高工作质量，以应付日益增长的使用计算机的要求。这一时期，程序自动化工作的重点，就是减轻程序的编写、调整和修改的工作量。

1-1 程序的编写和修改方面

为了减轻手工编写和修改程序的工作量，这一时期发展了程序库方法和符号地址法，建立了装配程序和编辑程序等系统。

程序库方法的基本思想，就是对一些常用的初等函数（如 10→2、2→10、SIN COS 等）、典型的计算方法（如，高斯消去法、龙盖库塔法等）以及标准化的计算问题（如晶体结构分析、水坝应力分析等），事先编好程序，汇集成库（称之为程序库），存放在磁带或磁鼓中。当解题程序中需要它们时，只要提供参数，直接引用有关程序，即可完成所需的计算。这就避免了重复编写和调整这部分程序所花费的时间。特别是，这种程序经过反复推敲，因此还有条数少、效率高的特点。

符号地址法，如同直接使用机器指令编写程序一样，不同的只是用符号代替真实指令中的指令地址或数据地址。在程序开始执行前，由一个称之为“代真程序”的程序，将符号地址代为真实地址，然后再执行代真后的程序。这种方法便于由若干个程序组成一个完整的程序，同时又易于修改程序。

装配程序：这种程序是基于代真程序发展起来的，它的主要功能是把存放在不同外部设备上的程序段以及所引用的库中标准程序，装配成为一个完整的解题程序，这种程序不仅便

于不同装配人员、不同时间编出的程序，而且它又是某些较大系统的组成部分。

编辑程序：它除了具有装配程序的功能外，还具有修改程序的功能，可以增加、删除或替换程序中的某些段落。功能较全的编辑程序，还可以把不同类型语言写出的程序编辑在一起。这种程序是操作系统的组成部分之一。

1-2 程序的调整和检查

用手工编写程序是一项既繁重又琐碎的工作，尤其是对大型问题，要想一次就编出毫无差错的程序几乎是不可能的。然而，任何一个微小的错误都会影响程序的正常运行或得到不正确的结果。因此，为了迅速查出程序中的错误，程序工作者建立了各种帮助检查程序错误的系统。概括说，有二种：一种是程序运行中的，另一种是运行失败后的。

1-2-1 程序运行中的调整系统

这种系统有二种目的：其一是显示程序运行的逻辑线路，称为线路示踪；其二是抽印计算公式中某个变量的值，称为抽印代码。这种系统的基本思想是：根据用户提供的信息，在解题程序的线路关键点或计算公式中插入打印指令；当程序执行时不断打印信息，前者便于程序设计者下机后证实逻辑线路的运行是否是合意图的，后者有助于验证计算公式的正确性。一旦程序调通、正式开工计算时，这种系统可以根据用户的命令，消除插入的指令，以便保证正式计算时的运行效率。这种系统是目前程序设计语言加工系统中的必要组成部分。

1-2-2 运行失败后的系统

这种系统包括有：输出有效存贮、打印现场以及错后检查程序。前两种较为简单，其目的是在用户程序失败后取得较多的信息，以便下机后分析失败原因。第一种输出有效存贮单元中的内容，第二种输出当前指令及其有关寄存器的内容。错后检查程序的目的，是当某段程序运行失败后，检查分析有关程序段，以便查出导致失败的原因和位置。

1-3 模拟系统

由于计算机不断发展，不断更新，随之而来的问题就是：在某台计算机（A）上，如何执行另一台计算机（B）的程序？解决这一问题的重要作用如下：

A．保证已有计算机上的程序，能在新计算机上运行。

B．在设计新计算机时，可在现有计算机上探讨新计算机的特点，统计必要的参考数据。

C．在新计算机交付使用之前，可在现有计算机上编写、调整新计算机上的程序和程序系统。

这种在计算机A上，模拟执行计算机B的程序的系统，称为B计算机的模拟系统。其实现梗概是：把计算机B的程序和初始数据，作为模拟系统的数据输入A计算机中，然后启动A机中的模拟系统程序，它就逐条取B机的指令进行分析，根据地址码取出B机相应的数据，再根据操作码转入模拟系统的不同子程序。如果操作码是运算型的，那么相应的子程序就按该操作在B机上的功能加工已取出的数据，然后继续加工下一条指令。如果操作码是转移型操作，那么相应子程序就按B机指令的功能，判断是否要转移，当需要转移时，模拟系统下一次就分析转移处的指令；否则，就分析下一条指令。

模拟系统种类很多，除模拟机器指令外，还可以模拟各种程序设计语言。在计算机设计自动化中，还可以建立一些模拟程序，模拟整机的体系功能或某部件的逻辑结构。前者称为体系模拟，后者称为逻辑模拟。这二种模拟系统，对新计算机的设计和研制，是很有价值的（参见附件“计算机的设计自动化问题”）。

