

842.153

可编程序袖珍计算器

●何德书 李凤亭 译 ●王基奎 校

電子工業出版社



可编程序袖珍计算器

[美] H·马利士 S·考肯 著

何德书 李风亭 译

—— 王基奎 校

电子工业出版社

内 容 简 介

可编程序袖珍计算器具有轻巧、体积小、可携带,有编程、记忆、打印等功能,且价格便宜的特点。本书详细介绍了 HP 公司各系列的可编程序袖珍计算器的特性、功能及编程技术,例举了数学、物理、财会统计等方面的应用算例。书中介绍的基本原理和使用方法,对其它类型的袖珍计算器也适用。本书深入浅出,对无任何编程知识的读者均可阅读,是一本很好的用户指南。

可编程序袖珍计算器

[美]H·马利士 S·考肯 著

何德书 李风亭 译

王基奎 校

电子工业出版社出版 (北京海淀区万寿路)

新华书店北京发行所发行 各地新华书店经售

北京市通县建新印刷厂印刷

开本: 787×1092毫米1/32 印张: 10.125 字数: 228千字

1987年3月第1版 1987年3月第1次印刷

印数: 1—4400册 定价: 1.95元

统一书号: 15290.376

译者序

在八十年代的今天，电子计算机已渗透到国民经济和日常生活的各个领域，并在四化建设中起到日益重要的作用。如何推广电子计算机的应用，以及培养大批计算机的研究和应用人才，是和四化进程密切相关的大事。

HP 公司的可编程序袖珍计算器在世界上有着较为广泛的市场。它的特点是：既像普通计算器那样轻便、体积小、可随身携带，又和大型计算机一样具有可编程序、记忆程序和数据，甚至打印清单的能力。科学研究和工程技术人员可以在实验室、厂房、工程现场、野外等用它随时进行计算、设计和数据处理；可以作为微处理器进行过程控制和数据采集工作。另外，因为它价格便宜，一般家庭和个人都买得起，所以若作为成人或儿童的计算机教学工具是最合适不过了。

本书前四章由李凤亭同志翻译，后七章由何德书同志翻译，王基奎同志校阅了全部译稿。由于水平所限，难免有错误之处，恳请读者指正。

译者 1985·6

序 言

一场真正的计算器革命发生于1971年，当时单在美国销售的袖珍计算器就有成百万台，计算尺成了过时的东西，而让位于这个电子学的奇迹。其实那时计算器所能做的事情不外乎是加减乘除，然而它的速度和准确度却给人们留下了深刻的印象。

在袖珍计算器的高潮之后，接踵而来的是另一场革命——可程序袖珍计算器。这场革命可以溯源于1973年12月。那时美国计算机和计算器制造行业的大厂家Hewlett-Packard公司将它的令人惊叹的可编程计算器——HP-65，推向世界。HP-65除了有可编程的特性之外，还可提供几乎是数学和科学计算用到的全部函数。它之所以会对计算器世界有如此的影响，有两方面的原因：一是它作为世界上出现的第一台可编程的袖珍计算器，在科技发展史上留下了光荣的一页；其次，用户可以将程序记录在涂有金属-氧化物的塑料薄片上。虽然HP-65开辟了一个全新的市场，但是终因它的价格过于昂贵（800美元）而受到限制。这个价格一直维持到1976年它的换代产品HP-67问世时为止。

为了占领更多的市场，1974年底，HP公司又推出了另一型号的可编程计算器HP-55。虽然HP-55在性能上不如HP-65，但是它能编程，并且售价只有HP-65的一半。

势在必行，其它的大企业，例如National Semiconductor（国家半导体公司）也已经跻身于可程序袖珍计算器

的领域。该企业于1975年1月推出了四种型号的可编程序袖珍计算器,它们是:4515 “Programmable Mathematician (可编程数学家)”, 4524 “Programmable Scientist (可编程科学家)”, 1625 “Programmable Financier (可编程金融家)”, 6035 Programmable Statistician (可编程统计家)”。当时这四种型号产品虽然技术水平和工艺水平比 HP 公司和得克萨斯仪器公司的产品来要差,但是它们的价格都很便宜,这就使得它们对于教育、科学和商业中的专业人员和学生有着极大的吸引力。

