中华学习机

紫金Ⅱ系列

APPLEⅡ

朱国江
詹丰兴 编著

数据处理

高等出版社

前　言

近年来，有关计算机的教材和译作出版甚多，受到广大读者的欢迎，对我国计算机的普及推广，起了很好的推动作用。

但是，目前关于青少年学习、应用计算机方面的书并不多见。为此，我们决定以国家教育委员会指定推广的国产优秀机型——CEC-I 中华学习机为选型机，编写一套中华学习机系列丛书，以满足广大青少年及不同层次的读者需求。这套系列书暂定六册，它们是：《中华学习机编程技巧》、《中华学习机数据处理》、《中华学习机汉字软件》、《中华学习机图形管理》、《中华学习机机器语言》、《中华学习机程序设计》。

本书是中华学习机系列书的第二册，系《中华学习机编程技巧》（气象出版社，1988年2月出版）的姊妹篇，主要介绍 BASIC 语言和 6502 机器语言在数据处理方面的应用和技巧，内容广泛，取材新颖，深入浅出，循序渐进。强调普及和提高结合，实用和技巧兼顾。书中提供了一定数量的工具软件和大量的实用程序，也有作者多年实践经验的总结，其中不少内容是现行同类书籍和资料中所没有的，具有一定的学术水平和较高的应用价值，相信会给广大读者提供切实而有益的帮助。

本书资料主要来自作者多年编程和教学实践，同时还吸收了国内外的先进经验，也选用了少量书籍和杂志上发表的优秀文章。书中阐述的原理、方法、思路和技巧，对其它程序设计和微机应用都有参考价值，而全部程序可以直接付诸

使用和稍作修改后引用。

本书共分八章，其中二，5,(4),(5)；三，4，5,6,12；以及第六章节由詹丰兴编写，其余一至八章各节均由朱国江编写，并由朱国江统一审校。

南京大学邹进上教授，对系列书的编写给予鼓励和支持，并审阅了本书的初稿，在此特表谢意。

编写这种适用性强、应用面广的综合书籍，尚属尝试，书中难免有不妥之处，恳切希望广大读者不各赐教。

编　者

1988年10月25日

目　录

一、软件加密技巧

1. 问题的提出

随着计算机在我国的日益普及和广泛使用，特别是在经济、铁路、电力、银行、公安、气象、民航、军事等重要部门和全国性信息系统中的应用，对软件和程序采取加密措施，已是程序设计中不可缺少的一个环节。对初学计算机语言并有一定编程能力的人来说，为使自己的程序有一定的权威，了解和掌握简单的保密措施及解密方法，也是必要的。

下面介绍的一些保密方法比较简单，但有一定的实用价值，如果同时采取几种加密措施，可能收到较为满意的保密效果。当然，限于篇幅，我们不可能也没有必要讲述高难度的保密方法，因为，对于具有专门知识而又存心解密的行家来说，总有解开的办法。

2. 从救回BASIC 程序谈起

假设有一个程序PRO-1：

PRO—1

```
5  REM PRO-1
10 A = 10
20 B = 5
30 C = 4
40  PRINT A + B - C
50  PRINT A - B + C
60  END
```

这是一个十分简单的BASIC程序，无需多加解释。我们想说明的是，它在内存程序区中是如何存放的。

用监控命令，可以看到程序区的内容如下，见PRO-2：

PRO-2

```
*800.841

0800- 00 0B 08 05 41 00 84 20
0808- 00 00 00 42 00 83 20 00
0810- 00 00 43 00 83 00 00 00
0818- 00 D0 35 00 24 08 1E 00
0820- 43 D0 34 00 2F 08 28 00
0828- BA 41 C8 42 C9 43 00 3A
0830- 08 32 00 BA 41 C9 42 C8
0838- 43 00 40 08 3C 00 80 00
0840- 00 00
```

