A. П. 叶尔晓夫著

快速电子計算机編制程序的程序

科 学 出 版 社

А. П. ЕРШОВ

ПРОГРАММИРУЮЩАЯ ПРОГРАММА ДЛЯ БЫСТРОДЕЙСТВУЮЩЕЙ ЭЛЕКТРОННОЙ СЧЁТНОЙ МАШИНЫ

Издательство Академии Наук СССР
Москва 1958

内 容 簡 介

本书系統地論述了苏联在程序設計自动化方面获得的成果之一——編制程序的程序（簡称 ПП）。ПП 能把电子計算机編制程序的大部分工作加以自动化。书内描述了初始信息的編碼規則和程序設計的主要算法，并全面說明了 ПП 的各个部分。书末附有苏联科学院快速电子計算机（БЭСМ）的簡短介紹。

本书可供計算数学和計算技术方面的科学研究人員、工程技术人員以及高等学校数学力学系师生閱讀。

快速电子計算机編制程序的程序

А. П. 叶尔晓夫 著
耿 立 大 譯

科 学 出 版 社 出 版 （北京朝阳門大街 117 号）
北京市书刊出版业营业許可証出字第 061 号
中国科学院印刷厂印刷　新华书店总經售

1959 年 8 月第 一 版　书号：1839　字数：114,000
1959 年 8 月第一次印刷　开本：850×1168 1/32
（京）0001—9,000　印张：4 7/16

定价：0.58 元

致中国讀者

获悉拙著"БЭСМ[1]編制程序的程序"得到中国同志的注意并将其譯成中文出版，甚为高兴.

本书写就后，苏联及其他国家已編出了更为完备、更便于工作的編制程序的程序，因而本书某些部分，特别是某些細节的描述已显陈旧. 針对此种情况，我愿提請初学者注意，不要刻板地掌握本书內容.

但是已有的經驗表明，本书所介紹的編制程序的程序的基本原理，总的說来还是正确和成功的. 而且，这里所用的某些程序設計方法也能用以編制其它非算术性質問題的程序.

因此我們認为，本书在目前对讀者还是有所帮助的，它既叙述了程序自动化的基本原理，又介紹了一个較大的非算术性質的程序.

A. П. 叶尔曉夫

1959 年 3 月

1) БЭСМ 是快速电子計算机的簡称，亦即 быстродействующая электронная счётная машина 各字的第一个字母的組合——譯者註.

序　言

本书介紹苏联在程序設計自动化方面最早获得的工作成果之一，书內所介紹的工作起初是在苏联科学院精密机械和計算技术研究所进行的，后来在 1954—1955 年間改在苏联科学院計算中心繼續进行.

除作者外，参加工作的还有上述各研究所的一些同志. 算术算子程序設計的基本原則是卡勒列夫 (Л. Н. Королев) 提出的. 潘諾娃 (Л. Д. Панова) 和伯捷留金 (В. Д. Поддерюгин) 参加了 ПП-1 的編制工作. 在討論 ПП 的設計方案时，庫罗奇金 (В. М. Курочкин) 提供了一系列宝貴的意見；节省工作单元算法的最后方案也是屬于他的.

建議讀者，在讀本书之前，先熟悉书末附录——БЭСМ 簡介 (参見目录).

1958 年 5 月

目　录

第三部分

ПП 的說明

引　言

自动程序設計是这样一类方法的总称，它能使計算机完成某些工作，这些工作通常在編制算題的程序时要由人来完成的．自动程序設計的工作是借助編制程序的程序（簡称 ПП）[1] 来完成的．

編制 ПП 的必要前提是建立程序設計的算法，亦即确定程序設計的一組形式規則．

这里介紹的 ПП 所实現的程序設計的算法是以一般称为算子法的程序設計方法为基础的．算子法的基础系由李雅普諾夫（A. A. Ляпунов）教授于 1953 年的程序設計講义中首先提出．这方法的主要特点是把任一程序看作为一个复合算子，該算子对計算机存儲单元的內容进行一定的操作，而它本身系由不同类型的初等算子所組成．