1975年6月,一个英国的计算机厂家 Sinclair Radionics (辛克莱射电电子学公司), 生产出19个键、24步的科学型可编程序计算器。

1975年7月, HP 公司又以它的重量不过6盎司, 售价为195美元的可编程袖珍计算器 HP-25, 使计算器界为之震动。

1976年7月, HP公司又宣布了HP-65的换代产品。这个新的可编程袖珍计算器叫 HP-67有224步,配有磁卡。与此同时, HP公司还宣布它的声望很高的产品 HP-25有了重大改进,这就是HP-25C, 它作为第一台具有“持续记忆”功能的可编程计算机而标新立异。

1977年12月, HP 公司生产出了 HP-19C 和 HP-29C。HP-19C 是第一批带有内部热感打印机的可编程的袖珍计算机商品, 零售价为325美元。而不带打印机的同类产品 HP-29C 售价为185美元。

这时, HP公司的方针作了某些改变,试图占领低档的袖珍计算器市场。该公司于1984年4月推出了E系列计算器。HP-33E 和 HP-38E 就是这一系列的产品。它们的零售价分

别为100美元和120美元。

本书打算详细介绍这些可编程的袖珍计算器，讨论它们的算术运算，特殊功能，和编程技术。对于读者不要求具备任何关于编程的预备知识。

今天我们面临品种繁多的可编程和不可编程计算器的广阔的市场，要想做出明智的选择是件不容易的事情，因为我们可能在各式各样的计算机特点面前手足无措。这仅仅是问题的一部分——从消费者的立场来看是如此。那么从推销员的角度来看，也存在他的问题。作为一个推销员，应该熟悉各种型号的计算器，而这些计算器在许多功能和处理方法上又各有差异。如何想象他能搞清楚这么多机型，特别是可编程计算器的优缺点呢？他们中间的许多人可能根本没有受过关于编程的基本训练。让他们对于这门学科的专门问题作出精辟的回答是不可能的。

在今天技术飞速发展的时代，许多拥有计算器或计算机的部门和单位都有电话可以直接和厂家联系，他们可以随时提出自己的技术问题。

本书将会对消费者或推销员有所帮助。每一种机器的每一个程序都给出一个操作过程，告诉读者如何正确地输入程序以及正确地起动计算器。用这种方法使推销员或者消费者感到犹如亲自在键入每个程序，看到计算器在工作，计算结果显示在眼前一样，而不必为各种各样的编程原理、逻辑，或者专门的计算器运算而烦恼。

H·马利士

S·考肯

目 录

第一章 程序设计技巧.....	1
一些基本的程序设计概念.....	2
用流程图表示一个数学问题.....	5
调试和编辑程序.....	7
第二章 可编程序袖珍计算器概述.....	9
可编程序袖珍计算器的逻辑.....	9
计算器显示.....	12
可编程序袖珍计算器的性能.....	14
第三章 普及型可编程序袖珍计算器.....	21
诺瓦可编程序计算器.....	21
辛克莱可编程序科学计算器.....	35
第四章 HEWLETT-PACKARD	48
可编程序袖珍 计 算 器.....	48
各种型号Hewlett-Packard计算器的共同特性.....	48
第五章 HP-25型和HP-25C型计算器.....	66
HP-25型计算器.....	66
HP-25C型计算器.....	67
HP-25和HP-25C的用户指南.....	70
HP-25和HP-25C的编程.....	72
在HP-25和HP-25C上调试和编辑程序.....	106
第六章 HP-55型可编程序科学袖珍计算器.....	108
HP-55用户指南.....	110
数字时钟.....	114
HP-55的编程.....	119
在HP-55上调试和编辑程序.....	135