1-4　查找、分类和排序系统

在科技计算领域中，尤其是在数据管理和信息加工领域中，查找、分类和排序是非常重要的。例如，在大量的参考书目中，企图查找某篇论文，或查找某作者的有关著作；也可以要求机器把所有书目按学科分类；或者要求计算机按标题或作者的字母顺序输出一张参考书目一览表。如果建立了查找、分类、排序的系统后，这些工作将是非常简单的。

设A是一些项目组成的集合，所谓查找，就是在A中找出具有某些性质的项目；而分类是把A中所有项目，按某一性质画分为互不相交的子集；排序就是把A中所有项目，按指定顺序(如，字母顺序或数值顺序)进行重排。

建立这种系统的难点不在于实现的难易，而在于系统的效率。现在仍然不断发表各种查找、分类和排序的方法以及理论研究文章。看来仍有不小的潜力可挖。

早期程序自动化方法和系统，虽然比较简单，自动化程度不高，但是它们对程序系统的发展起了较大的作用。甚至到目前为止，这里介绍的一些系统仍然使用，有些是较大系统的组成部分，有些是当代某些系统的雏型。

2.　程序设计语言概述

虽然，程序自动化工作者早期提出了一些简化程序设计的方法，缓和了一些矛盾，但是，这远远适应不了各种应用领域日益增长的、迫切希望使用计算机的要求。看来，比较彻底解决问题的办法，就是要把计算机从计算专业人员的手中解放出来，使应用领域中的广大工作人员，能够直接掌握使用计算机。这就给程序自动化工作者提出了一个刻不容缓的课题——简化程序设计方法，扩大计算机的使用范围。程序设计语言，作为应用领域和计算机之间的桥梁和纽带，在过去的十几年中取得了喜人的进展，结出了丰硕的果实。有关程序设计语言的过去概况和未来展望，参见美国计算科学专家奇塔姆，在IFIP南斯拉夫会议上所作的报告(参见附件)。我们在这里仅就某些重要程序设计语言的特点，做一些简短的介绍。

2-1　面向机器、过程和问题的程序设计语言

程序设计语言，对用户来说，是描述解题算法的手段；对加工系统来说，是信息的源泉、加工的依据。用户希望语言靠近他所要解决的问题，只要把问题描述清楚，就能得到问题的解答；然而，计算机所能接收的，却偏偏是要由二进位数字组成的程序和数据，甚至对一些极其微小的细节也是严格要求、一丝不苟。两者之间差别很大，其中要经过：提出问题、描述算法和编写程序这三个环节，才能最后完成整个程序设计的任务。早期这些工作完全是由人来承担的，但稍后一个时期，程序自动化工作者为了缩短上述差别、减轻人的工作，针对上述三个环节，提出了各种自动化程度不同的语言，概括分为如下三类：面向机器的语言、面向过程的语言和面向问题的语言。下面分别给以简单介绍。

2-1-1　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，其目的是使程序员摆脱计算机的一些细节问题(无需硬记操作码和地址码，摆脱二、十转换问题和存贮分配问题等)，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。它的主要思想是用符号形式表示机器指令，用记忆码代替机器的操作码，用标识符代替地址码和变址码。

例如，按如下公式计算圆柱体的表面积

$$S=2\times 3.14159\times R\times (R+H)$$

其中，S、R 和 H 分别表示圆柱体的表面积、半径和高。图 2 左端是汇编语言写出的程序，而右端是机器码写出的程序。相比之下，显见汇编语言写出的程序易读、易写。

符号指令地址码	记忆码	符号地址	指令地址	操作码	地址码	
K	:	R	1000	10	2002	机器码程序
	+	H	1	01	2003	
	*	'2'	2	03	2001	
	*	'3.14159'	3	03	2000	
	*	R	4	03	2002	
	⇒	S	5	11	2004	
			2000	3.14159		常数和变量
			1	2		
			2	R		
			3	H		
			4	S		

汇编语言程序

图 2　用汇编语言和机器语言写出的程序

汇编语言更进一步的发展是引进"宏指令"的功能，它的作用是把用户事先定义的某些固定程序格式，能够较方便、灵活地代到程序中需要代入的地方。这种方法扩大了汇编语言的灵活性和编写能力。

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。尤其是一些大型的、反复计算的问题程序，以及大规模的程序系统，很多还是用汇编语言编写。

2-1-2　面向过程的程序设计语言

这种语言是独立于计算机的。它的主要目的是使程序员摆脱机器的内部逻辑，从而集中精力考虑解题算法的逻辑和计算过程的描述。这类语言中最典型的代表是 FORTRAN、ALGOL 和 COBOL。

FORTRAN(FORmula TRANslation)是科技计算方面的重要语言，也是目前应用最广泛的语言。它用于描写数值计算过程，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。计算圆柱体表面积的例子，用赋值语句可以写成：

$$S=2*3.14159*R*(R+H)$$

该语句的意义是：计算右端表达式的值，赋给左端变量 S。

为了使计算过程有选择或重复迭代的功能，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它语句，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。

