LRN

TI PPC NOTES

****

V6N9/10, 1981

Newsletter of
the TI Programmable Calculator Club.

9213 Lanham Severn Road, Lanham MD 20706
or P.O. Box 710, Lanham MD 20706.

With this double issue we have completed the second year of our existence as a programmable calculator newsletter. It is traditional to look back at accomplishments. We can be proud of it: we have discovered two important new features on our machines, of which the original designers hadn't thought: Fast Mode and Graphics Mode. We also have advanced the art and technique of programming with numerous tricks and routines and have written and published many good, practical programs. With "we" I do not mean an editorial "we", of course, but rather a collective "we" club members, although some of us have been more active than others. Just thumb through the past pages of the Notes, to see the same names pop up, over and over again. I will not name names, lest I forget somebody, who will feel slighted at the omission. But I thank all the active ones for their continued cooperation. Without you I would be drifting from one TV program into the next and you wouldn't have the Notes to read.

Highlights in this issue? Patrick Acosta's Hexadecimal code implant procedure easily ranks as the star article. But there is also Superchecksum by Jules Bell, which, when proven accurate and reliable after a few months of use by several thousand members and after I have built up enough confidence in it through users' feedback, could be used to check the correct keying in of any published program. There is also Complex Keyboard by Bill Beebe, one of the best Bill ever wrote. And there is also Palmer Hanson's compilation of the Neef Fast Mode entry, together with a (sad?) reminder that it may already have been superceded by Patrick Acosta's new, programmable method. Then there is Morton Matthew's Rhymes, the one that took programming to new and exalted heights by making the TI-59 rhyme.

However, all contributors can be justifiably proud of this issue. It is a fine culmination of two years hard work and dedication.

May I remind you that it is time again to renew your dues. I have enclosed a loose leaf for your convenience. On it you will find a copy of your mailing label. Please make all the corrections you deem necessary on this label. Please use RED pencil or ink.

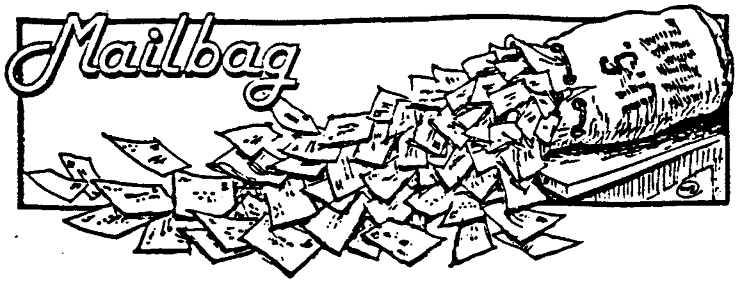
Predictions for the coming year? Judging by the falling prices of the TI-59 one would say that the odds for a new one from TI are rather favorable. As to what exactly it is going to be, your guess is just as good as mine. I have heard the wildest rumors, including one about the calculator having Basic or Pascal. I don't even listen to the rumors anymore. If TI doesn't come out with a new one this coming year, I'll publish some specification in next April issue.

May I wish you all, personally, an old-fashioned merry Christmas and a happy and very prosperous New Year.

See you in January 1982.

Maurice E.T. Swinnen.

IN THIS ISSUE:



Dear Maurice,

First I want to commend you for the fine newsletter. I find it outstanding and a big help.

I am writing to ask if it is possible to arrange for the club members to buy TI equipment through the club at near dealer's discount? This could mean a big savings to the membership and generate funds for the club.

Please keep up the good work,

C.C.D. Hampton VA

Dear C.,

Yes it would. Only, for the time being, the club is not allowed to "sell" anything, lest we void our "non-profit" status. But we are working on an incorporation for next year. We will see what TI thinks about your idea. Ed.

Maurice,

The print-out of the flag has forever sealed the mouth of one of my *****-friends, by the way. He has been staring at his machine for the past week while quitly singing "Oh say, can you print...by the dawns' early light..." or something like that.

Well, enough idle chatter... needlss to say, I have enjoyed the PPC Notes, particularly knowing that there are other fools out there who shamelessly waste their time in the never ending quest for shorter and faster.

Sincerely,

Frank O. Phoenix AZ.

I'll ask Wally to keep an eye on this one too. Seems to be one of the "Fred Fish" variety. Ed.

Dear Maurice,

The new LRN heading, while an improvement, does not send chills up my spine. Now that we have Michael Sperber's Graphic Mode, it might be nice to have a different heading each issue, something like Richard Snow's Stars and Stripes, Spock beaming aboard, etc.

P.A. San Antonio, TX.

Lots of people refer to the newsletter not anymore as the TI PPC NOTES but as LRN. How about keeping it that way? I could still put a nice drawing on the front page each time, as you suggest. Any other or similar votes from the rest of the membership? ED.

Dear Maurice,

Enclosed is a program for....
I saw Fred Fish yesterday. He still works here at Goodyear.
Keep up the quality of the PPC, Maurice. It's outstanding !

W.E.A. Phoenix, Arizona.

Thank you, Wally. And thank you for your report on our problem child Fred. As long as we keep an eye on him, be wont get away this time. Maurice.

Dear Maurice,

This letter will probably arrive after you leave for Belgium.So, welcome home!
With the advent of the new Sharp calculator and other innovations, we are all wondering if you have heard any scuttlebutt (rumors) about TI coming out with a new machine, with alpha keys and increased capacity?
There is no point in our having a spy in with Texas Instruments if we don't get a rumor now and then!

W.E.A. Phoenix, Arizona.

Dear Wally,

The fact that I pay you for keeping an eye on Fred Fish should not give you any ideas about me spying on my (part time) employer: Texas Instruments, so that you people could be furnished with some scuttlebut now and then, satisfying this way your almost insatiable appetite for information.
But, Wally, let me tell you what I heard from the sister of the secretary of Mr. Bucy, the brother of TI's general director..........

Maurice.

Dear Mr. Swinnen,

Your problem *(100 Random integers, v5n8)* is nothing more (or less) than that of shuffling a deck of cards. Rather than tagging the cards of an ordinary deck with integers from 1 to 52, imagine you have a deck with size M-N+1 tagged with the integers from N to M. Then shuffle the deck. (assumes M > N)

Assume the integer range desired is from 5 to 11 (M=11, N=5) The deck size is 11-5+1=7 and the card tags will be 5, 6, 7, 8, 9, 10 and 11. Shuffle this 7 card deck and you have the desired result. This is easily generalized to 100 integers or 1000. Incidentally, the card shuffle algorithms I have seen are either correct and slow or else fast and incorrect. (the suffle is not completely honest) A friend of mine, a PHD math type, has invented a method that is fast, correct (if the source of random numbers is good) and possibly minimal. He makes no claim for originality. If you wish it, I will supply it.

W.B. San Diego, CA.

While I appreciate your interest, rather than an algorithm (the theory) I would love to see a program that does all that in a time that would put Richard Snow's program to shame. As with most things, the proof here is also in the pudding. ED

SUPERCHECKSUM,- Jules Bell, Baltimore MD. Somewhere else in this issue is a program called
------------- Shoppinglist. As a comment to it, I said that Jules doesn't want to be known for such programs only. To proof it, here is one Jules definitely wants to be known for. It will generate a unique checksum for each bank of program or data memory you enter and, hence it's name SUPERCHECKSUM, it will subsequently generate a checksum for your entire program.

In v6n6/7p20 I reported on another attempt to write such a program. It appeared in Electronics Design, May 28, 1981, pp 133-135 and was written by Colin Gyles. According to recent information, it will also appear in the next issue of the PPX newsletter. I, personally, found the program rather unfriendly to the user, as it required you to do a lot of manual key punching. Moreover, the program does not handle 8's and 9's in the p-digit position of an octet. So the author recommended checking these troublemakers manually.

Jules's program is super-friendly. Key it in in normal, turn-on partitioning and record it in the same partitioning on one side of a mag card.

Although Jules used the Math/Utility module (and I still think this is the very best module TI, read Don O'Grady, ever produced) he used only those subroutines that give convenient prompting words. In the program he calls, PGM 02, D' prints BAD DATA, B prints READY, C prints RESULT and C' prints OPTION. I don't see why you couldn't simply overwrite those commands each one with a three step printing routine, such as, for example 8 8 PRT for READY, 9 9 PRT for BAD DATA or even 0 1/x PRT. You could also write 4 4 PRT for RESULT and simply leave OPTION out of it by putting NOPS instead. Or, for the adventuresome, write a print code subroutine or two in steps 228 through 239. I will publish your best solutions. And always remember that Jules's program uses direct addressing.

Now, if you want automatic listing of the data registers that this program uses to convert program to data and so do its tricks, press E. If you change your mind, press E again, etc. E contains a flag flip-flopping routine.

Next enter the number of the bank you want to be checked (one at a time, please) and press A. The word READY will be printed. Slide the mag card containing that bank into the slot. Jules's program contains an automatic read routine at this spot.

If you didn't press E (or pressed it an even number of times) registers 00 through 29 will be listed here. In the example printed at the end of the program listing, that spot is indicated with LIST I. By the way, I used, from this issues, Wallace Agy's MAZE, sides 1 and 2 and at the end generated a checksum for the whole program.

Now wait a couple of minutes and the printer will print RESULT, followed by BANK n. Then two numbers will be printed. These constitute the checksum.

Now enter the second bank number and when "ready" appears, slide also that mag card side into the slot. Again, after two minutes you will get a checksum, for the second bank.

If that is all your program to be checksummed has, press B to obtain a checksum for the entire program. Otherwise, wait for this step till all your other banks have been checked, as explained above.

A few notes: Disregard the overflows in the listing of data registers 00 through 29. The have no bearing on the program or its execution.

Also, make sure that no data is left in the registers of the program to be checked, unless that it is a permanent part of it, such as constants or print code registers.

And make absolutely sure that no left-over steps are present byond the last step of your program. Nobody will be able to guess what those steps were and so not be able to generate the same checksum for your program. Those left-over program steps will probably not affect the correct execution of your program, but they do influence the checksum.

And, finally, if you are curious as to the ckecksum of SUPERCHECKSUM itself, it is 37.097 0.5674404 .

The fanatics may obtain a copy of the original eight pages, with lots of explanation as to how it works, by sending $ 2.00 US to the club address: P.O. Box 710, Lanham MD 20706.

SEE PROGRAM ON NEXT PAGE, PLEASE

SUPERCHECKSUM, program listing, Jules Bell.

```
000  65  ×
001  03  3
002  22 INV
003  28 LOG
004  54  )
005  59 INT
006  55  ÷
007  03  3
008  22 INV
009  28 LOG
010  54  )
011  92 RTN
012  65  ×
013  03  3
014  22 INV
015  28 LOG
016  54  )
017  22 INV
018  59 INT
019  92 RTN
020  93  .
021  04  4
022  85  +
023  93  .
024  01  1
025  95  =
026  61 GTO
027  01  01
028  52  52
029  76 LBL
030  15  E
031  87 IFF
032  01  01
033  00  00
034  38  38
035  86 STF
036  01  01
037  91 R/S
038  22 INV
039  86 STF
040  01  01
041  91 R/S
042  32 X:T

043  01  1    B
044  04  4
045  01  1    A
046  03  3
047  03  3    N
048  01  1
049  02  2    K
050  06  6
051  69 OP
052  04  04
053  32 X:T
054  69 OP
055  06  06
056  92 RTN
057  76 LBL
058  11  A
059  42 STO
060  56  56
061  32 X:T
062  00  0
063  22 INV
064  67  EQ
065  00  00
066  71  7:
067  36 PGM
068  02  02
069  19 D'
070  91 R/S
071  32 X:T
072  71 SBR
073  00  00
074  42  42
075  05  5
076  69 OP
077  17  17
078  47 CMS
079  06  6
080  69 OP
081  17  17
082  36 PGM
083  02  02
084  12  B
085  04  4

086  94 +/-
087  22 INV
088  96 WRT
089  87 IFF
090  01  01
091  01  01
092  05  05
093  03  3
094  69 OP
095  17  17
096  00  0
097  22 INV
098  90 LST
099  06  6
100  69 OP
101  17  17
102  25 CLR
103  98 ADV
104  98 ADV
105  03  3
106  00  0
107  42 STO
108  47  47
109  00  0
110  42 STO
111  48  48
112  02  2
113  09  9
114  42 STO
115  49  49
116  73 RC*
117  49  49
118  24 CE
119  50 I×I
120  29 CP
121  67  EQ
122  00  00
123  23  23
124  32 X:T
125  01  1
126  32 X:T
127  67  EQ
128  00  00

129  20  20
130  77  GE
131  01  01
132  34  34
133  35 1/X
134  28 LOG
135  75  -
136  59 INT
137  95  =
138  42 STO
139  46  46
140  71 SBR
141  00  00
142  00  00
143  44 SUM
144  48  48
145  44 SUM
146  59  59
147  43 RCL
148  46  46
149  71 SBR
150  00  00
151  12  12
152  44 SUM
153  48  48
154  44 SUM
155  59  59
156  01  1
157  94 +/-
158  44 SUM
159  49  49
160  97 DSZ
161  47  47
162  01  01
163  16  16
164  36 PGM
165  02  02
166  13  C
167  43 RCL
168  56  56
169  71 SBR
170  00  00
171  42  42

172  98 ADV
173  43 RCL
174  48  48
175  71 SBR
176  00  00
177  00  00
178  99 PRT
179  43 RCL
180  48  48
181  71 SBR
182  00  00
183  12  12
184  99 PRT
185  98 ADV
186  36 PGM
187  02  02
188  18 C'
189  36 PGM
190  02  02
191  12  B
192  98 ADV
193  91 R/S
194  76 LBL
195  12  B
196  36 PGM
197  02  02
198  13  C
199  03  3    P
200  03  3
201  03  3    R
202  05  5
203  03  3    O
204  02  2
205  02  2    G
206  02  2
207  69 OP
208  04  04
209  43 RCL
210  59  59
211  71 SBR
212  00  00
213  00  00
214  99 PRT

215  43 RCL
216  59  59
217  71 SBR
218  00  00
219  12  12
220  69 OP
221  06  06

222  00  0
223  42 STO
224  56  56
225  42 STO
226  59  59
227  91 R/S
```

```
           1.      BANK
       READY           LIST I
       RESULT
           1.      BANK

     33.483
  0.2099888

       OPTION
       READY

           2.      BANK
       READY           LIST II
       RESULT
           2.      BANK

      15.25
     0.5108

       OPTION
       READY

       RESULT
     48.733
  0.7207888       PROG
```

CODES 21 and 26.- In v6n8p1 I reported that John Worthington had discovered a TI-59 "and several TI-58's at the office" which showed a print-out of N! for code 21 and HI* for code 26. They even executed respectively as N-factorial and HIR IND.