为了說明算子法的本質，并了解其基本概念和定义，我們来看表 1 的程序[2]．

此程序用以計算 n 阶方陣（存放在存儲单元 $a+1$ 至 $a+n^2$ 中）和列向量（存放在单元 $b+1$ 至 $b+n$ 中）的乘积．所得的列向量 $c_i(i=1,\cdots,n)$ 再按下式变換：

$$c_i' = \begin{cases} c_i + n & c_i \leqslant 0 \\ c_i + 1 & c_i > 0 \end{cases} \qquad i = 1, \cdots, n.$$

如考查該程序的指令在解題过程中所起的作用，則我們发現，

---

1) ПП 是俄文 программирующая программа（中文譯为編制程序的程序）两字的縮写——譯者註．

2) 这里及以下各处指令皆用下法表示

θ	a	b	c

其中 θ 是操作碼；a, b, c 分别是第一，第二，第三地址．
如还須指出指令的地址，就在左边再加上写有指令地址的一列．

表 1

$K+1$	,ПЧ	$K+29$		$K+6$
$K+2$	,ПЧ	$K+30$		$K+7$
$K+3$	,ПЧ	$K+31$		$K+8$
$K+4$	ПЧ			$a+1$
$K+5$	ПЧ			$a+2$
$K+6$	ПЧ			$c+1$
$K+7$	×	$a+1$	$b+1$	a
$K+8$	+	$c+1$	a	$c+1$
$K+9$	СК	$K+7$	$K+35$	$K+7$
$K+10$	+	$a+2$	$K+40$	$a+2$
$K+11$	<	$a+2$	$K+41$	$K+7$
$K+12$	,−	$K+7$	$K+36$	$K+7$
$K+13$	СК	$K+6$	$K+37$	$K+6$
$K+14$	СК	$K+8$	$K+38$	$K+8$
$K+15$	+	$a+1$	$K+40$	$a+1$
$K+16$	<	$a+1$	$K+41$	$K+5$
$K+17$	,ПЧ	$K+32$		$K+19$
$K+18$	,ПЧ	$K+33$		$K+24$
$K+19$	ПЧ	$c+1$		a
$K+20$	<		a	$K+23$
$K+21$	+	a	$K+41$	a
$K+22$	ИЦУК			$K+24$
$K+23$	+	a	$K+40$	a
$K+24$	ПЧ	a		$c+1$
$K+25$	СК	$K+19$	$K+39$	$K+19$
$K+26$	СК	$K+24$	$K+37$	$K+24$
$K+27$	<	$K+24$	$K+34$	$K+19$
$K+28$	СТОП			
$K+29$	ПЧ			$c+1$
$K+30$	×	$a+1$	$b+1$	a
$K+31$	+	$c+1$	a	$c+1$
$K+32$	ПЧ	$c+1$		a
$K+33$	ПЧ	a		$c+1$
$K+34$	ПЧ	a		$c+1+n$
$K+35$		1	1	
$K+36$	×		n	
$K+37$				1
$K+38$		1		1
$K+39$		1		
$K+40$	数目«1»			
$K+41$	数目«n»			

不同指令的作用完全不同，而且它們也只和解題过程的不同方面有关.

指令 $K+6$, $K+7$, $K+8$ 执行矩陣乘法的主要計算操作. 指令 $K+21$ 和 $K+23$ 直接变換結果列向量. 这些通常称为程

序的計算指令或算术算子．算术算子的指令就是按給定公式直接变换初始数量信息和中間数量信息的指令．

指令 $K+20$ 完全是另一种类型的，該指令并不执行任何計算操作，但它能确定下一步計算的方向．它检查算題所包含的一定的邏輯条件，即列的正分量使之按一公式变换，而非正分量按另一公式变换．程序中这种由多个可能的計算方向中选择下一步計算方向的部分称为邏輯算子．

指令 $K+11$，$K+16$ 和 $K+27$ 也是检查邏輯条件的，但具有另一种性質，因为程序中出現了反复执行的部分——循环．这些指令的工作就是要保証相应的循环重复执行一定的次数．