ALGOL(ALGebraic Oriented Language)语言的前身是IAL(International Algebraic Language)语言。在1960年，于法国巴黎召开了ALGOL国际会议，会上接受该语言作为国际语言，并发表了"算法语言ALGOL-60报告"。ALGOL语言类似FORTRAN语言，也是用于描述数值计算过程的。在两种语言中可以找到很多相似成分。下面我们着重地谈一下ALGOL语言的主要特点。第一，该语言的语法、语义以及三种文本（参考语言、出版语言和机器表示)是经过精心选择、反复推敲过的。整个语言完整清晰，尤以BNF语法描述方法最为明显。第二，ALGOL程序是嵌套结构的，而FORTRAN是程序块结构的。第三，ALGOL中引入递归过程、动态存贮分配等概念；对标识符加入完备的说明，因之语法检查较为全面。

ALGOL语言虽不如FORTRAN语言使用广泛，但由于上述特点，使它在理论和技术上，对以后的程序设计语言和编译算法的发展，有着深远的影响。

COBOL(COmmon Business Oriented Language)是为描述商业数据处理问题而设计的语言，其特点是浅显、易懂。语句本身基本是自我说明的，例如，COBOL语句可以写成：

COMPUTE TAX=MONTHLY PAY * TAX RATE-DEDUCTIONS

该语句的效果是：计算量TAX(税额)，它等于MONTHLY PAY(月薪)乘上TAX RATE(税率)，再减去量DEDUCTIONS(扣除额)。

简而言之，COBOL语言是作为大众化语言设计的。它在发达的英语国家中，已广泛地用于政府部门和商业部门，并在行政和用户的强制要求下，该语言的统一和标准化程度是较好的。

2-1-3 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，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。这类语言的典型代表是报表语言和判定表语言。例如，判定表语言，只要求用户把问题按图3所示的表格形式，提交给计算机，即能得到所要的解答。

规　　则	1	2	3	4
条　件　1	Y	Y	Y	其　它
条　件　2	Y	Y	N	
条　件　3	Y	N	Y	
操　作　1	V	V		
操　作　2	V			V

图3　判定表

表中，Y、N分别表示左端的条件成立或不成立。图三所示表格的效果是：根据规则1、2、3下的Y和N，检查左端相应的条件是否符合，如果某规则下的条件均符合，那么就执行相应规则标有"V"的操作；如果规则1、2、3均不符合，那么就执行规则4（其它）所标识的操作。

2-2 专用语言、通用语言和可扩充语言

随着计算机的发展和人们对计算机认识的加深，老的应用领域日益扩大，新的应用领域层出不穷。由于各领域利用计算机的要求不同，相应的语言也就有所差异。我们把只适用于特定应用领域和问题的程序设计语言称之为专用程序设计语言。不同程序设计语言的差别，首先反映在它所要加工的对象以及施于这些运算对象上的运算。这也就是说，在设计不同的程序设计语言时，首先就是规定它的数据结构和运算。例如，算法语言中的基本数据结构是标量和数组，而运算是普通的算术运算。但是，对图形语言来说，它的基本数据结构应当能够反映各种几何图形，而它的运算应当包括：图形的放大、缩小、旋转、移动以至合并、分离等。专用语言是千变万化的，我们这里难于一一叙述。有关计算机的设计自动化及程序系统设计自动化方面的专用语言，请参阅附件中的相应段落。

专用程序设计语言愈来愈多，这不仅促进了程序设计语言的发展，而且也产生了抽象研究程序设计语言的要求。从本质上来看，所有语言除具有"特性"外，又都具有"共性"。这就促使人们研究语言的总概念，研究其功能，以便寻求一种适合需要的程序设计语言。这些研究中的一个重要课题，就是希望设计一种包罗万象的、具有很多专用语言特点的程序设计语言。这一研究除具有理论价值外，还有很大的现实意义。当计算机用于边缘学科和边缘领域时，自然而然地提出了建立多用途语言的客观要求。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，并增加了显示，照象等外围装置。这一切都要求扩充语言的能力，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：一种办法是搜罗各种程序设计语言的特点，使之汇集在同一语言中；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。用前一种办法构造的语言，称为汇集型语言，它的典型代表是PL/1和ALGOL68；用后一种办法设计的语言，称为可扩充语言，它的典型代表是ECL。下面分别介绍一下PL/1和ECL。

PL/1(Programming Language/one)语言，是在1963年，由IBM公司联合两个用户组织SHARE(科技方面的)和GUIDE(商业用户组织)，为360机设计的程序设计语言。该语言最初称为新语言NPL(New Programming Language)，后改名为多用途语言MPPL(Multi Purpose Programming Language)，最后才定名为PL/1。当时的设计意图，是想使它成为一种充分发挥现代计算机能力的，又兼有各种语言优点的多用途语言。在这种思想指导下，PL/1吸收了FORTRAN、ALGOL和COBOL各种语言的优点。简要说，PL/1语言有如下特点：

