COBOL
程序设计基础

魏际昌　王立福　编
刘　愈　应　煊

辽宁科学技术出版社
一九八三年·沈阳

内 容 摘 要

本书以国际标准化组织推荐文本——《程序设计语言 COBOL》为基准，系统介绍了 COBOL 的基本概念，数据的组织形式，源程序的标识、设备、数据及过程四个部分，源程序的设计步骤和设计方法；分别讨论了排序、报表打印、程序分段、程序库各功能模块，作业控制语言及如何组织完整的 COBOL 作业。

本书可供大专院校计算机软件专业、经济管理专业的师生及广大经济管理技术人员参考使用。

COBOL

程序设计基础

魏际畏 王立福 刘 愈 应 煊 编

辽宁科学技术出版社出版（沈阳市南京街6段1里2号）
辽宁省新华书店发行 沈阳新华印刷厂印刷

开本：850×1168 1/32 印张：13 插页：2 字数：300,000
1983年7月第1版 1983年7月第1次印刷

责任编辑：李殿华 责任校对：李秀芝
封面设计：吴凤旗

印数：1—6,300
统一书号：15288·2 定价：1.50 元

目　录

第一章　COBOL的基本概念

在电子计算机上进行数据处理，必须首先编写程序。编写程序的过程称为程序设计。编写程序所使用的语言称为程序设计语言。COBOL 是许多高级程序设计语言的一种。COBOL 一词是 COmmon Business Oriented Language 的缩写，意思是：面向商业的通用语言。COBOL 最适用于各类企业、部门对大量的数据进行加工处理，如报表汇总、库房管理、工资管理、银行业务、图书资料和情报检索等，都可以通过COBOL 程序来解决。

一、COBOL 的发展概况

二十世纪五十年代末期，适用于科学和工程技术计算的高级程序设计语言，如ALGOL、FORTRAN 等出现以后，企业界强烈要求能设计一种适合于企业数据处理的高级程序设计语言。因此，1959年 5 月，在美国五角大楼召开了由企业界、政府和计算机制造商代表参加的专门会议，经过充分酝酿和广泛讨论之后，决定着手设计一套适合于企业数据处理需要的高级程序设计语言。1960年 4 月，美国发表了第一个 COBOL 说明书；1961年 6 月，对这个说明书作了必要的修改之后，正式发表，这就是第一个 COBOL 文本。此后，许多计算机制造商都在自己生产的计算机上配置了 COBOL 编译系统。1965年12月，发表了另一个 COBOL 文本，该文本在前文本的基

础上有新的发展。1968年8月，由美国标准化研究所批准，把1965年文本定为美国标准COBOL文本，称为ANSI COBOL。ANSI 是 American National Standards Institute 的缩写，意为：美国国家标准学会。1970年7月，国际标准化组织所属的一个技术委员会，提出了一个 R1989 号文本——《程序设计语言 COBOL》，简称 ISO COBOL（ISO是 International Standards Organization 的缩写，意为：国际标准化组织），经在全体会员国中巡回审阅，除在编译方面有少许修改外，被美、英、法、苏、日等二十一个会员国承认，目前已成为国际标准文本了。本书主要介绍 ISO COBOL以及如何使用它编写程序。

二、COBOL 的基本特征

目前各国所生产的计算机，无论是大型、中型、小型或微处理机，一般都配有COBOL系统。COBOL 在企业和经济部门，在图书管理、地质勘探以及科学实验中都得到广泛的应用。所以如此，是由 COBOL 的基本特征决定的。

COBOL 的可分割性是它的第一个特征。标准的COBOL被人为地划分成一个核心和七个具有独立性的功能模块，其中每一功能模块又按其功能范围的大小，分成二至三个等级，等级越高功能越大。功能模块及其等级的划分如第三页表所示。