When I first received the story it sounded rather incredible. First of all, the patent didn't support it and there were no other reports about it. But then, Worthington and Regelman both claimed it to be the absolute truth. And those two rank high on my ladder of trustworthiness.

So, I send out three copies for review. The three reviewers took their time and sat on the story for ages. Untill finally prodded by me, they threw up, figuratively of course, their collective hands and said they had no way to check the veracity of it. Either is was true or it wasn't. It was as simple as that. They were no experts in that field.

Finally, after all these months of delay and anxiety about the story getting stale, I decided to publish it as one of the most sensational discoveries so far, one of the "eat-your-heart-out-Sperber" variety.

Unfortunately, I had not noticed the exact date of the letter in which John told me first about it: April 1, 1981. You can easily guess the rest. Three days later I received a "howling" phone call from John, asking me to tell him what date the letter was written. Emil hasn't called me as yet. It is not a question of courage to face me, I think. From what I hear he is still in traction at a local hospital, a case of excessive laughter and jaw dislocation.

Never mind those two scoundrils. Another April first is coming down the road

PROGRAMMING THE TI-55 SLIDE RULE CALCULATOR. This book by Stephen L. Snover and Mark A. Spikell, which was originally slated to be out by January 29, 1982 has been delayed till spring, according to publicist Debra Leitner at Prentice Hall,Inglewood Cliffs, NJ. 201(592-2641)

COMPUTATION: FINITE AND INFINITE MACHINES, Minsky, Printice Hall, Inc. Inglewood Cliffs, N.J., USA. Club member R. Taysum found in section 2.4, p.24 the following statement: "any finite-state machine, if left completely to itself, will fall eventually into a perfectly repetitive pattern."

So he wrote the following short program to demonstrate this:

```
LBL E  +  4  PAU  +  10  =  PAU  DIV  2  =  PAU  GTO E
```

With any number, even zero, in the display, press E and see rather soon the sequence 14, 18, 28 appear, ad infinitum.

FIRMWARE,- Dave Leising: *There is a difference between the firmware in the "old" TI-59 and the "newer" TI-58 C. In the 58C, for example, the Inverse Polar-to-Rectangular routine is two steps shorter than the one in the TI-59 and does not use HIR 7. Starting at location 250 and ending at step 283 the TI-59 reads as follows:*

(HIR 08 X^2 + X:T HIR 07 X^2) $\sqrt{X}$ X:T ((HIR 17 DIV IXI) INV COS + (HIR 18 DIV HIR 17) INV TAN) CE RTN *while the one for the TI-58C reads as:* ((CE DIV X:T HIR 08) INV TAN + (HIR 18 DIV IXI) INV COS) CE (X:T X^2 + HIR 18 X^2) $\sqrt{X}$ X:T RTN

DATA PACKING,- Björn Gustavsson. This is a good candidate for our own module. I mean the data packing routine, NOT Björn himself, of course. It is faster than MU-08. Like MU-08 it divides each register into a number of pseudo registers. Unlike MU-08 all pseudo registers have the same size. Every pseudo register can store an integer N, $0 \leq N \leq MAX$, in which MAX is chosen by the user.

If MAX is chosen, say, to be 12, then each number to be stored is multiplied by a power of 13 and summed to a register. The idea for this neat trick comes from 52-Notes, n2n11p5.

User Instructions:

1. *Enter MAX and call or press A.*
2. *To store a number, enter it and press X:T. Enter the # of the pseudo register you want it stored in and call/press B.*
3. *To recall, enter the # of the pseudo register and press C.*
4. *To swap the X-register's contents with a pseudo register, press X:T. (Remember that what in the display is visible are the first ten digits of the contents of the X-register.) Enter the pseudo register # and call/press D.*

```
000  76 LBL   018  53  (    036  82 HIR   054  54  )    072  59 INT   090  82 HIR   108  54  )
001  13  C    019  59 INT   037  18  18   055  52 EE    073  65  x    091  18  18   109  65  x
002  53  (    020  65  x    038  22 INV   056  22 INV   074  43 RCL   092  54  )    110  82 HIR
003  32 HIR   021  48 EXC   039  59 INT   057  52 EE    075  00  00   093  74 SM*   111  18  18
004  08  08   022  01  01   040  29 CP    058  33 X²    076  54  )    094  01  01   112  54  )
005  55  ÷    023  54  )    041  67  EQ   059  54  )    077  92 RTN   095  82 HIR   113  74 SM*
006  53  (    024  82 HIR   042  00  00   060  82 HIR   078  76 LBL   096  17  17   114  01  01
007  01  1    025  58  58   043  47  47   061  08  08   079  12  B    097  92 RTN   115  82 HIR
008  02  2    026  02  2    044  43 RCL   062  54  )    080  53  (    098  76 LBL   116  17  17
009  55  ÷    027  44 SUM   045  00  00   063  53  (    081  53  (    099  14  D    117  92 RTN
010  43 RCL   028  01  01   046  65  x    064  59 INT   082  32 X:T   100  53  (    118  76 LBL
011  00  00   029  82 HIR   047  53  (    065  75  -    083  82 HIR   101  53  (    119  11  A
012  28 LOG   030  68  68   048  43 RCL   066  53  (    084  07  07   102  32 X:T   120  42 STO
013  54  )    031  53  (    049  00  00   067  24 CE    085  75  -    103  75  -    121  00  00
014  59 INT   032  73 RC*   050  45 Yˣ    068  55  ÷    086  32 X:T   104  32 X:T   122  69 OP
015  42 STO   033  01  01   051  82 HIR   069  43 RCL   087  13  C    105  13  C    123  20  20
016  01  01   034  55  ÷    052  18  18   070  00  00   088  54  )    106  82 HIR   124  92 RTN
017  54  )    035  53  (    053  59 INT   071  54  )    089  65  x    107  07  07
```

PC100 ON THE FRITS.- Another one in the department of wierd failures. And I who was under impression that numeric values couldn't be used as labels !

```
000  76 LBL
000  11  11
000  76 LBL
000  12  12
000  01  1
000  04  4
000  00  0
000  69 OP
000  76 LBL
001  12  12
001  01  1
001  04  4
001  00  0
001  69 OP
001  00  0
001  00  0
000  76 LBL
002  12  12
002  01  1
002  04  4
002  00  0
002  69 OP
002  00  0
002  00  0
003  01  01
003  04  4
003  00  0
003  69 OP
003  00  0
003  00  0
003  00  0
000  01  1
004  04  4
004  00  0
004  69 OP
004  00  0
004  00  0
004  00  0
004  00  0
000  04  4
005  00  0
```

MAZE, - Wallace A. Agy.

This program allows a person to plot his way through an immaginary maze based on a 10 by 10 grid. There is only one route through the maze, and this route is contained in memories 26 thru 41. This group of memories is permanent, and for use the data is transferred to memories 10 thru 25, and is erased as used. The route contained in the program is shown in illustration A. The object is to move from space 0 to space 99 in the fewest possible moves. Only one route is possible, and re-entry to a space is not permittbd.(or necesary).

The maze as shown may take 20 minutes to go through, if one is not allowed to map his course (most valid results) nor has no previous knowledge.

Movement from square to square is accomplished by entering the proper square number as shown in illustration A, and pressing A.

EG: If in square 41; To move UP, enter 40, press A,printout says "40 nter" (entry to that square was allowed)

To move down, enter 42, press A,printout would be "42 WALL", display reads 41 (last square enterred).

To move LEFT enter 31, printout says "31 WALL" (re-entry is not allowed)

To move RIGHT enter 51, printout "51 WALL" indicates entry not allowed (off path of maze).

Regardless of position;

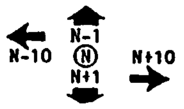
to move right add 10 to present position

to move left subtract 10 from present position

to move up subtract 1 from present position

to move down add 1 to present position

Or to illustrate graphically:

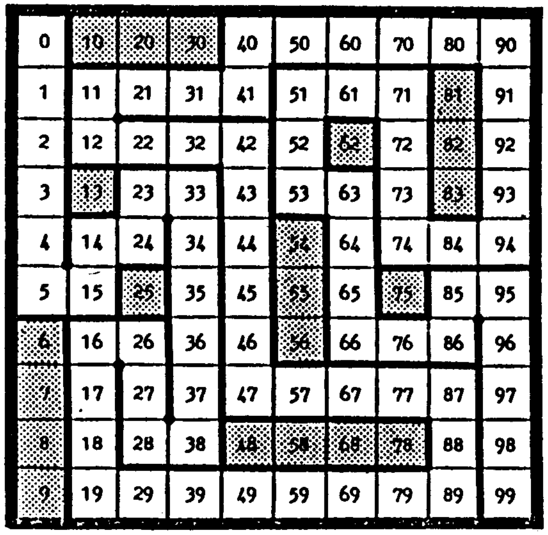


For a valid phsychological test, the instructions are:

"Imagine a 10 by 10 grid with the squares numbered vertically in the first column 0 thru 9, second column 10 thru 19, third column 20 thru 29 and so on thru the last vertical column 90 thru 99. This makes the first row horizontally 0 to 90 in TENS, second row horizontally 1 to 91 in TENS, ETC. Starting in square 0, try to get to 99 in the fewest possible moves. There is only one route through the maze. Enter the desired square number and press A."

You may enter another route by plotting the route and entering the numbers in sequence in reverse order, in memories 26 thru 41. Rules may be changed, such as allowing diagonal moves. Path may be shortened as long as start is "0" and finish is "99".

Instructions: record 4 sides in standard power up status. Press E to initialize. Printer response is "0 nter". To move, follow previous instructions, press A. If entry was correct, printout is the new space along with NTER. If not correct, printout is the attempted move along with WALL, display shows the last valid square enterred. When goal is reached, END is printed along with quantity and type of errors that may be evaluated to show psychologic motivations.



0	10	20	30	40	50	60	70	80	90
1	11	21	31	41	51	61	71	81	91
2	12	22	32	42	52	62	72	82	92
3	13	23	33	43	53	63	73	83	93
4	14	24	34	44	54	64	74	84	94
5	15	25	35	45	55	65	75	85	95
6	16	26	36	46	56	66	76	86	96
7	17	27	37	47	57	67	77	87	97
8	18	28	38	48	58	68	78	88	98
9	19	29	39	49	59	69	79	89	99