A. PL/1程序类似FORTRAN程序，也是由若干主程序块和若干子程序块组成的。但是，在PL/1的程序块中可以嵌入另外的块，因之又有ALGOL的特点。

B. PL/1中引入很多数据属性刻画各类数据的性质。数据结构中，除简量、数组外，还引入了"结构"，用以描述不同种类数据所组成的集合。扩充数据结构的同时，PL/1中还

增加了数组表达式、结构表达式以及符号处理、表格加工等功能。

C. 为了充分发挥计算机的并行、中断的功能，PL/1 中引入相应的语言成分。

D. 语言中增加了"编译时功能"，可在编译时加工修改源程序，以提高结果程序质量和语言的灵活性。

PL/1 语言的优点是功能强，适用于各种领域。但其缺点是语言庞大，在中小型计算机上难于实现，尤其是对于只用其中一部分内容的用户来说，将有很大的一部分程序闲置无用。

ECL 语言是美国哈佛大学计算中心新近完成的结果，它的前身是 ELF。ECL 的基础语言是 EL1，而提供扩充手段的系统称为 ECL。在 EL1 语言中，除通常的算术型、布尔型和字符型等基本数据类型外，还把程序、指示字以及 MODE 作为基本数据类型。其次，除普通的算术运算外，语言还把过程、分程序、循环等都当作值处理。这就使 EL1 语言具有可扩充的本质。为了便于语言的扩充，系统还建立了很多内部过程。从而使 ECL 系统，除了能够对语法、数据结构、运算和控制进行扩充外，还能对运算符赋予新的含义。(详见附件"程序系统自动化研究现状")

2-3 交互程序设计语言和自动程序设计

2-3-1 交互程序设计语言

上机工作的目的不外是调整程序、试算方案或正式计算。但对目前流行的使用计算机的工作方式来说，不管何种上机目的，总要人们事先周密地考虑有关工作的全部细节，并用程序加以描述，直至全部工作准备就绪后才能上机。概括说，这种工作方式的特点，是把工作过程划分为程序准备和上机运行这样两个阶段，也就是说，把人的思考时间和机器的执行时间割裂开来。其缺点有二：首先，在程序准备阶段时，程序员孤军作战，不但没有利用计算机，发挥其助手作用，反而束缚了自己创造性的思考。把大量的精力用于推敲程序的细节，考虑各种逻辑可能性的影响；其次，在机器运行阶段时，如因考虑不周发生意外情况，那么程序员是难于补救，甚至是无法插手的，从而不能及时指挥机器，获得预期效果。

针对上述缺点，人们自然希望计算机不但能够执行编好的程序、算出结果，而且还希望它能在编写程序的过程中，有助于我们选择实现方案、验证程序的正确性、指出错误的原因和所在。这样，机器就不断提供信息，程序员就不断补充、修改自己的程序，直至最后编出全部正确的程序为止。

上述想法在过去是不现实的。因为，编写程序是一项高度思维的过程，是一步一步形成和深化的。不能设想程序员慢慢思考时，会让高速计算机停止工作，这将造成巨大的浪费。然而，利用程序语言能做些什么事，取决于有什么样的计算机。自从计算机引入分时系统后，可以在同一台计算机上执行多个程序。每个程序员都可以利用自己的终端设备直接和机器通讯，并且他可以把自己认为是唯一使用计算机的人。当需要思考时，就停止终端设备，而其它程序照常运行；当需要计算机工作时，只要再启动终端设备，机器就按照命令继续工作。这就使上述想法有了实现的可能性，从而产生了交互式会话程序设计语言。

交互式会话程序设计语言有二个特点：其一会话性，是指程序员和机器彼此构成交谈的双方，程序员可用这种语言命令计算机工作，而机器的回答又使程序员获得更多的信息；其二交互性，是指程序的编写、验证、修改和运行是交错在一起的。这种语言将使程序员和机器紧密结合。程序员可以在终端设备旁考虑并构成他所要的程序，也可以用纸带输入事先编

好的程序，然后等待程序的运行，发生错误时当场修改，重复该过程，直至编出正确程序，算出最后结果为止。交互式会话语言的典型代表有 BASIC、APL、LCC 等。

BASIC(Beginers All-purpose Symbolic Instruction Code)语言取模于 FORTRAN 语言。它是一浅显、易懂的会话程序设计语言，大约学习 2—4 小时即可掌握。源程序可从键盘打入，或自纸带输入。当从键盘打入源程序时，可附带穿出纸带，以避免再次上机时重按键盘。该语言小巧、灵活，附有简便的修改方法，适用于小型计算机，目前广泛使用，颇受用户欢迎。

APL(Aprogramming Language)是一专门为交互性使用而设计的语言，它是在 1962 年由 Iverson 首先发展起来的，目的是为了描述数据处理的算法，但一直未能在计算机上使用。直至 1966—1967 年分时系统出现后，APL 才作为强有力的会话语言，得到广泛的应用。该语言基本上是处理数组的语言，有丰富多彩的加工和构造数组的运算，APL 程序的特点是简洁、紧凑，程序设计方法别具独特风格，表达式中的运算符不分优先顺序，一律从右至左计算，大部运算符都有一目运算和二目运算的含义。