程序中的循环产生了一些不同类型的特殊算子．

除比較指令外，为了实現限制循环重复次数的邏輯条件，还須在每个循环中产生一个随循环不断重复而单調改变的变量．当該变量取給定終值时，循环的工作就应結束．如該变量不能由包含于循环中的算术算子所产生，則为求得該变量須引入一些特殊指令．当循环重复次数已知时，可以取循环已重复过的次数为該变量之值．把該变量和已給的循环重复总次数（即变量終值）相比較，我們就能保証循环重复給定的次数．在表 1 的程序中，循环 $K+5$ 至 $K+16$ 和 $K+7$ 至 $K+11$ 中系用此法保証所需的重复次数．計算循环重复次数的变量通常称为参数．由此定义可見，参数就是取整数值并在循环每重复一次时增加一个单位值的变量．存参数变化值的单元称为参数的計数器（单元 $\alpha+1$ 和 $\alpha+2$）．

使用計数器去保証循环重复一定的次数就使程序中产生一些特殊算子——送入参数初值的算子（指令 $K+4$ 和 $K+5$）和改变参数值的算子（指令 $K+10$ 和 $K+15$）．

程序中所有三个循环（$K+5$ 至 $K+16$，$K+7$ 至 $K+11$ 和 $K+19$ 至 $K+27$）皆具下列特点：循环中包含有随循环不断重复而改变自身形式的指令（指令 $K+6$，$K+7$，$K+8$，$K+19$ 和 $K+24$）．这些指令称为可变指令．通常可变指令系通过由其

地址部分中加或減某一常量的方法而改变，該常量称为改变地址常数（单元 $K+35$，$K+37$，$K+38$，$K+39$）. 可变指令在循环工作中系利用特殊的改变地址算子（指令 $K+9$，$K+13$，$K+14$，$K+25$，$K+26$）加以改变的. 可变指令在循环中既已改变，則要求在程序中引入能使可变指令恢复为初始形式的特殊指令. 恢复方法有两种：或由恢复算子加以实現，該算子由可变指令的某地址中減去該地址在循环重复工作中所得的全部增量（指令 $K+12$）；或由送入算子加以实現，該算子把先已存好的可变指令初始形式（由单元 $K+29$ 至 $K+33$）送入程序中相应的地方（指令 $K+1$，$K+2$，$K+3$，$K+17$，$K+18$）.

循环中某些指令的地址随循环不断重复而改变的事实可簡述如下：可变地址依賴于該循环的参数. 事实上，根据参数的流动值总能确定相应的可变地址值. 可变地址随循环的重复而改变一个常量的关系可以用可变地址对参数的綫性关系来表述. 可变地址随循环而改变的事实亦可称为依該循环的参数而改变地址.

参数的概念及可变地址对参数的依賴关系的概念在算子法中甚为重要，因为利用这些概念可精确表述編制含有循环的程序时所产生的問題.

如分析程序例子之前所已指出的，程序系由不同部分或不同算子所組成. 算术算子按算題的公式进行計算；邏輯算子实現确定計算进行方向的邏輯条件；改变地址算子、改变参数算子及恢复算子則和循环程序有关.

程序設計算子法的实質在于：由某一算題算法的初始形式过渡到解决該算題的完全編好的程序这一过程要分为两个阶段.

第一阶段　編程序的人分析算題，編出程序的邏輯图. 邏輯图指出解題程序由哪些上面已列举的算子組成及在程序中这些算子的相对排列关系是怎样的.

除写出邏輯图本身的符号外，还須編出邏輯图中各算子的信息，这些信息指出算子在程序中所要作的工作. 例如算术算子的信息就是該算子工作时所要計算的一組公式；按参数改变地址算

子的信息将指明那些可变地址在哪些算子中依参数而改变、每次改变多少；邏輯算子的信息就是該算子所应实現的邏輯条件；諸如此类等等.

第二阶段　編程序的人根据邏輯图中各算子的信息直接給各算子单独編出程序；分配存儲器并整編出符合于邏輯图的程序.

基托夫的"电子数字計算机"一书中对算子法也作了介紹[1].

把程序設計的整个过程分为两个阶段，同时也就很恰当地把程序設計工作划分开了.

第一阶段包含程序設計的全部"理智"工作，因編制邏輯图时，編程序的人所作的工作和被編算題的特点有关，这里的工作包括程序的一般組織和最佳程序方案的选择.

第二阶段主要包含程序設計的"机械"工作部分，包括个别算子的程序設計；存儲器的分配，工作单元的选择等.