有人会提出疑问，你如何知道上述BASIC程序，在内存中是从$0800开始，到$0841结束的？$0800是用户区的首地址（指文本状态），或者说是存放程序的起始地址；而在监控状态下往后扫描程序区，连续出现三个全0字节，就是程序尾。所以，我们在用CALL-151↙后出现＊时，键入0800↙，然后继续扫描（即不断按RETURN键），看到连续出现三个全0字节时停止，就找到了上述BASIC程序对应的机器语言程序的程序尾。

现在，我们假定用了NEW命令，再用LIST命令时，显然，原程序的清单什么也列不出来。

我们再回到监控状态，键入800·841↙，再察看程序区，发现$801和$802地址中的内容改变了，分别由0B 08改为00 00了，但其它地址中的内容，并没有任何变动。这就给我们一个启发，恢复$801和$802中的值，就恢复了

链头，这样程序就连接上了，可用LIST命令打出清单。

具体的做法是，跳过$801至$804（它们是链指针和行号），从$805开始往后查找第一次出现的00字节，这是程序行的终止标志。下一个字节为另一个程序行的头（链指针），本例中，它的位置是$080B。这样，只要把$801中改为$0B，把$802中改为$08，程序就连接上了。用LIST命令，就可以看到上述被NEW掉的BASIC程序。

即在监控状态下，键入 801：0B 08↵即可，返回BASIC状态，LIST就起作用。

此时应注意，不要RUN上述BASIC程序。

如果想运行上述BASIC程序，应在监控状态下，扫描查找连续三个全0字节，它就是程序尾，而三个全0字节后的下一个地址，本例是$0842，必须把这个值放入变量表的首址$69和$6A中去，即把$69号单元改为$42，而把$6A号单元改为$08。具体做法是：JCALL-151↵

*69：42 08↵

返回BASIC状态，这样，恢复了的程序即可运行。

RUN↵

11

9

]

综上所述，只要把程序区的链头接上（本例是$801单元改写为$0B，$802单元改写为$08），被NEW掉的程序即可再现。把变量表首指针改过来（本例是把$69单元改为$42，$6A单元改为$08），即可重新运行程序，否则会把程序冲掉。

若要键入新的程序行（包括修改），或将程序存盘，程序

的尾指针也应改过来。程序尾指针地址是$AF和$BO，即AF：42 08↙。

上述BASIC程序，没有开辟数组，若有数组，则数组首、尾指针也应作相应改动，他们的指针地址分别是$6B，$6C和$6D，$6E。

上面介绍了如何恢复一个被破坏了的程序的方法，通俗的说，就是救回BASIC，从这个例子，给我们一个有益的启示，人为地破坏（或修改）链指针（链头），可以用于程序的加密保护。

事实上，将链值作任何修改，都可以从表面上破坏一个程序。下面介绍的几种保密措施，都是基于上述思想。

3. 几种简单的加密措施

前面已经指出，人为地（有意识地）破坏链指针，可以用于程序的加密。众所周知，在APPLE-Ⅱ或紫金-Ⅱ机上，BASIC程序的存贮是从2049（即16进制的801）这个地址开始的，因此，只要设法改动$801单元的内容，就可以使LIST命令失效。

(1) POKE 2049，0

在编制并调试好程序后，键入POKE 2049，0。结果程序可以运行，但用LIST时，只能在屏幕上显示第一行程序。程序的其它部份什么也看不到，从而达到程序的保密效果。

(2) POKE 2049，1

如果在程序调试完毕后，键入POKE 2049，1，那么执行LIST命令，屏幕上出现连续显示的程序第一行，而毫无休止的进行，这也起到了程序的保密作用。执行RUN命令，

程序正常运行。

(3) POKE 214，128

将要保密的程序运行一次后，键入 POKE 214，128，再键入任何指令或键盘上的任意符号，结果均变成执行RUN命令，而使程序自动运行。当然，程序的清单是列不出来的，并且一行也没有。

(4) POKE 2049，n

为了达到程序加密的目的，把 $801 单元内容置成 n（n是一个小于255的任意正整数），此时情况又不同，屏幕上可能显示一些乱七八糟的东西。

总之，改动某些地址单元的内容，将使列表指令失效，达到程序加密的效果。不言而喻，这些人为地破坏程序的方法，是可以重新恢复的。

4. 几种解密方法

(1) 0 保密的程序

对用 POKE 2049，0 保密的程序，解密的方法之一是，键入一个行号（除已经保存的程序的第一个行号外），然后回车，再执行 LIST 命令，即可显示原程序清单，这种方法简单有效。