LCC(Language Convensation Computing)是一会话性很强的程序设计语言，这种语言可提供一种更加自然的程序设计方法，即所谓先粗后细法：先概括，后加细，编程序是比较困难的，因此先分段编，后再联成一体，分段编写的程序是不完整的，但可用该语言的加工系统帮助完善，加工系统不断发出信息，指出不完善的地方，程序员就不断地修改补充，直至最后编出完整程序。

2-3-2　自动程序设计

由于计算机广泛使用，已要求建立一些跨越学科和领域的大系统，而普通的一些程序系统将是它的"自然"组成部分，这就是说，目前已开始面临"程序系统工程"时代，为此，前二、三年相继召开了两次国际性程序系统工程会议，计算机网的出现为程序系统工程的发展提供了巨大的可能性，然而，目前迫切需要解决的问题就是缺少必要的、强有力的工具，计算机作图学和程序活化学是两个新的研究课题，它们试图为建立大系统的程序设计方法增加视觉和听觉等感知工具，这些感知工具和交互式会话系统合在一起将提供一些半自动程序设计工具，它们将有助于减轻复杂程序设计的工作量。

然而，作为程序自动化工作的最终目的来说，应当是建立自动程序设计手段，这就是说，只要用户陈述它的问题，而不要求他给出详尽的解决算法，就能得到问题的结果。除了少数专门领域外，大多数领域距此要求还有相当大的距离。这一问题是十分复杂的，野心较大的系统是：用户只要陈述输入输出问题的关系，而后系统就构造一过程，将输入转换为所要的输出。野心稍小的方向是在 1966 年由 Floyd 首倡，并由 King 在 1969 年发展了的系统，称之为自动程序验证系统，它的主要思想是：在算法的每一步附以一组谓词，并断言如果该步之前为真，则执行该步后仍旧为真，然后验证程序系统就穷尽所有可能性，企图证实谓词前后是无矛盾的，这样就断定了程序做了些什么工作。

总观上述，随着计算机应用领域不断向着纵深发展，程序设计方法也不断朝着简单化、大群化、自动化的方向前进。在过去的十几年中，它经历了：手工程序设计、符号机器语言程序设计，高级算法语言程序设计，交互式会话语言程序设计等阶段。然而，摆在程序自动化工作者面前的任务还是十分艰巨的，今后将会不断涌现出用户更加满意的适用于更广范围的程序设计手段!

3. 计算机程序系统概述

计算机程序系统可概括为三大类：一类是面向用户的，一类是面向维修、管理人员的，另一类是面向计算机本身的。

3-1 面向用户的程序系统

这类程序系统是便利用户使用计算机解决自己问题的。为此，首先要使用户易于编写解题程序，其次便利用户调整、修改、装配他的程序。

为了便于用户编写程序，在计算机程序系统中，一方面是分门别类地收集了很多事先编好的程序和资料，以便用户直接引用、查找，其中，按应用领域不同，分别包括有：科技计算，商业应用以及资料管理等方面的程序和资料，而在每个领域中，又按典型方法和标准问题加以划分。把这些程序和资料收集在一起，存于后备存贮器中，就构成应用程序库和资料库。另一方面程序系统提供了各类程序设计语言，其中包括有不同领域、不同用法的程序设计语言(参见本节2)。对于这些语言中的每一种语言，程序系统中都有相应的加工程序，它们把相应语言写出的程序翻译为机器指令程序，并执行它算出计算结果，这些加工程序就构成了语言加工系统。

为了便于程序的调整、修改、装配、编辑以及模拟(参见本节1)，程序系统中也应包括相应的程序，由于这些程序是辅助语言加工系统工作的，故统称为辅助系统。

总括说来，面向计算机用户的程序系统，包括以下系统：

语言加工系统：其中有汇编语言、Fortran、Algol、Cobol、PL/1、BASIC、APL、LISP等。

辅助系统：调整程序、装配程序、编辑程序以及模拟程序等。

应用程序库和资料库：包括有科技计算程序、商业用程序以及资料管理程序等。

3-2 面向维修、管理人员的程序系统

围绕特定计算机工作的，除了计算机用户外，还有机器维修人员和行政管理人员。随着计算机不断扩大，机器维修人员和行政管理人员的工作量亦不断有所增加。为了减轻这些人员的劳动，计算机程序系统中配置如下系统：