第七章	HP-65型可编程序计算器	136
	HP-65用户指南	136
	HP-65的编程	148
	在HP-65上调试和编辑程序	190
第八章	HP-67型可编程序计算器	192
	HP-67用户指南	192
	HP-67的程序设计	204
	在HP-67上调试和编辑程序	252
第九章	HP-19C和HP-29C型计算器	254
	HP-19C/HP-29C型计算器的程序设计	255
	HP-19C和HP-29C上调试和编辑程序	287
第十章	HP-33E型计算器	290
	HP-33E型计算器的程序设计	291
	HP-33E上调试和编辑程序	311
	个人程序设计员俱乐部(PPC)——前身是HP-65用户俱乐部	312
第十一章	袖珍式可编程序计算器	
	在科学、工业和教育方面的作用	314

第一章 程序设计技巧

计算机程序设计不仅是一个需要大量个人经验的高报酬专业，而且也是最令人满意和最有前途的专业之一。这专业本身就其开始来说还只不过是小孩，但是自从五十年代计算机出现以来一直突飞猛进地发展着。目前世界各大学就这个学科开设了大量的课程，而且在中学开设诸如FORTRAN、ALGOL、PL/I、BASIC、COBOL等计算机语言方面的课程，也正变得越来越普遍。

为什么如此需要计算机呢？原因是有史以来我们第一次能以电的速度解决问题。这些电子奇迹本身并不具备固有的智慧。这种智慧是由我们为得到解答而编写的适当的指令所决定的。这样一个指令序列叫做程序，它是人为安排的，恰象剧院中演戏，按照节目单一幕一场的表演那样。

非得是个天才方可以成为一个程序员吗？当然不是。任何一个稍有智力和逻辑感的人都可以编程序，不需什么预先的训练。当然，逻辑感越强的人越容易些。看来那些擅长于解谜和游戏的人将会成为优秀的程序员。而且这些人几乎是带着一种爱好转向程序设计的。象生活中许多其它事情一样，程序设计也是随着成功而改进的。一个人一旦写了一个程序，不管它是怎样简单，都好象提供了一种力量推动他走向新的高度。对于某些人来说，程序设计成了他的一种嗜好，它既是人们生活中的乐趣又没有任何消极作用。

一些基本的程序设计概念

人们常面临用计算器计算一个复杂的或许又是冗长的数学表达式的问题。于是他决定一个敲键步骤的序列，并一步步地执行，直到获得并显示最终解答。如果需要计算同一个数学表达式，他实际上必须重复求解第一个解答时的步骤序列。如果这个指令序列必须重复数百次甚至数千次，这将成为一个可怕的杂务，当然人们希望有一种方法，使计算器可以“记忆”在第一次计算该表达式所用的那个指令序列，

而允许用户对每一个新的操作数据集重新启动这个存于计算器“存贮器”中的指令序列。

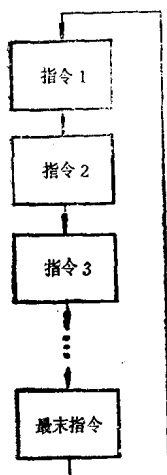


图 1-1

现代的可编程计算器的基本特征，正是提供了这个能力：存贮和执行存于计算器存贮器中的一个叫做**程序**的指令序列，并且可使用指令处理任意多批数据。

这个序列操作的一个重要变化是，当程序执行到一个特殊的部位时，可以自动地重复这组指令。这就提供了程序设计的一个基本性能，即叫做**循环**的功能。改变一个程序的正常顺序的功能是通过叫做**无条件转移**的方式完成的，如图 1-1 所示。

当要在遇到一个特定的条件时使程序转移到别的点时，另一个重要的因素要加到这个序列操作中(图1-1)，这叫做**条**

件转移，它提供了在程序里做较复杂的判断的方法。

作为这两种不同情形的一个类比，可以想象一个面包师的学徒从烘炉里取出糕点的工作。取出后，他必须冷却糕点，并放在盒子里，然后再存入冰箱。他将简单地重复这些步骤，如图 1-2 所示。