Illustration A

MAZE, Wallace Agy, Listing.

```
 0.     NTER
 1.     NTER
 2.     NTER
 3.     NTER
 4.     NTER
 5.     NTER
 6.     WALL
15.     NTER
25.     WALL
16.     WALL
14.     NTER
13.     WALL
24.     NTER
23.     NTER
33.     NTER
```

```
        504.  10
 3332241415.  11
 3837363534.  12
 1716262728.  13
 4939291918.  14
 8889796959.  15
 4757677787.  16
 4243444546.  17
 2111122232.  18
 6050404131.  19
 9291908070.  20
 7374849493.  21
 5251617172.  22
 6665646353.  23
 9695858676.  24
     999897.  25
  504030201.  26
 3332241415.  27
 3837363534.  28
 1716262728.  29
 4939291918.  30
 8889796959.  31
 4757677787.  32
 4243444546.  33
 2111122232.  34
 6050404131.  35
 9291908070.  36
 7374849493.  37
 5251617172.  38
 6665646353.  39
 9695858676.  40
     999897.  41
         0.   42

 1731371735.  55
   43132727.  56
```

```
000  76 LBL
001  11  A
002  42 STO
003  08  08
004  43 RCL
005  09  09
006  32 X:T
007  09  9
008  09  9
009  67  EQ
010  02  02
011  13  13
012  43 RCL
013  08  08
014  67  EQ
015  01  01
016  47  47
017  71 SBR
018  00  00
019  35  35
020  43 RCL
021  56  56
022  69 OP
023  04  04
024  43 RCL
025  08  08
026  69 OP
027  06  06
028  69 OP
029  00  00
030  69 OP
031  22  22
032  43 RCL
033  57  57
034  91 R/S
________________
035  43 RCL
036  08  08
037  32 X:T
038  43 RCL
039  58  58
040  67  EQ
041  00  00
042  79  79
043  43 RCL
044  08  08
045  75  -
046  43 RCL
047  57  57
048  95  =
049  32 X:T
050  43 RCL
051  50  50
052  67  EQ
```

```
053  00  00
054  85  85
055  01  1
056  00  0
057  67  EQ
058  00  00
059  91  91
060  01  1
061  67  EQ
062  00  00
063  91  91
064  01  1
065  00  0
066  94 +/-
067  67  EQ
068  00  00
069  97  97
070  01  1
071  94 +/-
072  67  EQ
073  00  00
074  97  97
075  01  1
076  44 SUM
077  45  45
078  92 RTN
________________
079  01  1
080  44 SUM
081  42  42
082  61 GTO
083  00  00
084  43  43
________________
085  01  1
086  44 SUM
087  46  46
088  61 GTO
089  00  00
090  55  55
________________
091  42 STO
092  50  50
093  01  1
094  44 SUM
095  43  43
096  92 RTN
________________
097  42 STO
098  50  50
099  01  1
100  44 SUM
101  44  44
102  92 RTN
________________
103  76 LBL
104  15  E
105  25 CLR
```

```
106  42 STO
107  08  08
108  42 STO
109  02  02
110  42 STO
111  07  07
112  42 STO
113  50  50
114  01  1
115  00  0
116  42 STO
117  01  01
118  42 STO
119  03  03
120  02  2
121  06  6
122  42 STO
123  04  04
124  69 OP
125  00  00
126  01  1
127  06  6
128  42 STO
129  05  05
130  73 RC*
131  04  04
132  72 ST*
133  03  03
134  69 OP
135  23  23
136  69 OP
137  24  24
138  97 DSZ
139  05  05
140  01  01
141  30  30
142  25 CLR
143  42 STO
144  09  09
145  42 STO
146  57  57
147  43 RCL
148  08  08
149  75  -
150  43 RCL
151  57  57
152  95  =
153  42 STO
154  50  50
155  43 RCL
156  57  57
157  42 STO
158  58  58
```

```
159  43 RCL
160  09  09
161  42 STO
162  57  57
163  73 RC*
164  01  01
165  55  ÷
166  01  1
167  00  0
168  00  0
169  95  =
170  22 INV
171  59 INT
172  65  ×
173  01  1
174  00  0
175  00  0
176  95  =
177  42 STO
178  09  09
179  73 RC*
180  01  01
181  55  ÷
182  01  1
183  00  0
184  00  0
185  95  =
186  59 INT
187  72 ST*
188  01  01
189  69 OP
190  27  27
191  43 RCL
192  07  07
193  32 X:T
194  05  5
195  22 INV
196  67  EQ
197  02  02
198  04  04
199  69 OP
200  21  21
201  25 CLR
202  42 STO
203  07  07
204  43 RCL
205  55  55
206  69 OP
207  04  04
208  43 RCL
209  57  57
210  69 OP
211  06  06
```

```
212  91 R/S
________________
213  69 OP
214  00  00
215  01  1
216  07  7
217  03  3
218  01  1
219  01  1
220  06  6
221  69 OP
222  04  04
223  43 RCL
224  02  02
225  69 OP
226  06  06
227  03  3
228  02  2
229  03  3
230  05  5
231  03  3
232  01  1
233  03  3
234  07  7
235  69 OP
236  04  04
237  43 RCL
238  42  42
239  69 OP
240  06  06
241  25 CLR
242  42 STO
243  42  42
244  03  3
245  06  6
246  03  3
247  07  7
248  03  3
249  05  5
250  01  1
251  03  3
252  69 OP
253  04  04
254  43 RCL
255  46  46
256  69 OP
257  06  06
258  25 CLR
259  42 STO
260  46  46
261  02  2
262  02  2
263  03  3
264  02  2
```

```
265  01  1
266  03  3
267  02  2
268  07  7
269  69 OP
270  04  04
271  43 RCL
272  43  43
273  69 OP
274  06  06
275  25 CLR
276  42 STO
277  43  43
278  03  3
279  05  5
280  03  3
281  07  7
282  01  1
283  05  5
284  06  6
285  05  5
286  69 OP
287  04  04
288  43 RCL
289  44  44
290  69 OP
291  06  06
292  25 CLR
293  42 STO
294  44  44
295  03  3
296  01  1
297  02  2
298  07  7
299  03  3
300  02  2
301  02  2
302  02  2
303  69 OP
304  04  04
305  43 RCL
306  45  45
307  69 OP
308  06  06
309  25 CLR
310  42 STO
311  45  45
312  98 ADV
313  98 ADV
314  98 ADV
315  98 ADV
316  61 GTO
317  15  E
________________
```

SERIES/PARALLEL CONVERTER,- Bill Beebe Jr., Lilburn, Georgia. The two short routines will convert a series resistance and reactance (either a capacitance or an impedance) into an equivalent parallel resistance and reactance or vice-versa. These routines will appeal to the electronics engineers and technicians among the membership. You can use them either as standalone programs or as subroutines in a larger program.

Version 1 is the shortest and, paradoxically, the slowest running. Version 2 maintains standard AOS operation (this will warm Mike Shanok's heart to see that at least one member took Mike's advice in last issue's newcomer's corner) needing three pending operations. Check your manual for P/R and INV P/R requirements, if you plan to use these routines as your subroutines.

Operation, given here for both routines, is simple. Place either resistance OR reactance in the t-register and enter the remaining element in the display. To convert from a series pair to an equivalent parallel pair, press A.

To convert from a parallel pair to an equivalent series pair, press B.

It makes no difference how the lements are entered, that is, in what order, but the converted elements are returned in the same order in which the original elements were entered.

In the listing below, version 1 is on the left, version 2 on the right.

```
000  76 LBL
001  11  A
002  71 SBR
003  00  00
004  15  15
005  35 1/X
006  32 X:T
007  35 1/X
```

```
008  32 X:T
009  92 RTN
________________
010  76 LBL
011  12  B
012  71 SBR
013  00  00
014  05  05
015  32 X:T
```

```
016  22 INV
017  37 P/R
018  32 X:T
019  35 1/X
020  32 X:T
021  37 P/R
022  32 X:T
023  92 RTN
________________
```

```
000  76 LBL
001  11  A
002  71 SBR
003  00  00
004  15  15
005  35 1/X
006  32 X:T
007  35 1/X
008  32 X:T
```

```
009  92 RTN
________________
010  76 LBL
011  12  B
012  71 SBR
013  00  00
014  05  05
015  53  (
016  40 IND
017  55  ÷
```

```
018  53  (
019  32 X:T
020  55  ÷
021  53  (
022  33 X²
023  85  +
024  32 X:T
025  33 X²
```

```
026  85  +
027  32 X:T
028  00  0
029  54  )
030  54  )
031  32 X:T
032  54  )
033  92 RTN
________________
```

COMPLEX KEYBOARD, Bill Beebe JR, Lilburn, Georgia.

STEP	PROCEDURE	ENTER	PRESS		OUTPUT/MODE (see legend below)
1	Add X to Y	jy,x:t,y jx,x:t,x		E A	y. sum,x:t,jsum
2	Subtract X from Y	jy,x:t,y jx,x:t,x		E B	y. difference,x:t,jdifference
3	Multiply Y by X	jy,x:t,y jx,x:t,x		E C	y. product,x:t,jproduct
4	Divide Y by X	jy,x:t,y jx,x:t,x		E D	y. product,x:t,jproduct
5	Find the complex root of a complex number	jy,x:t,y jx,x:t,x		E *B'	y. result,x:t,jresult[1]
6	Raise a complex number.by a complex power	jy,x:t,y jx,x:t,x	SBR	E y^x	y. result,x:t,jresult[1]
7	To find a complex log to a complex base ($\log_X(Y)$)	jy,x:t,y jx,x:t,x	SBR	E *log	y. result,x:t,jresult[1]
8	To exchange X with Y		SBR	x:t	R01:display, R02:t-register y,x:t,jy
9	To clear all		SBR	CLR	R01=R02=display=t-register=0.

STEP	PROCEDURE	ENTER	PRESS	RESULT
The following operations work only on the complex display				
10	Find 1/X	jx,x:t,x	SBR 1/X	1/x',x:t,1/jx'
11	Find $\sqrt{X}$ [1]	"	SBR $\sqrt{X}$	$\sqrt{x}$',x:t,$\sqrt{jx}$'
12	Find X^2	"	SBR x^2	x'^2,x:t,x'^2
13	Find ln(X) [1]	"	SBR lnx	lnx',x:t,jlnx'
14	Find exp(X) [1]	"	*A'	exp(x'),x:t,jexp(x')
15	Find sin (X) [1]	"	SBR *sin	sinx',x:t,jsinx'
16	Find cos (X) [1]	"	SBR *cos	cosx',x:t,jcosx'
17	Find tan (X) [1]	"	SBR *tan	tanx',x:t,jtanx'
18	Find $\tan^{-1}$(X) [1]	"	*E'	x',x:t,jx'
19	Find $\sin^{-1}$(X) [1]	"	*C'	"
20	Find $\cos^{-1}$(X) [1]	"	*D'	"

[1] Leaves calculator in radian mode

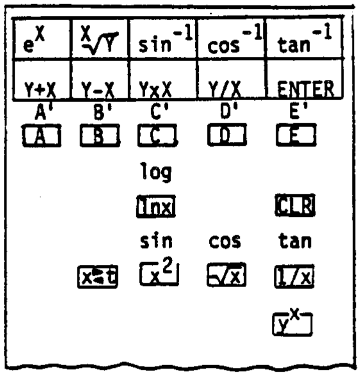
	MEMORY DIRECTLY AFFECTED								SBR	SPECIAL
FUNCTION	H8	H7	R01	R02	R03	R04	#PARENS	#PENDING OPS	LEVELS	FUNCTIONS
A (+)			*	*						
B (-)			*	*						
C (x)			*	*			3	4		
D (/)			*	*			3	4	1	
E (Enter)			*	*						
*A' (e^x)	*	*					SEE P/R-----------		1	P/R
*B' ($\sqrt[x]{y}$)	*	*					3	4		P/R
*C' ($\sin^{-1}$)					*		2	2	1	
*D' ($\cos^{-1}$)					*		2	2	1	
*E' ($\tan^{-1}$)					*	*	3	4		
lnx	*	*					SEE P/R-----------		1	P/R
CLR			*	*						
*log	*	*	*	*			3	4	2	P/R
x:t			*	*						
x^2							3	3		
$\sqrt{x}$	*	*					SEE P/R-----------		1	P/R
1/x							3	3		
*sin	*	*					1	2	1	
*cos	*	*					1	2	1	
*tan	*	*	*	*	*	*	3	4	2	
y^x	*	*	*	*			3	4	2	P/R

Abstract of Program

Through the use of user-defined keys and the SBR key this program turns the standard keyboard into a complex number calculator. The program will do all functions of the MLM programs 4, 5, and 6, except *Pgm 5 C (rectangular-to-polar). The program provides a single level push-type stack for simple chaining of calculations, similar to RPN. This program is completely self-contained and does not call any other module programs.

Method, Equations, Sketches, Limitations, References, Error Recovery:

The program turns the calculator into a complex calculator via the user-defined and the SBR keys. The keyboard layout is shown below.



Each of the user-defined keys accesses a particular function as shown.

Each of these functions is accesed by first pressing SBR before pressing the specific function.
NOTE: Pressing SBR *log will will find the result of log(Y) to the base X.

Memory use is structured in such a way that the display and t-register contain the X operand real and imaginary parts, respectively, while the Y operand real and imaginary parts are stored in R01 and R02, respectively.

When a function involving two operands is called (add, subtract,etc.) the result is in both the complex X and Y storage areas. In other words, the result becomes the new Y. When a function involving only the X operand is called the result is left in the complex X display. The Y storage areas are left undisturbed. The exception to this is the CLeaR and Enter functions, in which the contents of the Y storage areas are overwritten by zeroes or user entered data.

The program always accepts and returns data in the format"imaginary = t-register, real = display."

EXAMPLE: if U = 2+j3, V = 1-j and W = 1+j then evaluate sin(U) + ln(U^2)x(UxV)^W

ENTER	PRESS	OUTPUT	COMMENT
3 x:t 2	E	2.	enter U
1 +/- x:t 1	C	5.	real (UxV)
1 x:t 1	SBR y^x	-1.058423508	real (UxV)^W
3 x:t 2	SBR x^2	-5.	real (U^2)
	SBR lnx	2.564949357	real ln(U^2)
	C	-10.67460184	real ln(U^2)x(UxV)^W
3 x:t 2	SBR *sin	9.154499147	real sin(U)
	A	-1.520102692	real
	x:t	4.137630867	imaginary

See program listing on next page.

ERRATUM, (to v6n6/7p30, Newcomer's Corner). The examples given to prove that one cannot call more than six nested subroutines, even when using user-defined labels, were not well chosen, I am afraid. But here is a routine that will easily demonstrate my point:

```
LBA A 1 STO 00 LBL B  E  PAU 1 SUM 00 B LBL E  RCL 00 RTN
```

Press A and see the display count up to 6, then become blank for a few seconds and stop with a 7 displayed. Loop B, with its subroutine call to E, has been executed six times and stops at the seventh attempt.
Now insert a GTO just after SUM 00 before B. (to do it, in calculator (keyboard) mode, press GTO 012 LRN 2nd INS GTO LRN , and the GTO will in in place.)
Now press A gain and see the display count up ad infinitum.