(2) 1 保密的程序

对用 POKE 2049，1 保密的程序，用上述简单的方法，键入任意一个行号来解密就失灵了。

破密的方法之一，见2.中救回 BASIC 程序的步骤和方法，因为，人为地破坏链指针，只是局部的，而不是大面积的，所以可以用2.中类似的方法进行修复。

考虑到用2.中的方法比较烦琐，现改用机器语言子程序

的方法。

如前所述，一个NEW命令，可以清除内存现行程序及其变量和数组，这个命令的核心是取下程序区的头一个链指针，并把它置0（即作为程序尾标志），同时，将程序尾、变量首、数组首，尾指针都改成LOMEM+2的值。而程序区的其它部份以及变量表、数组表中的内容等并没有改变，只要将程序区的链头接上，被NEW删掉的程序立即可以再现出来。至于要重新运行程序，还需把变量表首指针改正过来，其它诸如修改，增加新的程序行，存盘等等，程序的尾指针也应作相应的改动。所有这些，都要了解BASIC程序在内存中的存贮方式，这里不再赘述。

下面介绍的是一个机器语言子程序，只要在监控状态下，键入从$6000开始到$6065结束的机器码，并把它以BSAVE PRO-3，A$6000，L$65存盘，就可随时救回被NEW删掉的BASIC程序。同时，存盘的PRO-3程序，还可以对某些加密程序，进行破译。

机器语言子程序PRO-3：

```
PRO-3
*6000.6065

6000- A9 00 85 07 85 06 A9 08
6008- 85 08 A0 05 B1 07 C9 00
6010- F0 0F C8 D0 F7 E6 08 D0
6018- F3 B1 07 C9 00 F0 18 D0
6020- EB A5 06 C9 01 F0 2D A5
6028- 08 8D 02 08 98 8D 01 08
6030- A9 01 85 06 4C 12 60 CB
6038- C8 C0 01 F0 06 30 04 4C
6040- 46 60 ED 08 D0 00 98 85
6048- 69 85 AF A5 08 85 6A 85
```

```
6050- B0 4C 5B 60 C8 D0 C2 E6
6058- 08 D0 BE EE 01 08 D0 03
6060- EE 02 08 4C BF 00
```

假设有一个BASIC程序PRO-4，若对它执行NEW命令，或者进行了加密保护，现在可以利用机器语言子程序PRO-3，对程序PRO-4进行解密，下面是运行的实例。

```
PRO-4
]LIST

5  REM PRO-4
10 A =  PEEK (43634) +  PEEK (4363
     5) * 256
20 B =  PEEK (43616) +  PEEK (4361
     7) * 256
30  PRINT "START ADDRESS:  ";A
40  PRINT "LENGHT:  ";B

]NEW


]LIST


]CALL24576

5  REM PRO-4
10 A =  PEEK (43634) +  PEEK (4363
     5) * 256
20 B =  PEEK (43616) +  PEEK (4361
     7) * 256
30  PRINT "START ADDRESS:  ";A
40  PRINT "LENGHT:  ";B
```

操作方法说明如下：

·CALL-151↙，键入6000 · 6065的机器语言子程序。

·按CTRL-RESET，返回BASIC状态。

·存盘：BSAVE PRO-3，A$6000，L$65。

·清内存：NEW↙

·调一个BASIC程序，如LOAD PRO-4↙，LIST↙，看到PRO-4程序的清单。

·NEW↙，再LIST↙，则名为PRO-4的程序被清除，屏幕上什么也没有。

·BLOAD PRO-3↙

·键入CALL 24576↙，再LIST↙，PRO-4程序又复现在屏幕上。

注意：在救回BASIC程序PRO-4前，必须先调PRO-3程序进内存，否则CALL 24576↙不起作用。

上面介绍的方法，无论对POKE 2049，0、对POKE 2049，1、或者对POKE 2049，n保密的程序，都有满意的解密效果。应用本节介绍的方法时，注意在使用CALL 24576↙后，可能进入监控状态，那不要紧，只要按RESET键，返回BASIC状态，就可进行LIST和RUN了。