核心的主要功能是为在计算机内部处理数据提供必要的语言能力；表处理的主要功能是提供定义一个表，并且按表的相对位置来存取一个数据项的语言能力；顺序存取用于提供一种按记录写入文件的顺序来存取一个文件记录的语言能力；随机存取用于提供按程序员所给的控制字来随机存取一个文件记录的语言能力；排序用于提供按照用户要求对一个文件中的记录

核心	功能模块						
	表处理	顺序存取	随机存取	排序	报表打印	程序分段	程序库
2 级	3 级	2 级	2 级	2 级	2 级	2 级	2 级
1 级	2 级	1 级	1 级	1 级	1 级	1 级	1 级
	1 级		0 级	0 级	0 级	0 级	0 级

重新排序的语言能力；报表打印用于提供通过对一个报表的描述就能自动产生这个报表的语言能力；程序分段用于提供规定目标程序覆盖要求的语言能力；程序库用于提供如何把一个程序或程序段存入程序库和如何从程序库中复制已经存入的程序或程序段的语言能力。

这种划分使得编译系统的设计者，可以根据自己机器硬件结构的特点和处理问题的范围等，从中选取几个部分以及每个部分的等级，来组成自己的 COBOL 系统，这比重新设计一个系统要省事得多。一个最小的 COBOL 系统应由核心及各功能模块的最低级组成；但由于存在着 0 级（空级），所以实际上仅由核心、表处理及顺序存取的一级组成。一个最大的COBOL系统由核心及各功能模块的最高级组成。例如FELIX机的 COBOL 系统接近于最高级，但对某些较繁的功能（如报表打印等）作了删节；而 Z—80 机的 COBOL 系统则接近于最低级。

COBOL 的第二个特征是成文自明。在 COBOL 中使用了大量的英语词汇和句型，很接近于英语，很多语句都使用英语的自然格式，因此写出的程序一目了然，不用另加说明。例如，要从应得工资额中减去应扣除部分而得到实际工资额，写成 COBOL 语句就是：

SUBTRACT DEDUCTION FROM GROSS—PAY
GIVING NET—PAY.

数据处理和企业的其他业务一样，是不断发展变化的，程序的修改是经常的，成文自明有利于使修改程序的工作由不同的人在不同的时间和场合来进行。

COBOL 的第三个特征是具有通用性。由于绝大多数计算机都配有 COBOL 系统，而且所有的 COBOL 系统都以标准文本为基准，所以同一个 COBOL 程序只要稍加修改就可以在各种型号的机器上编译和运行，这给程序设计工作者带来了极大的方便。

COBOL 的第四个特征是能对大批量的数据进行极严密的组织和描述。通常都是把数据组织成记录、文件，预先存放在外部介质上，使用时再分批进行读/写和处理。

当然，COBOL 也有缺点，主要是过程部分的语句格式复杂，计算能力较差，不适用于数值计算。

三、COBOL 字符集

COBOL字符是组成COBOL 源程序的最小单位，ISO COBOL 文本规定以下51个字符为 COBOL 字符：

①字母字符，即A, B, ……, Z二十六个英文字母；

②数字字符，即0，1，2，……，9十个阿拉伯数字；

③其他专用字符，包括：

+	加号（或正号）
-	减号（或连字符、负号）
*	乘号（或星号）
/	除号
=	等号

$	货币符号
,	逗号（或十进小数点）
;	分号
.	句号（或十进小数点）
'	单引号
(	左圆括号
)	右圆括号
>	大于符号
<	小于符号
	空格（什么也不写）