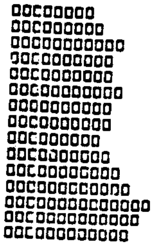
COMPLEX KEYBOARD, Bill Beebe, Jr. Listing.

```
000  76 LBL
001  25 CLR
002  25 CLR
003  29 CP
004  76 LBL
005  15  E
006  42 STO
007  01  01
008  32 X:T
009  42 STO
010  02  02
011  32 X:T
012  92 RTN
013  76 LBL
014  32 X:T
015  48 EXC
016  01  01
017  32 X:T
018  48 EXC
019  02  02
020  32 X:T
021  92 RTN
022  76 LBL
023  12  B
024  94 +/-
025  32 X:T
026  94 +/-
027  32 X:T
028  76 LBL
029  11  A
030  44 SUM
031  01  01
032  43 RCL
033  01  01
034  32 X:T
035  44 SUM
036  02  02
037  43 RCL
038  02  02
039  32 X:T
040  92 RTN
041  76 LBL
042  14  D
043  71 SBR
044  35 1/X
045  76 LBL
046  13  C
047  53  (
048  46 INS
049  65  ×
050  53  (
051  32 X:T
052  65  ×
053  53  (
054  32 X:T
055  65  ×
056  43 RCL
057  01  01
058  75  -
059  43 RCL
060  02  02
061  65  ×
062  32 X:T
063  54  )
064  48 EXC
065  01  01
066  54  )
067  32 X:T
068  85  +
069  32 X:T
070  54  )
071  42 STO
072  02  02
073  32 X:T
074  43 RCL
075  01  01
076  92 RTN
077  76 LBL
078  35 1/X
079  53  (
080  46 INS
081  55  ÷
082  53  (
083  32 X:T
084  94 +/-
085  55  ÷
086  53  (
087  33 X²
088  85  +
089  32 X:T
090  33 X²
091  85  +
092  32 X:T
093  00  0
094  54  )
095  54  )
096  32 X:T
097  54  )
098  92 RTN
099  76 LBL
100  33 X²
101  53  (
102  32 X:T
103  65  ×
104  53  (
105  32 X:T
106  65  ×
107  53  (
108  33 X²
109  75  -
110  32 X:T
111  33 X²
112  54  )
113  32 X:T
114  02  2
115  54  )
116  54  )
117  32 X:T
118  92 RTN
119  76 LBL
120  34 √X
121  32 X:T
122  22 INV
123  37 P/R
124  32 X:T
125  34 √X
126  53  (
127  32 X:T
128  55  ÷
129  02  2
130  54  )
131  37 P/R
132  32 X:T
133  92 RTN
134  76 LBL
135  16 A'
136  70 RAD
137  22 INV
138  23 LNX
139  32 X:T
140  37 P/R
141  32 X:T
142  92 RTN
143  76 LBL
144  23 LNX
145  70 RAD
146  32 X:T
147  22 INV
148  37 P/R
149  32 X:T
150  23 LNX
151  92 RTN
152  76 LBL
153  17 B'
154  71 SBR
155  35 1/X
156  76 LBL
157  45 YX
158  71 SBR
159  32 X:T
160  71 SBR
161  23 LNX
162  13  C
163  16 A'
164  61 GTO
165  15  E
166  76 LBL
167  28 LOG
168  71 SBR
169  23 LNX
170  71 SBR
171  32 X:T
172  71 SBR
173  23 LNX
174  61 GTO
175  14  D
176  76 LBL
177  38 SIN
178  71 SBR
179  02  02
180  01  01
181  39 COS
182  71 SBR
183  02  02
184  16  16
185  38 SIN
186  54  )
187  92 RTN
188  76 LBL
189  39 COS
190  71 SBR
191  02  02
192  01  01
193  38 SIN
194  94 +/-
195  71 SBR
196  02  02
197  16  16
198  39 COS
199  54  )
200  92 RTN
201  70 RAD
202  82 HIR
203  07  07
204  32 X:T
205  82 HIR
206  08  08
207  53  (
208  22 INV
209  23 LNX
210  55  ÷
211  02  2
212  75  -
213  61 GTO
214  02  02
215  26  26
216  54  )
217  32 X:T
218  82 HIR
219  18  18
220  53  (
221  22 INV
222  23 LNX
223  55  ÷
224  02  2
225  85  +
226  35 1/X
227  55  ÷
228  04  4
229  54  )
230  53  (
231  46 INS
232  65  ×
233  82 HIR
234  17  17
235  92 RTN
236  76 LBL
237  30 TAN
238  71 SBR
239  32 X:T
240  42 STO
241  03  03
242  32 X:T
243  42 STO
244  04  04
245  43 RCL
246  02  02
247  32 X:T
248  43 RCL
249  01  01
250  71 SBR
251  01  01
252  78  78
253  71 SBR
254  32 X:T
255  71 SBR
256  01  01
257  90  90
258  14  D
259  43 RCL
260  04  04
261  32 X:T
262  43 RCL
263  03  03
264  61 GTO
265  32 X:T
266  76 LBL
267  18 C'
268  71 SBR
269  03  03
270  03  03
271  22 INV
272  38 SIN
273  71 SBR
274  02  02
275  87  87
276  32 X:T
277  94 +/-
278  32 X:T
279  92 RTN
280  76 LBL
281  19 D'
282  71 SBR
283  03  03
284  03  03
285  22 INV
286  39 COS
287  32 X:T
288  53  (
289  43 RCL
290  03  03
291  85  +
292  53  (
293  33 X²
294  75  -
295  01  1
296  54  )
297  34 √X
298  54  )
299  23 LNX
300  94 +/-
301  32 X:T
302  92 RTN
303  53  (
304  70 RAD
305  42 STO
306  03  03
307  85  +
308  01  1
309  54  )
310  53  (
311  53  (
312  33 X²
313  85  +
314  32 X:T
315  33 X²
316  54  )
317  75  -
318  34 √X
319  48 EXC
320  03  03
321  65  ×
322  04  4
323  54  )
324  53  (
325  53  (
326  34 √X
327  75  -
328  48 EXC
329  03  03
330  44 SUM
331  03  03
332  54  )
333  55  ÷
334  02  2
335  22 INV
336  49 PRD
337  03  03
338  54  )
339  94 +/-
340  92 RTN
341  76 LBL
342  10 E'
343  53  (
344  70 RAD
345  65  ×
346  53  (
347  33 X²
348  42 STO
349  03  03
350  94 +/-
351  75  -
352  32 X:T
353  42 STO
354  04  04
355  33 X²
356  85  +
357  01  1
358  54  )
359  32 X:T
360  02  2
361  54  )
362  22 INV
363  37 P/R
364  32 X:T
365  53  (
366  53  (
367  43 RCL
368  03  03
369  85  +
370  53  (
371  43 RCL
372  04  04
373  85  +
374  01  1
375  54  )
376  33 X²
377  54  )
378  55  ÷
379  53  (
380  46 INS
381  75  -
382  04  4
383  65  ×
384  43 RCL
385  04  04
386  54  )
387  54  )
388  53  (
389  23 LNX
390  55  ÷
391  04  4
392  54  )
393  53  (
394  32 X:T
395  55  ÷
396  02  2
397  54  )
398  92 RTN
```

HISTOGRAM SUBROUTINE,-Bill Beebe Jr. In v2n5p2 of the PPX newsletter (September 1978) ------------------- I had a bar graph generator (inside an article called More Joy of the PC100) which I thought was the ultimate of its kind. In less than 3 seconds it printed a bar of 20 characters long. Shorter bars required proportionally shorter times. And indeed, for a long time it stood as the standard of how to write such a routine.

I am sad to say I have been dethroned. Not only have all the (necessary I thought) NOPs been removed but the routine runs faster to boot.

To use it, place a number between zero and 20 in the display, this number representing the height of the bar required, and press or call A.



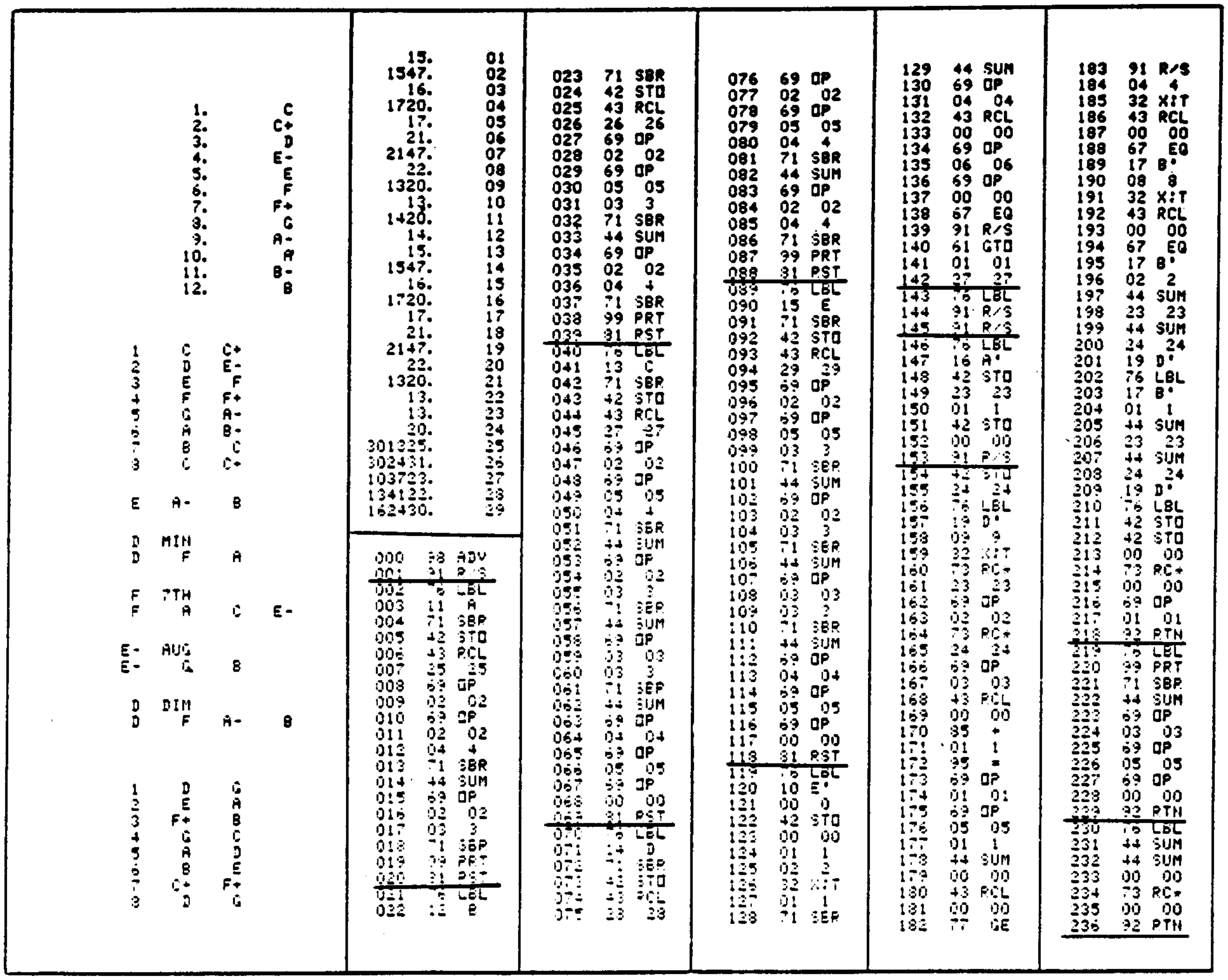
```
000  03   3
001  02   2
002  03   3
003  02   2
004  03   3
005  02   2
006  03   3
007  02   2
008  03   3
009  02   2
010  69  OP
011  01  01
012  03   3
013  02   2
014  03   3
015  02   2
016  03   3
017  02   2
018  03   3
019  02   2
020  03   3
021  02   2
022  69  OP
023  02  02
024  03   3
025  02   2
026  03   3
027  02   2
028  03   3
029  02   2
030  03   3
031  02   2
032  03   3
033  02   2
034  69  OP
035  03  03
036  03   3
037  02   2
038  03   3
039  02   2
040  03   3
041  02   2
042  03   3
043  02   2
044  03   3
045  02   2
046  69  OP
047  04  04
048  69  OP
049  05  05
050  92 RTN
051  76 LBL
052  11   A
053  69  OP
054  00  00
055  55   ÷
056  05   5
057  75   -
058  04   4
059  85   +
060  59 INT
061  55   ÷
062  05   5
063  95   =
064  65   ×
065  01   1
066  00   0
067  95   =
068  50 I×I
069  42 STO
070  00  00
071  83 GO*
072  00  00
```

MUSICAL CHORDS,-Wallace E. Agy of Phoenix, Arizona, wrote this "wierd one."(his own words and admission!) I personally do not think it is so wierd. It is rather practical. The program generates major diatonic scales for comparison and transposing, and if given the key note and the type of chord outputs the tones included in that chord.

The recording instructions are: Key in the program and store the data registers in 6 OP 17 partitioning (power up partitioning) and write banks 1 and 4 on a mag card.