程序設計第二阶段的工作比第一阶段較为定型，但第二阶段的工作却又最繁重，而且在第二阶段上編程序的人最易出錯.

因此，首先加以自动化并交給計算机去完成的正是程序設計的第二阶段，这一阶段的工作很容易加以形式的規定并写为改变信息的精确算法. 另一方面把程序設計第二阶段交給計算机去完成对节省人力和減少程序中的錯誤具有很大意义.

把程序設計第二阶段加以局部自动化的任务首先由 A. A. 李雅普諾夫教授于 1953 年提出. 1954 年巴格里諾夫斯卡婭 (Г. С. Багриновская) 在其毕业論文中提出了能自动为改变地址算子及恢复算子編制程序的程序.

把程序第二阶段加以全部自动化的思想，即編出編制程序的程序的思想属于刘毕姆斯基 (Э. З. Любимский) 和卡梅宁 (С. С.

---

1) 該书已有中譯本，1958，科学出版社出版. 另外科学出版社 58年出版的 Ю. Д. 斯梅格列夫斯基所著"程序設計基础"一书則完全是以算子法为基础而編写的——譯者註.

Камынин)[1]，他們于1954年10月在李雅普諾夫教授領导的程序設計討論班上报告了編制这种程序的基本原則，并介紹了某些程序設計算法．他們和苏联科学院数学研究所的一組工作人員于1955年3月編出了ПП的一种方案，該方案与本书所要介紹的方案在初始信息的特性和程序設計的算法上皆有所不同．

魯蒂斯豪泽尔（H. Rutishauser）[2]在程序自动化方面的早期工作也屬于这个方向，他在工作中提出了把程序設計第二阶段加以局部自动化的某些程序設計算法，但看来并未得到实际应用．

本书所介紹的ПП能把程序設計第二阶段加以全部自动化．算題的邏輯图及其算子的信息以編碼形式輸入計算机．ПП能編出算子的程序；分配存儲器；整編程序并輸出已編好的程序的信息．

1) Э. З. Любимский 和 С. С. Камынин: "程序自动化"，見"苏联数学机器和仪器制造的发展道路"会議文集，莫斯科，1956年3月12日—17日．

2) H. Rutishauser: Automatische Rechnenplanfertigung bei Programmgesteuerten Rechnenmaschinen, Basel, 1952.

第一部分

算題的初始信息

第一章

算子的几种类型

下面将确定在算題的邏輯图(ПП将利用这个邏輯图)中可以使用的几种初等算子,即算术算子;邏輯算子;循环及所謂的非标准算子.

§1. 算术算子

算术算子A是按一組公式进行一次計算的程序部分. 所謂公式即下列等式:

$$F(x_1, \cdots, x_n, c_1, \cdots, c_n) = y,$$

式中 y 是公式結果变量的符号, F 是对变量 $x_1, \cdots, x_n$ 和常数 $c_1, \cdots, c_n$ 进行的某些基本操作的任意組合. 以后变量和常量皆称为数量. 任一公式的左方称为表达式.

基本操作包括表2中的对数量进行的各种操作.

显然,該表中每一操作在 БЭСМ 上都能由一条或几条机器指令,或由存于 ДЗУ 中的标准程序来加以实現.

联結两个数量 x 和 y 的操作(如 $x+y, x:y$)称为二值位操作,而数量 x 和 y 称为它的分量. 变換单个数量 x 的操作称为对变元 x 的单值位操作. 单值位操作包括 Ex, $\mathrm{mod}\, x$, $ИП_n x$, $\leftarrow_n x$, $\downarrow x$, $\mathrm{ctg}\, x$, $\mathrm{tg}\, x$, $\ln x$, Выд x, $\exp x$, $\arcsin x$, $\mathrm{arctg}\, x$, $\sin x$, $\cos x$, $\sqrt{x}$ (在操作 $ИП_n x$ 和 $\leftarrow_n x$ 中, n 是参数而不是变量), 操作 x^2 和 x^3