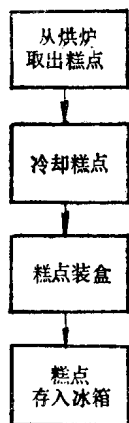


图 1-2

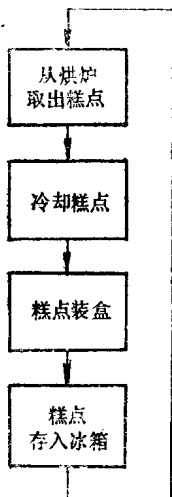


图 1-3

自然面包店生产的糕点是很多的，这个学徒的工作就是对每一块糕点重复上述的操作步骤。

这可以用图来表示，如图 1-3 所示。在这里我们有一个序列首尾相接的循环概念。

现在假定这学徒在下午 5 点钟结束他一天的工作，并且不想加班。那么在每存入冰箱一块糕点后，他应该问问自己是否已到了下午 5 点钟。如果是，他就穿上他的外套下班。如果不是，他就去拿出下一块糕点，再完成一遍上述步骤。

应该注意的是在这里这个判断是每次执行过程中都进行的，即是否到该停的时候了？这可以用图表示，如图 1-4 所示。

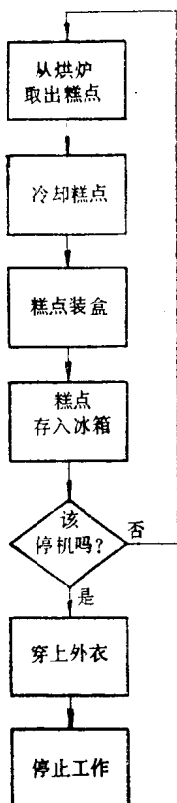


图1-4

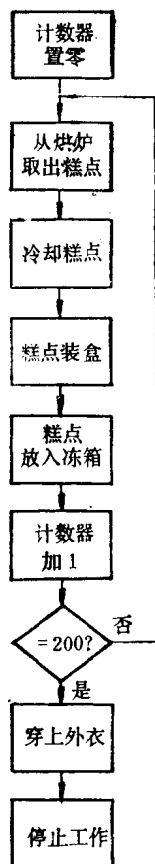


图1-5

这种图在程序设计中叫**流程图**，是表明程序控制流的一个非常有用的方法。

现在让我们改善我们的面包师的工作，以便说明程序

设计的一些更重要的原理。按照常规我们假定这学徒已做完200块糕点算完成他的一个工作日，而不管他用了多少时间。如果他在下午3点钟完成，则他可以在下午3点钟就离开，但是，如果他在下午7点钟才完成，那么他就要工作到下午7点钟而不多得报酬。

这个新的情况意味着在整个工作日都必须对加工的糕点块计数。当然在这一天的开始，这计数是零。一旦一块糕点存入冰箱，他就随即在这数上加1。由于我们可以假定他渴望尽可能快地停止工作，每次他加一块糕点，就检查一次数字，看是否已经达到200。如果已经达到，他穿上他的上衣停止工作。如果数字还未达到200，他就留下来加工下一块糕点。图1-5的流程图代表了這個新的情况。

在程序设计中使用计数器的概念是头等重要的。这里，计数器只是一个告诉我们什么时候退出循环的手段。换句话说，要根据计数器的值进行转折点的判断。

注意，在图1-5里提了一个问题：“计数等于200吗？”（通常在菱形的“判断”框里写这样的问题）。如果回答是“是”，我们不返回到最开头（在那里计数器被置零），而是到随后的一步。

用流程图表示一个数学问题

现在让我们来看一个必须做一个基本判断的简单问题。我们将考察一个整数序列并决定其中有多少是偶数有多少是奇数。

根据定义，如果一个整数被2除而没有余数则它是偶数，如果有余数，则认为该数是奇数。在这个具体问题里有

两个计数器要用到：一个保存偶数的计数而另一个计奇数。

图 1-6 所示的流程图的输入数据，是自身相互独立的数。每个数却被考察以决定它是奇数还是偶数，而给相应的计数器加 1。一旦所有的数都被考察过，奇数计数器的计数和偶数计数器的计数就被显示，代表这个问题的输出结果。