The user instructions are:

1. *Read in banks 1 and 4 in the power-up partitioning.*
2. *Initialize: press 2nd E'. A chromatic scale is listed, next to numbers that will be the key notes. (see sample)*
3. *To output two scales, enter the first key note and press 2nd A'. Enter the second key note and press R/S.*
4. *To list notes in a chord:user-defined labels A through E were designated as follows: A=Major, B=Minor, C=Seventh, D=Augmented and E=Diminished. Enter key code and press appropriate letter key (A through E) and chord will be printed (see sample)*



```
          1.        C
          2.       C+
          3.        D
          4.       E-
          5.        E
          6.        F
          7.       F+
          8.        G
          9.       A-
         10.        A
         11.       B-
         12.        B

1     C    C+
2     D    E-
3     E     F
4     F    F+
5     G    A-
6     A    B-
7     B     C
8     C    C+

E    A-     B

D   MIN
D     F     A

F   7TH
F     A     C    E-

E-  AUG
E-    G     B

D   DIM
D     F    A-     B

1     D     G
2     E     A
3    F+     B
4     G     C
5     A     D
6     B     E
7    C+    F+
8     D     G
```

```
    15.     01
  1547.     02
    16.     03
  1720.     04
    17.     05
    21.     06
  2147.     07
    22.     08
  1320.     09
    13.     10
  1420.     11
    14.     12
    15.     13
  1547.     14
    16.     15
  1720.     16
    17.     17
    21.     18
  2147.     19
    22.     20
  1320.     21
    13.     22
    13.     23
    20.     24
301325.     25
302431.     26
103723.     27
134122.     28
162430.     29
```

```
000  98 ADV
001  91 R/S
002  76 LBL
003  11  A
004  71 SBR
005  42 STO
006  43 RCL
007  25  25
008  69 OP
009  02  02
010  69 OP
011  02  02
012  04  4
013  71 SBR
014  44 SUM
015  69 OP
016  02  02
017  03  3
018  71 SBR
019  99 PRT
020  81 RST
021  76 LBL
022  12  B
023  71 SBR
024  42 STO
025  43 RCL
026  26  26
027  69 OP
028  02  02
029  69 OP
030  05  05
031  03  3
032  71 SBR
033  44 SUM
034  69 OP
035  02  02
036  04  4
037  71 SBR
038  99 PRT
039  81 RST
040  76 LBL
041  13  C
042  71 SBR
043  42 STO
044  43 RCL
045  27  27
046  69 OP
047  02  02
048  69 OP
049  05  05
050  04  4
051  71 SBR
052  44 SUM
053  69 OP
054  02  02
055  03  3
056  71 SBR
057  44 SUM
058  69 OP
059  03  03
060  03  3
061  71 SBR
062  44 SUM
063  69 OP
064  04  04
065  69 OP
066  05  05
067  69 OP
068  00  00
069  81 RST
070  76 LBL
071  14  D
072  71 SBR
073  42 STO
074  43 RCL
075  28  28
076  69 OP
077  02  02
078  69 OP
079  05  05
080  04  4
081  71 SBR
082  44 SUM
083  69 OP
084  02  02
085  04  4
086  71 SBR
087  99 PRT
088  81 RST
089  76 LBL
090  15  E
091  71 SBR
092  42 STO
093  43 RCL
094  29  29
095  69 OP
096  02  02
097  69 OP
098  05  05
099  03  3
100  71 SBR
101  44 SUM
102  69 OP
103  02  02
104  03  3
105  71 SBR
106  44 SUM
107  69 OP
108  03  03
109  03  3
110  71 SBR
111  44 SUM
112  69 OP
113  04  04
114  69 OP
115  05  05
116  69 OP
117  00  00
118  81 RST
119  76 LBL
120  10 E'
121  00  0
122  42 STO
123  00  00
124  01  1
125  02  2
126  32 X:T
127  01  1
128  71 SBR
129  44 SUM
130  69 OP
131  04  04
132  43 RCL
133  00  00
134  69 OP
135  06  06
136  69 OP
137  00  00
138  67  EQ
139  91 R/S
140  61 GTO
141  01  01
142  27  27
143  76 LBL
144  91 R/S
145  91 R/S
146  76 LBL
147  16 A'
148  42 STO
149  23  23
150  01  1
151  42 STO
152  00  00
153  91 R/S
154  42 STO
155  24  24
156  76 LBL
157  19 D'
158  09  9
159  32 X:T
160  73 RC*
161  23  23
162  69 OP
163  02  02
164  73 RC*
165  24  24
166  69 OP
167  03  03
168  43 RCL
169  00  00
170  85  +
171  01  1
172  95  =
173  69 OP
174  01  01
175  69 OP
176  05  05
177  01  1
178  44 SUM
179  00  00
180  43 RCL
181  00  00
182  77  GE
183  91 R/S
184  04  4
185  32 X:T
186  43 RCL
187  00  00
188  67  EQ
189  17 B'
190  08  8
191  32 X:T
192  43 RCL
193  00  00
194  67  EQ
195  17 B'
196  02  2
197  44 SUM
198  23  23
199  44 SUM
200  24  24
201  19 D'
202  76 LBL
203  17 B'
204  01  1
205  44 SUM
206  23  23
207  44 SUM
208  24  24
209  19 D'
210  76 LBL
211  42 STO
212  42 STO
213  00  00
214  73 RC*
215  00  00
216  69 OP
217  01  01
218  92 RTN
219  76 LBL
220  99 PRT
221  71 SBR
222  44 SUM
223  69 OP
224  03  03
225  69 OP
226  05  05
227  69 OP
228  00  00
229  92 RTN
230  76 LBL
231  44 SUM
232  44 SUM
233  00  00
234  73 RC*
235  00  00
236  92 RTN
```

CALCULATOR TIPS AND ROUTINES. (especially for the HP-41C/41CV) Edited by John Dearing. Corvallis Software, P.O. BOX 1412, Corvallis, OR, 97339-1412, USA. 130 pages. Words of praise simply fail me and I feel totally inadequate to do real justice to this marvelous book. It has 26 chapters, on all possible subjects related to the HP-41C. By this time you probably have asked yourself "what this has to do with the TI-59." Just this: I wish somebody would sit down and do the same for our machine. That is, compile all those practical tips and routines known for our calculator in one readable book. There have been attempts, like Harald M. Otto's Trickbuecher, which appear at regular intervals. But nobody has tried his/her hand yet at a real opus. I "bet-ye" this one is going the become THE reference work for years to come. If, like I, you are conversant in RPN, by all means send for one.

PRINTED WIRING DESIGN HOURS.- Wallace E. Agy, Phoenix, AZ. Most electronic devices, (including the TI-59) consist of a number of components placed on a printed circuit. To designers of printed circuits it is of great value to be able to estimate how many hours it is going to take to develop a given circuit. Here is a program that gives a reasonable estimate as to the time to lay it out, to draw the schematic diagram, to draw the assembly drawing and to digitize the printed wiring board. This is done based on the number of holes needed on the board and on it's density.

Output is based on the following formulas, derived (empirically) from actual cases and experience.Factors may be changed in the program if circumstances warrant it.

$$\textit{Lay out hours} = \frac{\textit{Number of holes}}{27} + 16$$

$$\textit{Assembly drawing} = \frac{\textit{Number of holes}}{112} \quad \textit{(minimum 8 hours)}$$

$$\textit{Digitizing} = \frac{\textit{Number of holes}}{750} + \textit{Line density factor' X Number of layers}$$

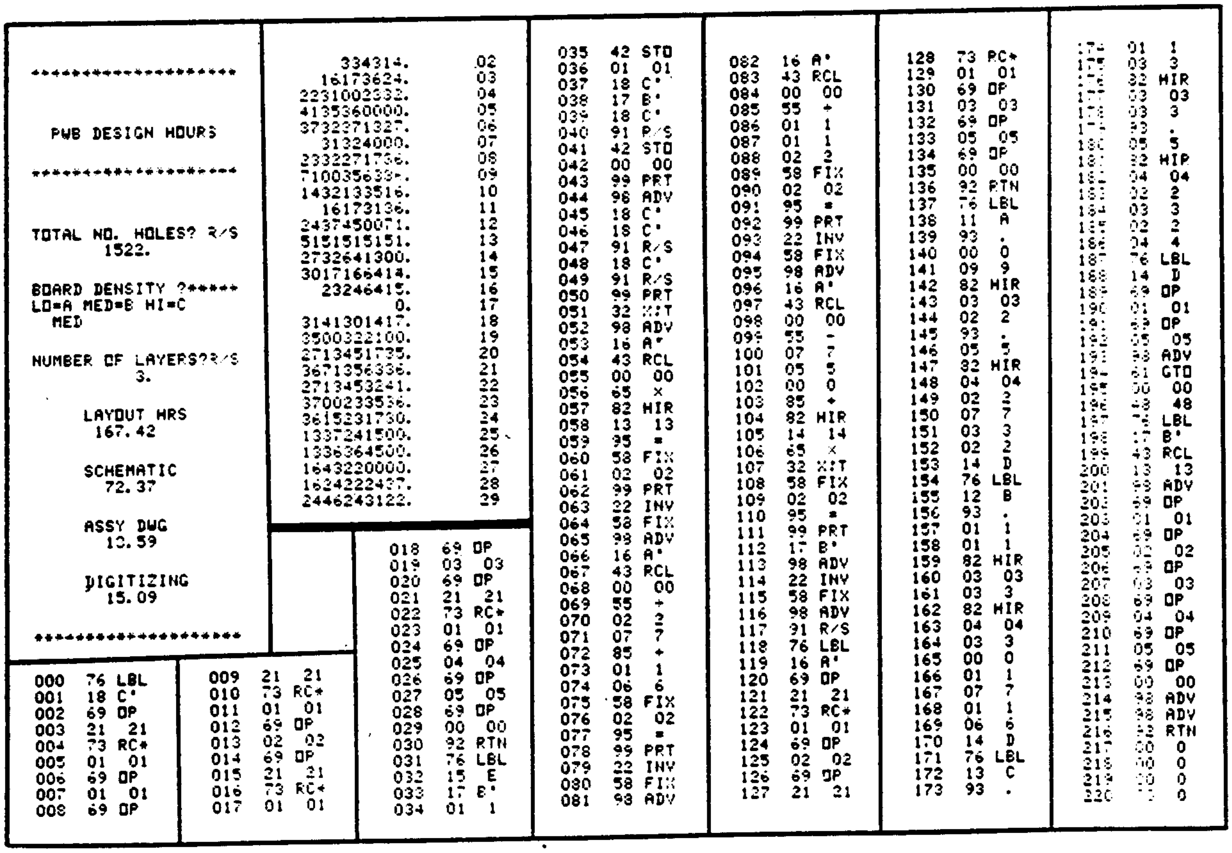
in which factor' = LO= 2.5, MED= 3 and HI=3.5 ;

Example: We calculated the total number of holes to 1522 as follows:

(63) 16 pin dips =	*1008*
(9) 14 pin dips =	*126*
(38) resistors =	*76*
(24) capacitors =	*48*
(8) 8 lead op-amps =	*64*
(2) 50 pin connectors =	*100*
total	*1522*

Estimate the density using experience, based on type of circuit, complexity, area available for, and quantity of, runs.

Key in the program and data memories and record sides 1 and 4. Press E to initialize. The program will print out all prompts. To run again, press E.



```
********************

  PWB DESIGN HOURS

********************

TOTAL NO. HOLES? R/S
       1522.

BOARD DENSITY ?*****
LO=A MED=B HI=C
  MED

NUMBER OF LAYERS?R/S
         3.

     LAYOUT HRS
      167.42

     SCHEMATIC
       72.37

     ASSY DWG
       13.59

     DIGITIZING
       15.09

********************
```

```
     334314.     02
    16173624.    03
  2231002332.    04
  4135360000.    05
  3732371327.    06
    31324000.    07
  2332271736.    08
  7100356334.    09
  1432133516.    10
    16173136.    11
  2437450071.    12
  5151515151.    13
  2732641300.    14
  3017166414.    15
    23246415.    16
           0.    17
  3141301417.    18
  3500322100.    19
  2713451735.    20
  3671356336.    21
  2713453241.    22
  3700233536.    23
  3615231730.    24
  1337241500.    25
  1336364500.    26
  1643220000.    27
  1624222437.    28
  2446243122.    29
```

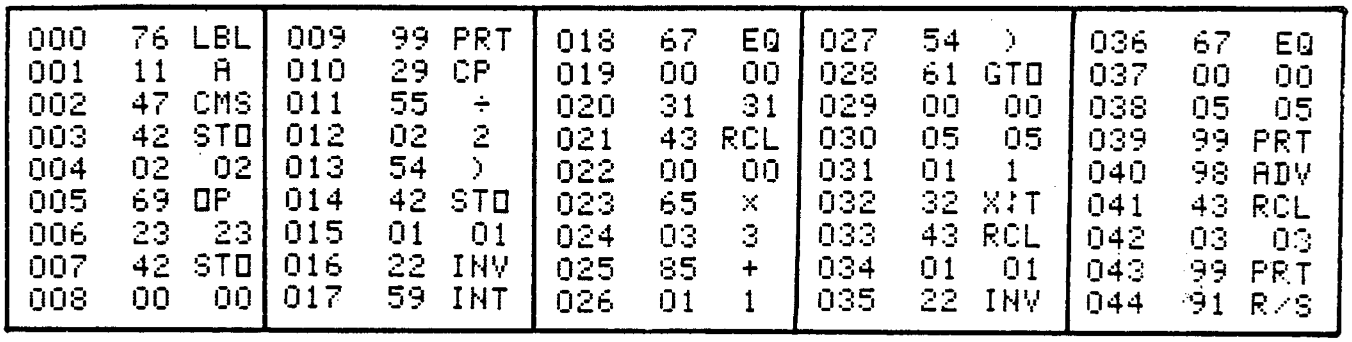
```
000  76 LBL
001  18 C'
002  69 OP
003  21  21
004  73 RC*
005  01  01
006  69 OP
007  01  01
008  69 OP
009  21  21
010  73 RC*
011  01  01
012  69 OP
013  02  02
014  69 OP
015  21  21
016  73 RC*
017  01  01
018  69 OP
019  03  03
020  69 OP
021  21  21
022  73 RC*
023  01  01
024  69 OP
025  04  04
026  69 OP
027  05  05
028  69 OP
029  00  00
030  92 RTN
031  76 LBL
032  15  E
033  17 B'
034  01  1
035  42 STO
036  01  01
037  18 C'
038  17 B'
039  18 C'
040  91 R/S
041  42 STO
042  00  00
043  99 PRT
044  98 ADV
045  18 C'
046  18 C'
047  91 R/S
048  18 C'
049  91 R/S
050  99 PRT
051  32 X:T
052  98 ADV
053  16 A'
054  43 RCL
055  00  00
056  65  x
057  82 HIR
058  13  13
059  95  =
060  58 FIX
061  02  02
062  99 PRT
063  22 INV
064  58 FIX
065  98 ADV
066  16 A'
067  43 RCL
068  00  00
069  55  ÷
070  02  2
071  07  7
072  85  +
073  01  1
074  06  6
075  58 FIX
076  02  02
077  95  =
078  99 PRT
079  22 INV
080  58 FIX
081  98 ADV
082  16 A'
083  43 RCL
084  00  00
085  55  ÷
086  01  1
087  01  1
088  02  2
089  58 FIX
090  02  02
091  95  =
092  99 PRT
093  22 INV
094  58 FIX
095  98 ADV
096  16 A'
097  43 RCL
098  00  00
099  55  ÷
100  07  7
101  05  5
102  00  0
103  85  +
104  82 HIR
105  14  14
106  65  x
107  32 X:T
108  58 FIX
109  02  02
110  95  =
111  99 PRT
112  17 B'
113  98 ADV
114  22 INV
115  58 FIX
116  98 ADV
117  91 R/S
118  76 LBL
119  16 A'
120  69 OP
121  21  21
122  73 RC*
123  01  01
124  69 OP
125  02  02
126  69 OP
127  21  21
128  73 RC*
129  01  01
130  69 OP
131  03  03
132  69 OP
133  05  05
134  69 OP
135  00  00
136  92 RTN
137  76 LBL
138  11  A
139  93  .
140  00  0
141  09  9
142  82 HIR
143  03  03
144  02  2
145  93  .
146  05  5
147  32 HIR
148  04  04
149  02  2
150  07  7
151  03  3
152  02  2
153  14  D
154  76 LBL
155  12  B
156  93  .
157  01  1
158  01  1
159  82 HIR
160  03  03
161  03  3
162  82 HIR
163  04  04
164  03  3
165  00  0
166  01  1
167  07  7
168  01  1
169  06  6
170  14  D
171  76 LBL
172  13  C
173  93  .
174  01  1
175  03  3
176  82 HIR
177  03  03
178  03  3
179  93  .
180  05  5
181  82 HIP
182  04  04
183  02  2
184  03  3
185  02  2
186  04  4
187  76 LBL
188  14  D
189  69 OP
190  01  01
191  69 OP
192  05  05
193  98 ADV
194  61 GTO
195  00  00
196  48  48
197  76 LBL
198  17 B'
199  43 RCL
200  13  13
201  98 ADV
202  69 OP
203  01  01
204  69 OP
205  02  02
206  69 OP
207  03  03
208  69 OP
209  04  04
210  69 OP
211  05  05
212  69 OP
213  00  00
214  98 ADV
215  98 ADV
216  92 RTN
217  00  0
218  00  0
219  00  0
220  00  0
```