诊断修复系统：现代计算机的特点是外围设备多，线路结构复杂，机器维修人员面临着二个重要问题：一个问题是如何检查、维护机器，另一个问题就是如何监督机器正确运行。第一个问题，通常是提供一组完备的调机程序加以解决。当检查维修机器时，就运行这些程序，以便查出损坏或失效的部件、元件乃至线路。好的调机程序不仅能够发现问题，而且要精确地指出失效部件的部位和原因。第二个问题解决的办法，通常是在计算机中配置诊断修复程序，并在各部件中装配诊断线路。当某个部件出现异常现象时，相应部件的诊断线路就立即将故障信息通知给诊断修复程序。该程序就根据送来的信息，分析故障原因，并给出相应措施加以修复。如果分析不出原因、或难于自动给出修复措施时，该程序就输出故障信息，请示机器维修人员，并割断故障部件，等待维修人员替换、检修。发生故障时，诊断系统还应把有关发生故障的时间、部位、原因以及修复情况等记录归档，以作将来系统设计的参考资料。诊断修复系统应尽可能保证主机不停，必要时将某部件断肢，脱机维修；当万不得已，非停主机不可时，应将主存现状保留到后备存贮器中，以便故障排除后继续运行。

日常事务管理系统：该系统的目的是减轻机器管理人员的劳动，其职能如下：

系统运行记录：包括有机器的运行日记，故障的记录以及其它各类统计资料。

用户记录：用户登记录，新用户立户头时记入单位、口令、关键字、帐号等有关信息，当用户违法达一定次数时，给于除名。

会计记录：统计用户各次上机中，使用主机的时间，内存、外部设备的使用情况，以及纸张、纸带、卡片的消耗数量，并根据这些项目，计价收费，记入相应用户的帐目中。

3-3 面向计算机自身的程序系统

早期计算机，硬设备简单，程序系统缺少，采用的是手工操作方式，整个上机过程是在操作员直接控制下进行的。当程序运行中，无论是遇到故障错误、外部传输、手工操作或算完下机时，主机都要停机等待，反之，主机运行时，所有外部设备亦不工作。为了提高主机和外部设备的工作效率，硬设备和程序系统工作者所面临的问题是：压缩故障处理时间，提高主机和外部重迭工作能力，简化或代替操作员的操作，减少相继题目间的上下机交接时间。为此，硬设备相继引进了分时操作、中断处理以及多机、多道、多部件等技术，而程序系统也相应地建立了故障处理系统、输入输出控制系统，数据管理系统以及操作系统等。下面我们分别介绍一下有关系统的情况：

故障处理系统：机器运行中的故障，不外是程序错、机器错或操作错，为了减少故障处理时间，首先是要发现故障及时，其次是排除故障迅速。由于程序系统中建立了各种类型的程序调整系统和机器的诊断修复系统，这就大大压缩了诊断、排除故障的时间，尤其是分时操作、中断处理技术广泛使用以后，故障处理系统更加完备。目前，在一般计算机中，针对各类故障都提供了标准处理措施。一些大、中型计算机中，还引入了某些简便手段，以便用户可以根据自己的需要，在不同的程序段中，针对不同的故障，自行规定特殊处理办法。例如，计算结果上溢时，标准操作是打印运算符及运算对象的值，并请求操作员给出进一步的处理措施。针对这种情况，用户可在某程序段中，规定上溢时不打印信息，而是把结果处理为零，然后继续工作，这样不仅提高了计算机的效率，而且简化了上机操作。计算机更进一步发展，引入了多道程序的功能，这就使机器在出现故障时，不用停机等待操作员进一步处理，而是转去执行其它道程序，所以在操作员考虑对策，按选定措施操作时，几乎不影响主机运行时间。

输入输出控制系统：早期主机与外部设备是串行工作的，主机工作时外部停止工作，外部工作时主机又停机等待，这样，不仅主机未能充分发挥效率，就是外部设备也未能得到合理使用。自从分时、中断技术出现后，主机与外部设备有了并行工作的可能性。当需要某外部设备工作时，主机只要分时启动它，令其工作，然后主机继续工作，这样即可实现主机和外部的重迭工作。当外部工作结束后，外部设备立即发送中断信号，通知主机工作完毕。如果在主机继续工作中，再次需要同一外部设备工作时，那么主机再次分时，并且查看该外部设备上次工作是否完成，完成则启动它执行此次工作，未完则等它完成上次工作后，再启动它。显见，相继二次启动外部设备，如果间隔太短时，主机仍需等待，尤其对输入、输出这些慢外部设备来说，等待时间更长。因此，程序系统中提出了 I/O 通道处理方法，其主要思想是把全部输入信息事先输到一缓冲区中(通常在磁鼓或磁盘上)，当程序中需要输入时，就从缓冲区中取入；类似地，当需要输出时，也把输出内容先输到缓冲区，然后再从输出设备输出，这样就用高速外部设备代替低速外部设备，从而减少主机等待时间。当计算机引入多道技术后，一个更好的想法就是合理搭配各道程序，使它们在使用主机和各类外部设备时各自有所侧重，从而使主机和所有外部设备都在不停地工作，这样就使主机和外部设备的效率

较好地调动起来了。

输入输出设备的另一特点是品种日益增多，从符号贫乏的输入输出设备，发展到多符号输入输出设备和笔绘仪，最近又广泛使用了显示、照象、光笔等输入输出手段，不久将有实用的文字输入输出以及声控之类的手段出现。品种增多不仅增加程序系统的控制要求，而且还要增加这类信息到机器码的转换工作。