表 2

編　号	操作名称	操作符号	附　註
1	加法……	+	
2	減法……	−	
3	乘法……	×	
4	除法……	:	
5	x 的平方……	x^2	
6	x 的立方……	x^3	完全对应于机器上相应的指令
7	x 的整数部分……	Ex	
8	x 的模……	mod x	
9	将 x 的阶改变 n……	ИП$_n$ x	
10	把 x 的数字部分移 n 位…	$\leftarrow_n x$	
11	x 的符号* ……	sign x	
12	分出 x 的阶……	$\downarrow x$	
13	印刷表达式 F 的計算結果	$F, \Rightarrow 0$	
14	x 的余切……	ctg x	
15	x 的正切……	tg x	
16	x 的自然对数……	ln x	
17	把 x 譯为十进数……	Выд x	由 ДЗУ 中对应的子程序来計算
18	e^x……	exp x	
19	x 的反正弦……	arcsin x	
20	x 的反正切……	arctg x	
21	x 的正弦……	sin x	
22	x 的余弦……	cos x	
23	x 的平方根……	$\sqrt{x}$	

* * 对应于指令 | ПЧ ± | <1> | <x> | <y> |，即 $y = \text{sign}\, x$.

构成专門的一組.

在数学問題的公式和 ПП 所用的算术算子的公式之間可能有些差别. 算題中的公式具有数学等式的形式，因而 $a + b = a$ 在变量取任意值时沒有意义. 公式中数量的所有数值及計算結果的数值須存在存儲单元中，所以必須在存儲单元和公式数量之間建立对应关系. 公式左端的某一数量可能和公式結果占用同一个单元(通常在計算各种迭推公式时就会有这种情形). 編程序的人可用同一字母表示左边的数量和結果. 为避免把这种公式和普通等

式相混淆，在算术算子的公式中不用等号 ≪ = ≫ 而用对应号 ≪⟹≫，这个符号表示对左边諸量进行一組操作的結果相当于公式右边的量．这里的对应关系应这样去理解：对左边諸量占用的存儲单元进行操作而得的結果将送入存儲右边量的存儲单元中．从这一角度看来，公式 $a + b \Longrightarrow a$ 具有明确的意义．

公式的任一写法中都須对任一操作唯一地指出其作用域，即指出該操作的分量或变元．給出公式中每一操作的作用域后才能确定完成操作的容許順序，即保証公式能得以正确計算的順序．

通常利用括弧指明公式中任一操作的作用域．但如只使用括弧，則公式的信息量就会急剧增大．而为了減少公式中的括弧，还須引入一些附加規則，这些規則在很多場合下根据公式各操作符号的相对位置，不用括弧就能确定操作的作用域及其完成順序．众所周知，无括弧时，应先算乘方；再算单变元的初等函数；其次算乘除；最后算加減．

因之，不用括弧可写出下例：

$$\sin \ln x^2 \times \sqrt{y} : z^2 + y \times z^2 - y^2 \times \operatorname{tg} \ln x \Longrightarrow t,$$

如用括弧則为：

$$(((((\sin(\ln(x^2))) \times (\sqrt{y})) : (z^2)) + (y \times (z^2))) - \\ - ((y^2) \times (\operatorname{tg}(\ln x)))) \Longrightarrow t.$$

这里介紹的 ПП 系采用前种写法而稍加精确化．公式中容許的操作有如下四类：

第一类——計算数的平方或立方的操作；

第二类——单值位操作；

第三类——乘除操作；

第四类——加減操作．

必須指出，在任一正确写出的公式中，开弧（閉弧）唯一对应一閉弧（开弧）．对括弧中任一操作符号总能找出包含該操作的一对最里边的括弧．如操作符号不包含于括弧中，則公式开端称为左自然边界；如包含于括弧中，則包含該操作最里边的一对括弧之开弧称为左自然边界．同样，右自然边界或为公式之对应号，或为包

含該操作符号的最里边一对括弧之閉弧. 例如,在下列公式中:

$$\sin[(a+b)\times(c-d)]-x^2 \Longrightarrow y$$

閉方括弧系乘法操作的右自然边界,而开的方括弧則为左自然边界. 減法操作的右和左自然边界相应地为对应号和公式开端.

对任一类操作都可建立唯一确定其作用域的規則. 在以下各例中,符号～表示各公式中操作的完成順序是等价的.