ULAM'S (?) CONJECTURE,- *Professor Wilber J. Widmer of the University of Connecticut sent me this interesting program. Several years ago Martin Gardner discussed a remarkable conjecture on numbers in his regular feature, Mathematical Games, in Scientific American. Consider any positive real integer N. If N is even,divide it by 2; otherwise, if odd, multiply it by 3 and add 1. In either case repeat this operation on the new integer I(1) thus formed, and then continue the process to generate a sequence of integers I(1), I(2),...I(n). It is an unproven conjecture that this process will always yield I(n) = 1 in a finite number of steps n.*

For any given N, call this value of n i(N). E.g., for N = 10, the generated sequence is 5, 16, 8, 4, 2, 1 and i(10) = 6; after this the sequence cycles repeatedly with (1), 4, 2, 1. There is no apparent pattern to the sequence of i(N) for N = 0, 1, 2, ... Thus, i(0) = 4, i(1) = 3, i(2) = 1, i(3) = 7, i(4) = 2, etc. N = 1234567890 gives i(N) = 351 and N = 9999999999 gives i(N) = 180. It appears that for N = 10EXP40, i(N) = 395, but I have not been able to confirm this directly.

In a short note on this in the Jan/Feb 1979 HP-PPC Journal (v6n1p9) John Kennedy attributes this conjecture to the Polish mathematician Stanislaw Ulam (who came to the U.S. in the 1930's and is well-known for his work on the Los Alamos project and for a variety of mathematical research writings). But in a recent letter to me, Martin Gardner says that he is not aware that the conjecture is due to Ulam, and John Kennedy could not recall for me the source of his statement.

The TI-58/59 program which follows executes the sequence of any positive real integer N on input by N [A]. With a printer, the full sequence of I values is printed, along with i(N). Without the printer only i(N) is displayed at program stop; if the PRT statements at steps 009 and 039 are replaced with PAUses, the I value will be flashed.



```
000  76 LBL | 009  99 PRT | 018  67  EQ | 027  54   ) | 036  67  EQ
001  11   A | 010  29 CP  | 019  00  00 | 028  61 GTO | 037  00  00
002  47 CMS | 011  55   ÷ | 020  31  31 | 029  00  00 | 038  05  05
003  42 STO | 012  02   2 | 021  43 RCL | 030  05  05 | 039  99 PRT
004  02  02 | 013  54   ) | 022  00  00 | 031  01   1 | 040  98 ADV
005  69 OP  | 014  42 STO | 023  65   × | 032  32 X:T | 041  43 RCL
006  23  23 | 015  01  01 | 024  03   3 | 033  43 RCL | 042  03  03
007  42 STO | 016  22 INV | 025  85   + | 034  01  01 | 043  99 PRT
008  00  00 | 017  59 INT | 026  01   1 | 035  22 INV | 044  91 R/S
```

OP 2.. & OP 3.. ON REGISTERS 10 AND HIGHER.- Page V-27 of the manual shows as the last two lines: OP 20-29 = increment a data register 0-9 by 1.
OP 30-39 = decrement a data register 0-9 by 1.

The power of these OPs lies in the fact that they do not require you to put a numeric value in the display/x-register and as such may be placed almost anywhere in the loop or sequence you are programming. On the other hand, if you require to add or subtract 1 from a register higher than 9, the sequence always has to be "1 SUM NN", with its definite disadvantage of having to put that "1" in the x-register, which will often play havoc with your computation at hand. It more than often is a real problem to find a safe place to put and still have it in the loop.

But if you know "your OAS" this should not be a problem at all. Suppose, in a rather lenghty program, you "used up" all your lower-digit data registers and you still have registers 30 and up available. The sequence you had in mind is the following:

LBL A + RCL IND 00 X RCL IND 01 OP 30 DSZ 1 A =

and your are faced with a possible 1 INV SUM 30 somewhere. (or a 1 +/- SUM 30, if you prefer it better that way) Where do you put the 1 INV SUM 30 ?

Try it this way:

LBL A + RCL IND 30 X 1 INV SUM 30 RCL IND 31 DSZ 31 A =

(only thing to synthesize is the DSZ 31)

How does it work ? Your entry at pressing A, added to what is recalled indirectly from register 30 is pushed into a HIR register for safekeeping at the moment the program encounters the X sign. Then 1 is subtracted from register 31. The "1" in the x-register is subsequently overwritten by what is recalled indirectly from register 31. So, this is the safe spot where to put the 1 INV SUM nn sequence. Maurice Swinnen.

Hexidecimal Keycodes: Each program step in a TI-58/59 occupies one byte of memory and is in a BCD (binary coded decimal) format. This means that each digit (0 to 9) is stored in four bits. But four bits of memory can contain 16 possible "digits". This is the hexidecimal (base 16) system where the "digits" 10 to 15 are represented by the letters A through F respectively. That gives 256 theoretically possible keycodes (00 to FF). It is possible to create 60 of these hex-keycodes on the 58/59.

First you must get into ROM. At turn-on (with the ML module in place) press 3 Op 17 (9 Op 17 for the TI-59) Pgm 12 SBR 444 R/S DMS LRN. Then at a step number in ROM evenly divisible by 8, press Ins and return to RAM by pressing LRN RST. You will find 9 program steps written in user memory (beginning at the step number where you pressed Ins) the first of which will behave strangely. For instance, at step 000, you would see the strange keycode 24which we will denote h24 to distinguish it from the regular keycode for CE. If previously you had a 01 in RAM step 000, you will get the keycode h25;a 02 in RAM step 000 will give the normal keycode 20. This suggests...

RULE 1: Pressing Ins at a step number nnn evenly divisible by 8 in ROM will give the hex-keycode (A0-ab)+cd in RAM where ab is the keycode in ROM step nnn and cd is the keycode previously in RAM step nnn but the first digit of the hex-keycode will be BCD normalized.

For example, with 01 in RAM step 000, pressing INS at ROM step 000 with keycode 82 will put A0-82+01=1F in RAM step 000. The hex digit F is 15 in base 10 and the display will show the hex number 1F as 25 which we will call h25.Note that only the last digit at a step number divisible by 8 can be made hexidecimal. The first digit and the other downloaded keycodes are put into normal BCD form. By writing the proper keycode in RAM and pressing INS the right number of times, any of the 60 possible hex-keycodes can be created at any step number divisible by 8--e.g. to create the hex-keycode 2B at step 000,write SBR in RAM, then go to step 000 in ROM and press INS three times: 3x(A0-82)+71=CB which is BCD normalized to 2B and displayed as 31. Creating the keycodes you want where you want gets easier with a little practice.

RULE 2: Hex-keycodes revert to normal when moved about by editing.

Also note that pressing INS in ROM will move your RAM program up a step so if you want to use more than one hex-keycode in a program, you must start at the bottom of program memory and work your way up.

RULE 3: All the hex-keycodes, except those from h10 to h15, disable any label search.

If you want to jump past a hex-keycode in a program, you must either use absolute addressing or mask the hex-code from the label search by making it part of a two step instruction. RCL hxx will allow label searches without crashes or other quirks. Calling a label that exists before an unmasked hex-code is also alright.

None of these hex-codes print out calenders (sorry) but some do useful things.

h04 acts as a super clear. It clears the display, the t register, EE or ENG mode, the SBR return register, error conditions, all HIR registers (see V5N4/5 p13),resets all flags, goes to step 000, and stops execution. This could save some steps in an initialization routine. h32, h33, h45, h53, h64, h71, h72, and h83 do the same. (This on a 58C. On a 58/59, h04 may clear the program.)

h31, h41, and h51 act as a normal sine function. Writing INV before any of them gives a strange function that is probably used internally to calculate trig functions.

h54 will copy the t register into the display if immediately following x⇄t while preserving pending operations. This is a two step equivalent of copying the display into the t register (V5N6p4). Doing certain things between the x⇄t and h54 (including CP, strangely enough) may produce other results.

(continues on next page.)

h24 is the strangest hex-code of all. Create it at step 000 by pressing INS once and return to user memory and write 7 program steps in 001 to 007 so that you have: 000: h24, AB, CD, EF, GH, IJ, KL, MN
If you then press RST R/S, h24 will change the order in which the keycodes are interpreted to:
000: DA, FC, HE, JG, LI, NK, OL, 00.
This gives the equivalent of 15 program steps packed into 8 steps. SBR 001 will give a 7 step program, and SBR 000 will give an entirely different 8 stepprogram. All the programmer has to do is write his program so that either way the 8 keycodes are interpreted will be useful. An IQ of 160 and nerves of steel are recommended.

The following illustrates the usual behavior of hex-codes in multi-step instructions. Recall that h11 is the display of the hex-code 0B which might be written as 00,11 in base 10. This helps explain some of the following behavior.

```
 RCL h11:: recalls 00, runs SBR A
 FIX h11:: goes into FIX 0, runs SBR A.
 STF h11:: sets flag 0, runs SBR A.
 GTO h11:: runs SBR A, RTNs to step 001.
 SBR h11:: runs SBR A, RTNs to SBR 001, RTNs to step following h11.
 GTO 0x h11:: SBR A, RTNs to step 10x+1.
 SBR 0x h11:: SBR A, RTNs to SBR 10x+1, RTNs to step following h11.
 Dsz 01 h11:: if register 01≠0, runs SBR A, RTNs to step 001.
              if register 01=0, runs SBR A, RTNs to step following h11.
 EQ h11:: if x=t, runs SBR A, RTNS to step 001.
          if x≠t, runs SBR A, RTNs to step following h11.
```

This is the typical kind of behavior but some multi-step instructions will do completely different things.

With 60 hexidecimal keycodes and all the multi-step combinations, I suspect it will take a while before all the possibilities are understood.

Happy hexidecimal programming,
Patrick Acosta

TWELVE DAYS OF X-MAS,- It is traditional that, at the end of the year, I bring you a suitable holiday season program. Last year I brought you the Twelve Days of Christmas and this year I was at a loss to find a good one. I even contemplated a Christmas tree in Graphics Mode, but lack of time prevented me to embark on that adventure.

Then Lem Matteson came to the rescue. You might have seen his name somewhere else in this issue. In fact, I even thought of callingthis issue the Matteson-Beebe-Agy show.