一般地，除以上51个字符外，在具体的机器上还可以规定一些补充字符供使用。但在语法格式中出现的方括号、花括号等都不属于 COBOL 字符。

四、COBOL 字

一般而言，一个COBOL 字是满足以下三个条件的由字母字符、数字字符和连字符组成的字符组合：

①字符组合最多不能超过30个字符，两端以空格定界，中间不得出现空格；

②字符组合中至少要包含一个字母；

③连字符不能出现在字符组合的首或尾。

COBOL 字可分以下四类：

1. 保留字

凡是对编译系统具有特定语法意义的 COBOL 字，都称为保留字。保留字一律要求大写。保留字包括：

（1）关键字

凡是出现在语法格式中并且必须写的保留字，称为关键

字，关键字的下面都画有一条横线。在选用某一语法格式时，其中的关键字必须正确地书写，不能省略。关键字包括：

过程动词：如ADD、READ、WRITE,等等；

用于限定的连接词：如 OF 和 IN；

用于在逻辑运算中组成复合条件的连接词：如AND、OR、NOT，等等；

其他具有特殊功能的字：如 SECTION、DATA、PROCEDURE,等等。

（2）选择字

虽然出现在语法格式中但选用时可写可不写的保留字，称为选择字。写上它能增加程序的完整性和易读性，不写也不影响编译系统的正常工作。

（3）专用寄存器

专用寄存器是由编译系统开辟的，存放专用信息的存贮字段，其长度和类型都是预先定义了的，用户可以不必定义而直接使用。例如 LINE—COUNTER （行计数器），PAGE—COUNTER（页计数器）等。

（4）象征常字

象征常字是具有固定名称且具有特定含意的常字，程序员无须定义就可直接引用。象征常字的形式和含意如下：

ZERO
ZEROS } 表示一个或多个零

SPACE
SPACES } 表示一个或多个空格

QUOTE
QUOTES } 表示一个或多个引号

HIGH—VALUE } 表示一个或多个在计算机对照顺序中
HIGH—VALUES } 其值最高的字符

LOW—VALUE } 表示一个或多个在计算机对照顺序中
LOW—VALUES } 其值最低的字符

ALL 常字：表示一个或多个由非数值常字或除它本身外的象征常字组成的字符串

编译系统在编译时，将用相应的常字值来代替象征常字。象征常字的长度按以下原则确定：

①如果象征常字不与其他数据项发生联系，则除了ALL不能使用外，其他象征常字的长度皆为1；

②如果象征常字同其他数据项发生联系，则除了QUOTE(S)仅在最左字符位上产生一个单引号外，其他的长度由相关数据项的长度决定。例如，若数据项A的长度为5，要把象征常字 ZEROS 传送给A，则这时 ZEROS的长度为5。

2. 用户字

凡是根据需要由用户自己定义和拼写的字，称为用户字。用户字包括：

（1）数据名

在程序中出现的所有非常数数据项，为了在过程部分索引，必须起个名字，这个名字就是数据名。数据名的组成要遵照 COBOL 字的组成规则，例如：

HOURS（小时）

ENDING—INVENTORY(期末库存)

SALES—TAX—TOTAL（销售税合计）

AS27.5

315X6

等等都是符合要求的数据名。而下列字就不能作为数据名：

UNIT COST（单价）　　中间出现了空格

—MIN—VALUE（最小值）连字符出现在字符组合之首

31576　　没有一个字母

数据名相当于数学中常说的变量，它在程序中出现仅表示存贮器内某一存贮字段，并不指明具体的值，在不同时刻可能有不同的值。

（2）条件名

若一个数据项只能根据不同条件取几个预定的值，则该数据项称为条件变量。用来表示条件变量当前取值情况的名字，称为条件名。例如数据项 ABC 是个条件变量，它仅能取预定的三个值，对每种情况可赋予一个名字。如下表所示：

ABC 取值	条 件 名
18574	TJM—1
92485	TJM—2
39635	TJM—3

这里 TJM—1 就表示条件：ABC = 18574；TJM—2 表示条件：ABC = 92485；TJM—3表示条件：ABC = 39635。这样规定以后，如果我们希望当 ABC = 92485时，控制转移到P过程，就可以写出语句：

IF TJM—2 GO TO P.

而不必写成：

IF ABC IS EQUAL TO 92485 GO TO P.