規則 1. 平方或立方操作的变元为該操作符号左边的整个表达式,直到第一个任一其它三类操作符号[1] 或該操作的左自然边界:

$$a^{2^3}+\sin b^2+c\times(b^2+a^{3^3})^2 \Longrightarrow y\sim$$

$$\sim((a^2)^3)+\sin(b^2)+c\times(((b^2)+((a^3)^3))^2) \Longrightarrow y.$$

規則 2. 单值位操作的变元为該操作符号右边的整个表达式,直到第一个任一第三和第四类操作符号或該操作的右自然边界[2]:

$$\sin b^2\times\cos\ln(a+b)^{2^3} \Longrightarrow y\sim$$

$$\sim(\sin(b^2))\times(\cos(\ln((a+b)^{2^3}))) \Longrightarrow y.$$

規則 3. 乘法或除法操作的左分量是該操作符号左方的整个表达式,直到第一个任一第四类的操作符号[3]或該操作的左自然边界. 乘法或除法操作的右分量是該操作符号右方的整个表达式,直到第一个任一第三和第四类操作符号或該操作的右自然边界:

$$a\times(b+c)\times d:e+\sin b^2\times c^3 \Longrightarrow y\sim$$

$$\sim(((a\times(b+c))\times d):e)+(\sin b^2)\times(c^3) \Longrightarrow y.$$

規則 4. 加法或減法操作的左分量是該操作符号左方的整个表达式,直到其左自然边界. 加法或減法操作的右分量是該操作符号右方的整个表达式,直到第一个任一第四类操作符号或該操作的右自然边界:

---

1) 自然,完全包含于該操作左方或右方括弧中的操作皆不計在內(原註).
按照这一規則可看出, $a^{2^3}\sim(a^2)^3=a^6$,而不是通常理解的 $a^{2^3}=a^8$——譯者註.

2) 同 1).

3) 同 1).

$$a+(b-c)+d-e+(a\times b-c)\times d+j\Longrightarrow y\sim$$
$$\sim((((((a+(b-c))+d)-e)+((a\times b)-c)\times d)+j)\Longrightarrow y.$$

这些规则在颇大程度上和数学中通用的运算順序的規則相同，并把括弧数目大大減少.

最后須指出公式中所用的三种符号. 符号 ≪(ₙ≫ 表示一串 n 个开括弧，符号 ≪)ₙ≫ 表示一串 n 个閉括弧. 符号 ≪,⟹≫ 用作对应符号，但表示該公式最后一个操作执行时对計算結果要封鎖規格化.

在邏輯图中，算术算子可写为它所包含的一串公式，或用加有标号的字母A来表示. 在后一种情况該算子的公式要单独写出并用同一字母A和同一标号表示.

§2. 邏輯算子

邏輯算子最一般的工作情况大致如下：

設給出变量 $x_1, x_2, \cdots, x_s$ 和程序算子 $F_1, F_2, \cdots, F_m$. 对应于每一組变量 $x_1, x_2, \cdots, x_s$ 的值应有 $F_1, F_2, \cdots, F_m$ 中的一个算子工作. 和变量 $x_1, x_2, \cdots, x_s$ 以及算子 $F_1, F_2, \cdots, F_m$ 有关的邏輯算子的工作就是要具体确定对任一組变量 $x_1, \cdots, x_s$ 的值应有那个算子 $F_1, \cdots, F_m$ 工作，并把控制轉給該算子. 这样，邏輯算子的工作可看作是計算某个取 m 个值的变量 $x_1, \cdots, x_s$ 的邏輯謂語(多值謂語).

未必可能一般地确定任一多值邏輯謂語的有效程序設計方法，因此在ΠΠ中仅考虑一特殊情况，即邏輯謂語仅依賴于一个变量 x 的情况，此后这个 x 就称为邏輯算子的变量.

先指出一点.

为指明控制轉移，在給出邏輯算子时邏輯图中任一可能接受控制轉移的算子都編有号碼. 算子号碼 N 用特殊符号 $|_{\overline{N}}$ 表示，在邏輯图中該符号放在它所表示的算子之前，即 $|_{\overline{N}}$ A 表示算子A的号碼为 N.

下面我們来确定邏輯算子的工作.