5. 软件加密保护实例

下面是一个程序加密的实例，其措施是用“暗码锁”锁住，只有知道暗码的人，才能打开该“锁”，这种方法称之为口令保护，它的想法比较自然，也适合人们对“保密”概念的理解。于是计算机技术和常规的保密技术结合，交相辉映。

(1) 编制密码

为了实现口令保护，首先应确定口令，而这个口令应该是经常变换的，并且让计算机自动编制。例如键入一组字符或数字，计算机自动将它变成一组密码。方法是计算机随机产生一个整数，用它与输入字符的ASCII码运算，而得到新的ASCII码，这样，输入的一组字符就变成别人无法理解的一组密码了。

程序可以这样编写：见PRO-5：

PRO-5

```
ULIST

5  REM PRO-5
10 A =  INT ( RND (1) * 100)
20  INPUT M$
30  FOR I = 1 TO  LEN (M$)
40 A$ =  MID$ (M$,I,1)
50 E =  ASC (A$) + A
60  IF E > 255 THEN E = E - 255
70 N$ = N$ +  CHR$ (E)
80  NEXT I
90  PRINT N$
100  END
```

程序PRO-5中的90句，就是输入字符变成的密码。由于A值是一个随机整数，重复运行上述程序，每次产生的密码均不同。在实际使用上述程序时，应删去90句。

(2) 破译密码

由于上述编制密码的过程是随机的，即是输入相同的字符，每次产生的密码都不一样，所以还要一个破译密码的程序，否则连自己也不能轻易改动源程序，甚至可能无法运行自己的程序。

破译密码的程序，其设计思想同上，只要改动50句的E值为减A，60句的E值为加255，此外，还应保留密码程序中的A值。

(3) 程序实例

例如，我们真正需要保密的程序是：

```
500  FOR I=1 TO 10: PRINT I;
     "  ";: NEXT I
```

为简单起见，我们仅举了一行程序，实际上它可以是一个系统软件。

参照前面介绍的设计思想，可以方便地写出程序 PRO-6，注意 PRINT N$ 的语句已删除。为说明方便， A值取一固定常数，如A＝20。

```
PRO-6
ULIST

5  REM PRO-6
10 A = 20: INPUT "1 OR 2 OR 3    "$
     B: PRINT
20  ON B GOSUB 30,70,28
25  GOTO 10
28  PRINT "END": END
30  INPUT M$:C =  LEN (M$): FOR I =
     1 TO C:A$ =  MID$ (M$,I,1)
40 E =  ASC (A$) + A: IF E > 255 THEN
     E = E - 255
50 N$ = N$ +  CHR$ (E): NEXT
60  RETURN
70  INPUT M$: FOR I = 1 TO  LEN (M
     $):A$ =  MID$ (M$,I,1)
80 E =  ASC (A$) - A: IF E < 0 THEN
     E = E + 255
90 N$ = N$ +  CHR$ (E): NEXT
100  PRINT "INPUT K$"
```

```
105  INPUT K$
110  IF K$ <  > N$ THEN 10
120  GOSUB 500
130  RETURN
500  FOR I = 1 TO 10: PRINT I;" ";
     : NEXT I
510  PRINT
520  RETURN
```

下面是几个运行实例：

(a)
```
ÜRUN
1 OR 2 OR 3   1

?ASD
1 OR 2 OR 3   2

?UgX
INPUT K$
?UgXASD
1 2 3 4 5 6 7 8 9 10
1 OR 2 OR 3   3
```

(b)
```
END

ÜRUN
1 OR 2 OR 3   1

?ASD
1 OR 2 OR 3   2

?UgX
INPUT K$
?UgSASD
1 OR 2 OR 3   3

END
```

(c)
```
ÜRUN
1 OR 2 OR 3   1
```