Lem tried the original program by Bill Skillman in v5n9/10p27 and, since it was so short, he thought it might be converted to Fast Mode and still stay within the 159 step limit. However, when he changed the subroutines to GTO instructions with GO returns, it made it too long to fit in bank 1. He was about to give up when he noticed he could drop the "THE END" at the end. So, he ended up with a 163 step program. Now, at this point I, personally, would have loved to get my hands on that program. That is a real challenge, squeezing a program till it screems. Well, Lem squeezed the water out of Bill's program and ended up with the present 155 steps.*

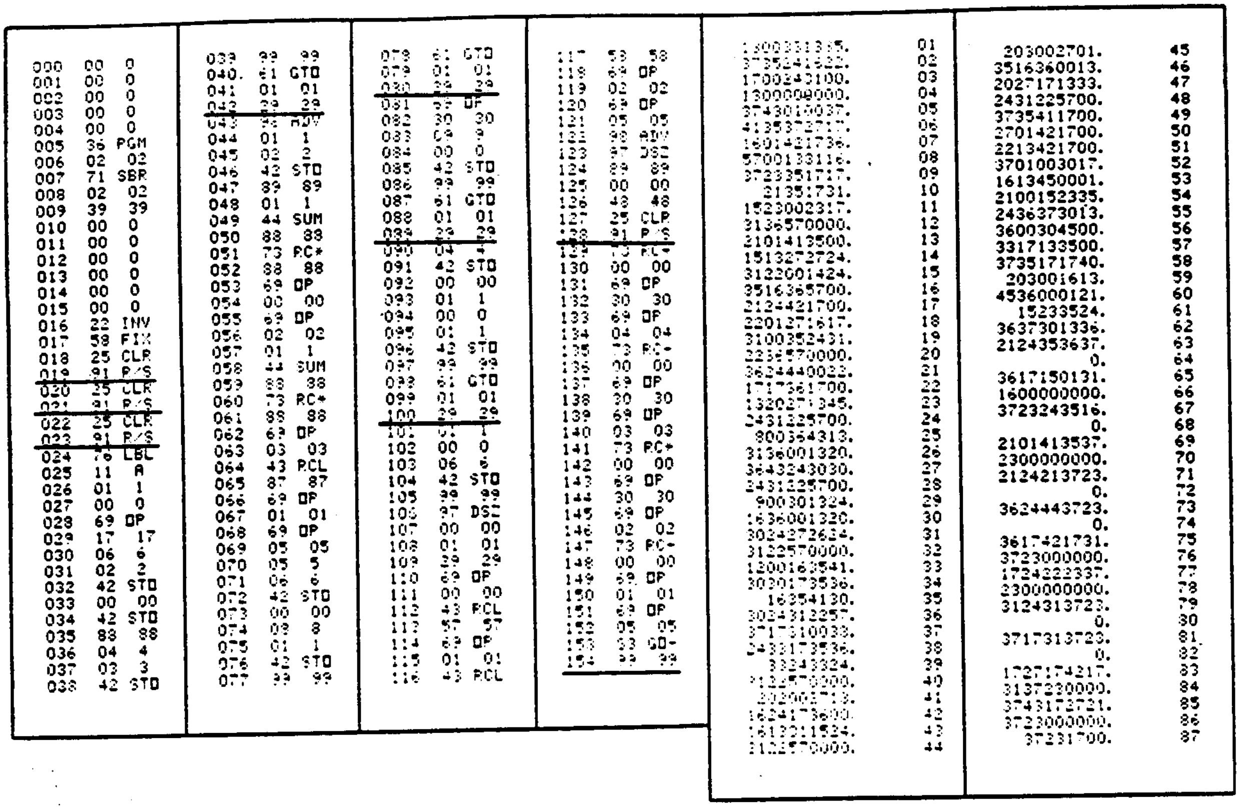
The program may run in either Normal or Fast Mode. In the Normal Mode it takes 4 minutes and 48 seconds to print the song and in Fast Mode this gets reduced to 2 minutes and 12 seconds. For regular mode, the instructions are the same as for Bill's program. Just press A to start the program

For Fast Mode, read card side 1 and press RST R/S. Then load all 4 card sides. The printing starts automatically when the fourth card side is loaded.

Note the zero at step 010, while all the other Fast Mode programs have a 9 there. If you put a 9 in Lem's program, it will print 90000001 when card 1 is entered. Now it prints a 1, as it is supposed to do.

See program listing on next page.

Twelve Days of X-Mas, Lem Matteson, program listing.



```
000  00  0
001  00  0
002  00  0
003  00  0
004  00  0
005  36 PGM
006  02  02
007  71 SBR
008  02  02
009  39  39
010  00  0
011  00  0
012  00  0
013  00  0
014  00  0
015  00  0
016  22 INV
017  58 FIX
018  25 CLR
019  91 R/S
020  25 CLR
021  91 R/S
022  25 CLR
023  91 R/S
024  76 LBL
025  11  A
026  01  1
027  00  0
028  69 OP
029  17  17
030  06  6
031  02  2
032  42 STO
033  00  00
034  42 STO
035  88  88
036  04  4
037  03  3
038  42 STO

039  99  99
040. 61 GTO
041  01  01
042  29  29
043  98 ADV
044  01  1
045  02  2
046  42 STO
047  89  89
048  01  1
049  44 SUM
050  88  88
051  73 RC*
052  88  88
053  69 OP
054  00  00
055  69 OP
056  02  02
057  01  1
058  44 SUM
059  88  88
060  73 RC*
061  88  88
062  69 OP
063  03  03
064  43 RCL
065  87  87
066  69 OP
067  01  01
068  69 OP
069  05  05
070  05  5
071  06  6
072  42 STO
073  00  00
074  08  8
075  01  1
076  42 STO
077  99  99

078  61 GTO
079  01  01
080  29  29
081  69 OP
082  30  30
083  09  9
084  00  0
085  42 STO
086  99  99
087  61 GTO
088  01  01
089  29  29
090  04  4
091  42 STO
092  00  00
093  01  1
094  00  0
095  01  1
096  42 STO
097  99  99
098  61 GTO
099  01  01
100  29  29
101  01  1
102  00  0
103  06  6
104  42 STO
105  99  99
106  97 DSZ
107  00  00
108  01  01
109  29  29
110  69 OP
111  00  00
112  43 RCL
113  57  57
114  69 OP
115  01  01
116  43 RCL

117  58  58
118  69 OP
119  02  02
120  69 OP
121  05  05
122  98 ADV
123  97 DSZ
124  89  89
125  00  00
126  48  48
127  25 CLR
128  91 R/S
129  73 RC*
130  00  00
131  69 OP
132  30  30
133  69 OP
134  04  04
135  73 RC*
136  00  00
137  69 OP
138  30  30
139  69 OP
140  03  03
141  73 RC*
142  00  00
143  69 OP
144  30  30
145  69 OP
146  02  02
147  73 RC*
148  00  00
149  69 OP
150  01  01
151  69 OP
152  05  05
153  93 GO*
154  99  99

1300331335.  01
3735241522.  02
1700243100.  03
1300000000.  04
3743010037.  05
4135372717.  06
1601421736.  07
5700133116.  08
3723351717.  09
  21351731.  10
1523002317.  11
3136570000.  12
2101413500.  13
1513272724.  14
3122001424.  15
3516365700.  16
2124421700.  17
2201271617.  18
3100352431.  19
2235570000.  20
3624440022.  21
1717361700.  22
1320271345.  23
2431225700.  24
 800354313.  25
3136001320.  26
3643243030.  27
2431225700.  28
 900301324.  29
1636001320.  30
3024272624.  31
3122570000.  32
1200163541.  33
3030173536.  34
  16354130.  35
3024312257.  36
3717310033.  37
2431173536.  38
  33243324.  39
3122570000.  40
 202001713.  41
1624173600.  42
1613211524.  43
2122570000.  44

 203002701.  45
3516360013.  46
2027171333.  47
2431225700.  48
3735411700.  49
2701421700.  50
2213421700.  51
3701003017.  52
1613450001.  53
2100152335.  54
2436373013.  55
3600304500.  56
3317133500.  57
3735171740.  58
 203001613.  59
4536000121.  60
  15233524.  61
3637301336.  62
2124353637.  63
         0.  64
3617150131.  65
1600000000.  66
3723243516.  67
         0.  68
2101413537.  69
2300000000.  70
2124213723.  71
         0.  72
3624443723.  73
         0.  74
3617421731.  75
3723000000.  76
1724222337.  77
2300000000.  78
3124313723.  79
         0.  80
3717313723.  81
         0.  82
1727174217.  83
3137230000.  84
3743172721.  85
3723000000.  86
  37231700.  87
```

SURVEY CALCULATIONS JOURNAL.- This journal in v1n7 has some good TI-59 programs. Up to now all the coverage they had was on HP-41C, but things have changed with the last issue. A little critique however seems to be necessary: The authors of those programs write very good RPN programs but after all this time haven't mastered the OAS system yet. I still see things like "))) =" (an equal sign terminates all pending operations, so you may leave all those closing parenthesis off) or "STO 24 X 2 X (RCL 28 DIV RCL 26) = (again, the parenthesis are not needed).

But, the program on Lambert projection is well put together, with HP-41C and TI-59 listings run side by side.

If you want more information about that journal, write Joe Bell at P.O. Box 6674 San Bernardino, CA, 92412.

RHYMES,- Morton P. Matthew, Litchfield, Connecticut. I don't know if you have noticed, but when I want to become solemn, I put the "Letter Gothic 96" typing element on my machine. Here again is such an occasion.

I have seem many versions of the Poem Machine, clean ones and the ones of the saucier kind. I even promised by Swiss friend to publish some of the latter versions in a future issue. All this, however is past history.

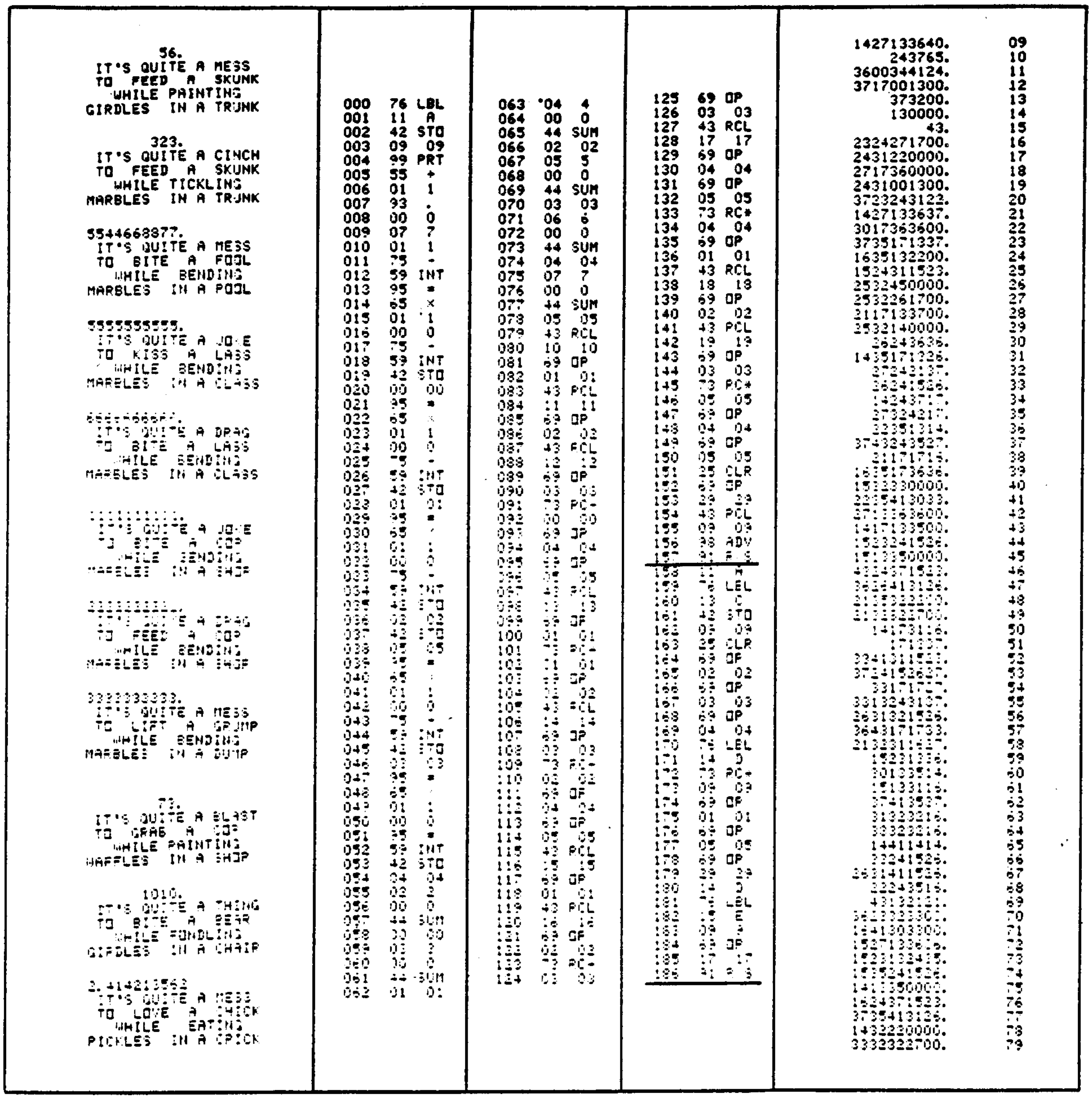
Mankind has made one giant step again on the ladder of progress. We may consider all Poem Machines obsolete from now on, swept away by the tides of forever advancing technology.

Morton P. Matthew, the Consulting Engineer P.E. from Litchfield, CT will go into the annals of programming as the first human being ever to make a TI-59 rhyme. The possibilities are legio, of course: simply substitute the words that turn you on most for the rather clean ones Mort has put in his rhyming machine and you will have produced the most porno of all rhyming programs ever. You don't even have to feel guilty about it, as any subsequent saucy rhyme coming out of the machine will be entirely the machine's doing.

Load the cards in normal partitioning, initialize with E to obtain_239.89 and enter any, we mean ANY, number and press A. We highly recommend 73, 1010 and √2 + 1 =.

Although negative numbers are allowed, we recommend to fasten your seat belts.