显然，使用条件名可以使程序更加简练。

（3）过程名

过程部分的节名、段名都称为过程名。过程名的组合除遵

照 COBOL 字的组成规则外，还可以完全由数字组成。如0012、018等，都可以当作过程名。

（4）记忆名

在程序的设备部分，程序员需给程序中使用的每个数据文件分配相应的外部设备。每件外部设备，系统都给它规定有特定的名字。但有时程序员想用简单的符号代替设备名，这样的简单符号称为记忆名。如果用记忆名代替了设备名，则需在设备部分加以说明，以后在过程部分就可以用记忆名来代替设备名使用。例如：系统给控制台规定的名字是 CONSOLE，如果程序员想用K表示控制台，就必须在设备部分写上“CONSOLE IS K”，K就是个记忆名。

3．系统名

系统名是由计算机制造商定义的用来标识计算机硬件设备的名字，不同的计算机上将有不同的规定。例如，可以用：

CONSOLE　表示操作控制台

SYSIN　　表示读卡机

SYSOUT　　表示宽行打印机

等等。系统名是预先定义了的，程序员可以直接使用。

4．常数

在数据处理过程中其值始终保持不变的数据项称为常数，前面介绍过的象征常字是一种具有固定名称的常数，由用户定义的常数不用再命名，它的名称与其值是完全一致的。常数包括：

（1）数值常数

由 0 ～ 9 这十个数字符号、正负号和十进小数点组成的常数称为数值常数，其最大长度为18位数字。数值常数的形成规则如下：

①必须至少有一个数字；

②不能有两个以上的数符出现，数符只能出现在最左边，如果一个数值常数不带数符，则表示该数为正；

③只能有一个小数点，小数点除不能出现在最右边位置外，可以出现在任何位置上，若没有小数点则表示该数为整数。例如：

1587

-11.54

+235.84

0.33333

都是合乎要求的数值常数。而下列写法就不正确：

1587.　　小数点出现在最右边

+12-58　　出现了两个数符

11.54-　　数符出现在右边

0.333333333333333333　　超过了18位数字

（2）非数值常数

凡是用两个单引号括起来的任意可允许的字符组合，都是非数值常数。单引号仅用于定界，不算在字符组合之中。例如：

'THIS IS A COMPUTER'

'NOVEMBER 15,1980'

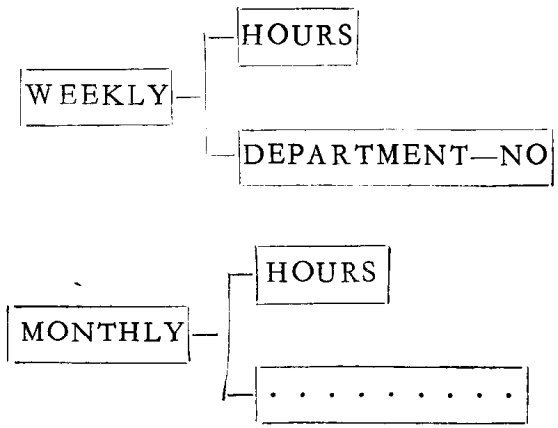
'77.50'

等都是非数值常数。非数值常数的最大长度为120个字符，单引号不计算在长度内，但中间的空格是计算在长度内的。注意：'77.50'是非数值常数，它不等于77.50。

五、限定语和标识符

COBOL 程序中所使用的每个数据名，在内存中都有一个相应的存贮字段，当索引某一数据名时，就应唯一地指向它所对应的存贮字段。例如编译系统给数据项 NAME 分配的内存字段为1001～1018这一空间，当我们要把信息“WANG—MIN”传送到 NAME 时，就是传送到内存 1001～1018 这个字段中。

然而有时会遇到两个记录或两个组合项包含相同数据名的情况，例如有两个组合项，它们的组成如下：



两个数据项都包含有 HOURS，而两个 HOURS 却是针对不同的变量的，它们在内存中各占一个存贮字段。如果简单地引用 HOURS，编译系统就分不清究竟指的是哪个字段。为使指向唯一，就必须写成：

HOURS OF WEEKLY

HOURS OF MONTHLY

或者写成：