除极少例外，输入和输出的概念对所有程序都是共同的。

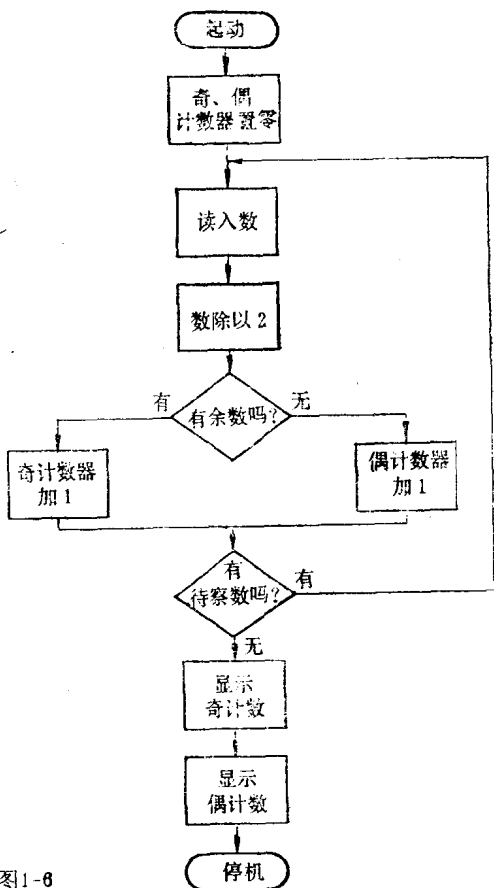


图 1-6

调试和编辑程序

在考察诸如本书所包括的那些程序时，不应错误地认为它们是第一次就写成这个样子的。很少有程序能第一次就运行的。况且，即使一个程序好象第一次就能运行，但往往有其偶然性，其偶然性就在于它不能对所有可能的数据都运行。编程过程可能是而且经常是一个完全无效的过程。人们在使用一个具体机器时需要先学习并且要对解决一个给定问题的步骤有一个清楚的理解，以求达到所要求的结果。

据我们所知，没有一个程序员即使经过努力，仔细地写出的程序，第一次上机就能运行；而总要经过许多挫折以后，才能使程序正确运行。错误或者是由编写程序造成的，即程序有逻辑错误，或者是由输入指令造成的，这似乎是程序设计的一个特点。这些错误习惯上称为故障(bugs)，而查找和排除这些故障的过程称为调试(debugging)。一旦找到故障，则校正和修改该程序的过程通常称为编辑(editing)。事实上，调试和编辑一个程序可能比写一个程序花的时间更长。

一个程序一经写出，修改应该说是必须的，这差不多总是事实。但了解下面这一点可以多少减轻一点程序员的忧虑，即并不总是需要重新从头键入程序，因为原来的程序有很多还是可以利用的。

在确信程序处于完善的工作条件之前，如可能的话，应该用试验数据检查它的输出并与已知的结果相比较。如果这些结果之间有矛盾则一定是有什么地方出错了，就不能不怀疑此程序。失败可能是由于程序的键入有错误，或者是程序

有逻辑错误。

在前一种情况下，仔细比较键入的程序和原先手写的程序，将会发现不一致的地方。后一种情况是怀疑有逻辑错误，建议用下面的方法：

1. 检查程序流程图，以确保各逻辑框是合乎逻辑顺序的。

2. 比较程序本身和程序流程图的逻辑是否一致。

3. 要确保指令的功能与原设计要求一致。这意味着要将纸上的程序从头到尾一步一步地走一遍，时刻注意程序用到的每个寄存器的内容。

4. 利用所使用的计算器上的各种有效的调试手段。这可能包括单步键，它允许用户每一次处理程序中的一个指令；暂停键，它暂时停止程序执行，使中间结果可以看到；或者利用可能提供的任何列表特性。