概括输入输出系统的功能，应有：信息的收集和发放、信息的转换、并行处理以及输入输出错误的检查和改正等。

数据管理系统：对于大型计算机的程序系统来说，数据种类花样烦多，存放位置又有所不同，系统中所涉及的数据，概括分为三类：用户数据、系统数据、会计数据。用户数据包括用户程序、初始数据、中间结果和最后结果，它们又因语言的种类、输入输出格式、存放的设备以及保存时间的长短，将有所不同；系统数据包括各类系统程序本身和平时为它们收集的统计资料，以及各系统运行时临时产生的数据；会计数据包括有用户一览表，用户户头以及有关帐目等。

为了便于管理使用，通常是把这些数据，分门别类地存于文件之中，而文件又是依据统一格式，按树形结构存于磁带或磁盘上。系统中还提供了增补、注销及合并文件的手段，同时，又有易于查找、转换、调进调出数据的措施。

对于存有重要数据的文件，系统中设有严格的保护措施，以防破坏。在用户的文件中，附有用户自己规定的口令或关键字，只有口令或关键字相符时，才能打开文件，取出数据，这将保证用户数据不致失密。

操作管理系统是一大型控制程序，计算机中所有其他系统几乎都是在它的控制下进行工作的，其主要功能是合理控制机器运行流程，有效管理机器设备、数据文件，乃至日常杂务，从而充分发挥机器效率，简化、代替操作员的操作，压缩相继任务之间的时间间隔。

操作管理系统的组成部分，包括有：任务管理系统、设备管理系统、语言加工系统、数据管理系统、故障处理系统以及任务说明书的解释系统。

如果用户要委托机器完成某项任务时，那么他应提供任务操作说明书，用某语言写出的源程序以及初始数据。当用户发出请求执行任务的信号时，机器立即中断分时，将控制转给任务管理系统，该系统工作如下：

首先是接纳新“任务”，将“任务”挂号，并通过数据管理系统，将用户的任务操作说明书、源程序和初始数据送至缓冲区排队，等待运行加工。

其次是根据机器运行情况及诸“任务”的优先性从缓冲区中挑选“任务”准备运行加工。选定的“任务”经设备管理系统分配所需设备和内存，然后送交语言加工系统编译、执行。

转至语言加工系统后，该系统就从缓冲区取进源程序，检查它是否合乎语法，并将它翻译成机器指令程序，然后执行这个程序。执行过程中，可能需要初始数据，算出中间结果或最后结果，这是通过数据管理系统进行转换、传输的。如果执行过程中，出现故障，此时将由故障处理系统按任务操作说明书加以处理，如此继续，直至算出最后结果，或因故障停止运行时，结束该“任务”，返回任务管理系统。

第三是“任务”完成返回后，注销该“任务”，整理有关结果和信息，计价收费送至外部，等待用户索取。

用户的“任务”是由操作管理系统，按用户任务操作说明书进行控制的。操作说明书的形

式及有关细节参见附件“操作系统简要介绍”。

目前，终端系统已在技术先进国家中广泛使用，计算机网已开始出现，这将为硬设备和程序系统工作者提出更多更新的问题。由于计算机网的工作流程复杂，信息的通路多、流量大，今后的数据管理系统和操作管理系统将更加庞大，更加复杂。除此之外，还应建立大型通讯系统，以适应计算机网发展的需要。总之，计算机及其程序系统在各领域中的作用，今后必将发生更大的影响。

4. 建立程序系统的有关问题和工具

本段讨论为研制程序系统提供工具的各项工作。本来这些工作是为建立编译系统以及编译自动化而提供的，后来在其它程序系统方面也逐步得到应用。

首先是由于建立编译系统是一件很复杂的工作，为了使这工作条理化、系统化，以便能按一定步骤去完成，并为了便于交流阅读、存档维护、查错修改，乃要求寻找较为完整而有效的方法；其次还由于建立编译系统的工作量大、所需人力多、周期长，乃进一步提出了编译自动化的要求。本节所讨论的这些研究工作就是在这样的情况下产生的。

编译系统分为两部分，一是语言，二是编译程序。而编译程序所依据的算法是与所加工的语言的特性密切相关的。

语言可分三个方面去进行研究：一是它的符号之间的关系，即所谓“语法”；二是它的符号与含义的关系，即所谓“语义”；其三是语言在使用方面的问题，即所谓“语用”。

语言的语法可以说是它的“骨架”。为了建立一语言的成系统的翻译方法，首先即应找到一描述程序语言语法的形式工具。这个工作在 1959 年基本上由 J. W. Backus 完成了，这就是有名的巴克斯范式(BNF)。

在这基础上，乃提出了许多语法翻译方法，各有其优点和缺点。目前较受注意的方法有：(1)优先方法：如 Floyd 1963 年的“优先算子文法”，Wirth-Weber 1966 年的“简单优先文法”，Ichbih-Morse 1970 年的“弱优先文法”。这些方法的优点是速度快，所占存储小，其缺点是不能查错，且要求改动文法。(2)递归方法：这方法的优点是易学、易改、易于插入语义程序，且能适应各种复杂语言并要求文法改动较少，缺点是较慢，且因插入语义程序太方便，容易使编译算法流于杂乱。(3)矩阵方法：矩阵方法类型很多，目前最新的一种是 Gries 1968 年提出的，据说是很有效的。这方法插入语义程序不像递归方法方便。(4)由左到右的方法。较实用的是 Deremer 1969 年提出的 SLR(K)，这方法的优点是速度快，限制少，缺点是存储状态较多。

为了使语言有精确的含义，更进一步为了便于自动查错乃至编译自动化，很自然地提出语义形式化的要求。这方面虽然研究工作很多，但能实际应用的较少。有名的工作如维也纳方法(PL/1 的形式描述)，ALGOL68，这两个系统的共同特点是通过给一语言建立一抽象解释系统的办法来给该语言的语义形式化。其优点是这些形式化方法的功能都很强，能描述各种复杂的语义，而缺点是这两形式系统都非常复杂，PL/1 的形式文本达数百页之多。这样的形式系统是难以推广使用的。不过，这两个工作在理论上都极有价值，推动着许多有意义的理论研究工作的开展。

除了上述从描述语言一级进行语义形式化的工作外，还有将编译算法中语义部分进行形式描述的工作，它是编译自动化研究的一个组成部分。有名的如 Feldman 的 FSL。

编译自动化的最根本的任务还是为描述编译算法提供高效的程序系统。一般用来描述编

译系统的语言是汇编语言或带有宏指令扩充的语言系统。这样系统的优点是灵活，但缺点是自动化程度太低，人的工作量太大。为了提高自动化程度，一种途径是建立"编译系统的编译系统"，有名的如 Feldman 的 FSL，Brooker-Morris 的CC。这种系统已实际用于编出某些常用语言的编译系统，但缺点是语言类型变化太大或优化程度要求较高时就显得不够有力。另一途径是所谓"系统程序设计语言"，有名的是 BLISS，其主要特点是在这语言中具有描述各种数据结构并在各种结构上进行运算的功能，它亦便于用来描述机器语言。因此它可用来描述各种程序系统算法，且能产生高效的结果指令(运行速度几乎与人手编无异，行数与人手编比较亦高达 3/4—8/9)。同时它还能在一台机器上为另一台机器编写程序系统。这是在美国很受重视的语言。除上述这些语言外，可扩充语言(如 ECL)也可能在程序系统自动化方面得到应用。(详情请参阅"程序系统自动化研究现状"一文)。

§3 我们的看法和希望

上面就我们目前所了解的情况，对程序设计语言和程序系统的发展情况，作了一些简单介绍，内容很不全面。但是仅就这些内容也能看出我们在开展程序系统工作方面，还是比较落后的。不论是工作的深度或广度，也不论是实际应用或理论研究方面，与世界先进水平还是有较大距离的。虽然，我们对国内外情况了解很少，做的工作也不多，但是按照毛主席关于"**中国人民有志气，有能力，一定要在不远的将来赶上和超过世界先进水平**"的教导，结合我们已往的工作经验，我们对今后工作的开展提出一些不成熟的看法。

1. 我们的看法

程序系统领域很广，研究课题很多，我们认为目前主要精力仍应集中在科技计算方面，具体说来应在如下五个方面开展工作：

A、新语言的调查和设计：设计适合我们需要的语言，开展会话语言及可扩充语言的研究。为了保持语言的相对稳定性，对已有的语言继续推广、完善，发挥更大作用。

B、结果程序的优化工作：在将已有编译程序规范化、典型化过程中，集中研究结果程序的优化，特别是在全局优化方面。

C、操作系统：在目前的程序系统中，它是功能最强、潜力最大的系统，尤其对大型计算机来说，有更重要的作用，应着重研究机器运行流程、数据管理方法以及上了终端设备后的问题。

D、程序系统自动化研究：即建立程序系统的工具，不仅能缩短研制程序系统的周期，而且能使我们更透彻、更有条理地理解程序系统的功能及实现。重点应放在编译程序的自动化上。

E、计算机设计自动化的问题：计算机设计自动化工作事在必行，相应的程序系统工作亦应开展。

2. 几点希望

发展程序系统工作，不仅程序系统工作者要勇于攻关，敢于创新，尽最大努力作好程序系统中的工作；而且还需硬设备方面给予大力协助，硬设备和程序系统两条腿走路，才能取得较理想的效果，现在将我们工作中感到的一些问题，以及今后开展工作将会遇到的一些问题，提出我们的希望，供有关方面参考。

A、现有计算机可靠性差：程序系统(尤其是编译系统)运行时，对机器要求严格，几乎