RHYMES, M.P. Matthew, Listing and sample.



```
          56.
  IT'S QUITE A MESS
  TO  FEED  A  SKUNK
    WHILE PAINTING
 GIRDLES  IN A TRUNK

          323.
  IT'S QUITE A CINCH
  TO  FEED  A  SKUNK
    WHILE TICKLING
 MARBLES  IN A TRUNK

 5544668877.
  IT'S QUITE A MESS
  TO  BITE  A  FOOL
    WHILE  BENDING
 MARBLES  IN A POOL

 5555555555.
  IT'S QUITE A JOKE
  TO  KISS  A  LASS
    WHILE  BENDING
 MARBLES  IN A CLASS

 6666666666.
  IT'S QUITE A DRAG
  TO  BITE  A  LASS
    WHILE  BENDING
 MARBLES  IN A CLASS

 1111111111.
  IT'S QUITE A JOKE
  TO  BITE  A  COP
    WHILE  BENDING
 MARBLES  IN A SHOP

 2222222222.
  IT'S QUITE A DRAG
  TO  FEED  A  COP
    WHILE  BENDING
 MARBLES  IN A SHOP

 3333333333.
  IT'S QUITE A MESS
  TO  LIFT  A  GRUMP
    WHILE  BENDING
 MARBLES  IN A DUMP

          73.
  IT'S QUITE A BLAST
  TO  GRAB  A  COP
    WHILE PAINTING
 WAFFLES  IN A SHOP

          1010.
  IT'S QUITE A THING
  TO  BITE  A  BEAR
    WHILE FONDLING
 GIRDLES  IN A CHAIR

 2.414213562
  IT'S QUITE A MESS
  TO  LOVE  A  CHICK
    WHILE  EATING
 PICKLES  IN A CRICK
```

```
000  76 LBL
001  11  A
002  42 STO
003  09  09
004  99 PRT
005  55  ÷
006  01  1
007  93  .
008  00  0
009  07  7
010  01  1
011  75  -
012  59 INT
013  95  =
014  65  ×
015  01  1
016  00  0
017  75  -
018  59 INT
019  42 STO
020  00  00
021  95  =
022  65  ×
023  01  1
024  00  0
025  75  -
026  59 INT
027  42 STO
028  01  01
029  95  =
030  65  ×
031  01  1
032  00  0
033  75  -
034  59 INT
035  42 STO
036  02  02
037  42 STO
038  05  05
039  95  =
040  65  ×
041  01  1
042  00  0
043  75  -
044  59 INT
045  42 STO
046  03  03
047  95  =
048  65  ×
049  01  1
050  00  0
051  95  =
052  59 INT
053  42 STO
054  04  04
055  02  2
056  00  0
057  44 SUM
058  00  00
059  03  3
060  00  0
061  44 SUM
062  01  01
```

```
063  04  4
064  00  0
065  44 SUM
066  02  02
067  05  5
068  00  0
069  44 SUM
070  03  03
071  06  6
072  00  0
073  44 SUM
074  04  04
075  07  7
076  00  0
077  44 SUM
078  05  05
079  43 RCL
080  10  10
081  69 OP
082  01  01
083  43 RCL
084  11  11
085  69 OP
086  02  02
087  43 RCL
088  12  12
089  69 OP
090  03  03
091  73 RC*
092  00  00
093  69 OP
094  04  04
095  69 OP
096  05  05
097  43 RCL
098  13  13
099  69 OP
100  01  01
101  73 RC*
102  01  01
103  69 OP
104  02  02
105  43 RCL
106  14  14
107  69 OP
108  03  03
109  73 RC*
110  02  02
111  69 OP
112  04  04
113  69 OP
114  05  05
115  43 RCL
116  15  15
117  69 OP
118  01  01
119  43 RCL
120  16  16
121  69 OP
122  02  02
123  73 RC*
124  03  03
```

```
125  69 OP
126  03  03
127  43 RCL
128  17  17
129  69 OP
130  04  04
131  69 OP
132  05  05
133  73 RC*
134  04  04
135  69 OP
136  01  01
137  43 RCL
138  18  18
139  69 OP
140  02  02
141  43 RCL
142  19  19
143  69 OP
144  03  03
145  73 RC*
146  05  05
147  69 OP
148  04  04
149  69 OP
150  05  05
151  25 CLR
152  69 OP
153  29  29
154  43 RCL
155  09  09
156  98 ADV
157  91 R/S
158  11  A
159  76 LBL
160  13  C
161  42 STO
162  09  09
163  25 CLR
164  69 OP
165  02  02
166  69 OP
167  03  03
168  69 OP
169  04  04
170  76 LBL
171  14  D
172  73 RC*
173  09  09
174  69 OP
175  01  01
176  69 OP
177  05  05
178  69 OP
179  29  29
180  14  D
181  76 LBL
182  15  E
183  09  9
184  69 OP
185  17  17
186  91 R/S
```

```
 1427133640.     09
     243765.     10
 3600344124.     11
 3717001300.     12
     373200.     13
     130000.     14
         43.     15
 2324271700.     16
 2431220000.     17
 2717360000.     18
 2431001300.     19
 3723243122.     20
 1427133637.     21
 3017363600.     22
 3735171337.     23
 1635132200.     24
 1524311523.     25
 2532450000.     26
 2532261700.     27
 2117133700.     28
 2532140000.     29
   26243636.     30
 1435171226.     31
   27243137.     32
   26241526.     33
   14243717.     34
   27324217.     35
   22351314.     36
 3743243527.     37
   21171715.     38
 1635173636.     39
 1532330000.     40
 2235413033.     41
 3713363600.     42
 1417133500.     43
 1532241526.     44
 1513350000.     45
 4324371523.     46
 3624413126.     47
 2113322200.     48
 2113322700.     49
   14173116.     50
     171337.     51
 3341311522.     52
 3724152627.     53
   33171717.     54
 3313243137.     55
 2631321526.     56
 3643171733.     57
 2132311517.     58
   15231316.     59
   30133514.     60
   15133116.     61
   37413537.     62
   31323216.     63
   33323216.     64
   14411414.     65
   22241526.     66
 2631411526.     67
   22243516.     68
   43132121.     69
 3622322300.     70
 1641303300.     71
 1527133615.     72
 1523132415.     73
 1535241526.     74
 1413350000.     75
 1624371523.     76
 3735413126.     77
 1432220000.     78
 3332322700.     79
```

A-MAZE-ING,- In v6n4/5p23-24 I presented a program under that name. It had, however, a
---------- bug or two in it, which made it produce less than perfect mazes once in a while. I asked the membership to come up with a solution. To no avail, I didn't get any takers. So, perusing a software catalog from my friend Peter Poloczek, in Franfurt, West Germany, I noticed a program called Labyrinth, the German word for maze.

Peter was so nice to send me a copy and it is surprisingly similar to our published one. This one, however, seems to run OK. The author is Andreas Biek.

Put your calculator in 3 OP 17, key in the program and record the two mag cards in this partitioning. (One could make it a little more friendly to the user by taking all the unused NOPs out of the program and writing an automatic repartitioning routine somewhere in the program)

You will not need any library module, as the program has its own, built in random number generator.

Read-in both mag cards in 3 OP 17, enter a seed between 0 and 1 and press B.
Enter the number of mazes you want, up to 10, and press C.
Enter the number of lines per maze for the first maze and press R/S, enter the number of lines for the second maze and press R/S, etc.

To start the ball rolling, press CLR A.

A-MAZE-ING, Andreas Biek, listing and sample.

```
                   1.        LAB
                   5.        ZLN
         3.141592654         RND

       IIIIIIIIIIII IIIIIII
       I   I     I      I I
       I I IIIII IIIII I I
       I I I     I        I
       I I I III I IIIIIII
       I I   I            I
       IIIIIIIIIIIIII IIIII

                   2.        LAB
                  10.        ZLN
         .5613352448         RND
```

```
000  76  LBL
001  19  D'
002  53   (
003  43  RCL
004  19   19
005  23  LNX
006  33  X²
007  42  STO
008  19   19
009  65   ×
010  01   1
011  00   0
012  00   0
013  00   0
014  54   )
015  22  INV
016  59  INT
017  92  RTN
018  76  LBL
019  15   E
020  55   ÷
021  00   0
022  32  X:T
023  22  INV
024  28  LOG
025  95   =
026  22  INV
027  59  INT
028  65   ×
029  01   1
030  00   0
031  95   =
032  59  INT
033  92  RTN
034  76  LBL
035  10  E'
036  01   1
037  74  SM*
038  00   00
039  43  RCL
040  00   00
041  65   ×
042  43  RCL
043  15   15
044  22  INV
045  28  LOG
046  52  EE
047  22  INV
048  52  EE
049  95   =
050  44  SUM
051  14   14
052  92  RTN
053  76  LBL
054  18  C'
055  04   4
056  42  STO
057  00   00
058  71  SBR
059  01   01
060  22   22
061  87  IFF
062  01   01
063  01   01
064  04   04
065  43  RCL
066  14   14
067  55   ÷
068  01   1
069  00   0
070  75   -
071  59  INT
072  42  STO
073  14   14
074  95   =
075  69  OP
076  10   10
077  75   -
078  01   1
079  95   =
080  50  I×I
081  65   ×
082  71  SBR
083  01   01
084  31   31
085  97  DSZ
086  15   15
087  00   00
088  97   97
089  71  SBR
090  01   01

091  22   22
092  22  INV
093  97  DSZ
094  00   00
095  01   01
096  15   15
097  87  IFF
098  01   01
099  01   01
100  04   04
101  71  SBR
102  01   01
103  31   31
104  97  DSZ
105  15   15
106  00   00
107  65   65
108  71  SBR
109  01   01
110  22   22
111  97  DSZ
112  00   00
113  00   00
114  65   65
115  69  OP
116  05   05
117  22  INV
118  86  STF
119  01   01
120  92  RTN
121  68  NOP
122  00   0
123  48  EXC
124  16   16
125  84  OP*
126  00   00
127  05   5
128  42  STO
129  15   15
130  92  RTN
131  01
132  04   4
133  65   ×
134  53   (
135  05   5
136  75   -
137  43  RCL
138  15   15
139  54   )
140  22  INV
141  28  LOG
142  33  X²
143  95   =
144  44  SUM
145  16   16
146  92  RTN
147  01   1
148  00   0
149  42  STO
150  00   00
151  69  OP
152  30   30
153  73  RC*
154  00   00
155  22  INV
156  67   EQ
157  01   01
158  51   51
159  92  RTN
160  65   ×
161  09   9
162  42  STO
163  16   16
164  01   1
165  65   ×
166  93   .
167  01   1
168  85   +
169  22  INV
170  59  INT
171  22  INV
172  67   EQ
173  01   01
174  77   77
175  43  RCL
176  00   00
177  52  EE
178  09   9
179  95   =
180  59  INT

181  22  INV
182  52  EE
183  97  DSZ
184  16   16
185  01   01
186  65.  65
187  92  RTN
188  01   1
189  01   1
190  42  STO
191  00   00
192  25  CLR
193  72  ST*
194  00   00
195  97  DSZ
196  00   00
197  01   01
198  93   93
199  92  RTN
200  19  D'
201  65   ×
202  08   8
203  95   =
204  59  INT
205  22  INV
206  28  LOG
207  95   =
208  52  EE
209  22  INV
210  52  EE
211  42  STO
212  10   10
213  65   ×
214  01   1
215  00   0
216  85   +
217  01   1
218  95   =
219  42  STO
220  14   14
221  18  C'
222  92  RTN
223  76  LBL
224  16  A'
225  01   1
226  42  STO
227  12   12
228  43  RCL
229  12   12
230  42  STO
231  13   13
232  32  X:T
233  43  RCL
234  11   11
235  15   E
236  67   EQ
237  03   03
238  75   75
239  42  STO
240  00   00
241  01   1
242  44  SUM
243  12   12
244  43  RCL
245  12   12
246  32  X:T
247  43  RCL
248  11   11
249  15   E
250  22  INV
251  67   EQ
252  02   02
253  41   41
254  19  D'
255  45  YX
256  53   (
257  73  RC*
258  00   00
259  55   ÷
260  06   6
261  00   0
262  54   )
263  65   ×
264  01   1
265  93   .
266  09   9
267  75   -
268  93   .
269  01   1
270  42  STO
271  15   15
272  65   ×
273  22  INV
274  74  SM*
275  00   00
276  73  RC*
277  00   00
278  22  INV
279  59  INT
280  69  OP
281  10   10
282  65   ×
283  43  RCL
284  12   12
285  85   +
286  01   1
287  93   .
288  01   1

289  95   =
290  42  STO
291  16   16
292  22  INV
293  97  DSZ
294  16   16
295  03   03
296  40   40
297  19  D'
298  65   ×
299  02   2
300  95   =
301  59  INT
302  75   -
303  19  D'
304  45  YX
305  05   5
306  95   =
307  50  I×I
308  65   ×
309  53   (
310  43  RCL
311  12   12
312  85   +
313  93   .
314  09   9
315  75   -
316  43  RCL
317  15   15
318  32  X:T
319  43  RCL
320  13   13
321  42  STO
322  15   15
323  95   =
324  59  INT
325  44  SUM
326  15   15
327  43  RCL
328  15   15
329  67   EQ
330  02   02
331  92   92
332  10  E'
334  01   1
335  74  SM*
336  00   00
337  61  GTO
338  02   02
339  92   92
340  01   1
341  44  SUM
342  12   12
343  43  RCL
344  12   12
345  32  X:T
346  09   9
347  77   GE
348  02   02
349  28   28
350  44  SUM
351  14   14
352  43  RCL
353  14   14
354  55   -
355  01   1
356  00   0
357  95   =
358  59  INT
359  42  STO
360  10   10
361  09   9
362  42  STO
363  00   00
364  73  RC*
365  00   00
366  59  INT
367  72  ST*
368  00   00
369  97  DSZ
370  00   00
371  03   03
372  64   64
373  18  C'
374  92  RTN
375  43  RCL
376  12   12
377  32  X:T
378  43  RCL
379  10   10
380  15   E
381  42  STO
382  00   00
383  22  INV
384  67   EQ
385  02   02
386  54   54
387  71  SBR
388  01   01
389  47   47
390  43  RCL
391  13   13
392  42  STO
393  15   15
394  10  E'
395  61  GTO
396  03   03

397  40   40
398  76  LBL
399  17  B'
400  01   1
401  42  STO
402  12   12
403  43  RCL
404  12   12
405  32  X:T
406  43  RCL
408  15   E'
409  42  STO
410  00   00
411  22  INV
412  67   EQ
413  04   04
414  18   18
415  71  SBR
416  01   01
417  47   47
418  09   9
419  87  IFF
420  02   02
421  04   04
422  29   29
423  65   ×
424  93   .
425  07   7
426  65   ×
427  19  D'
428  95   =
429  42  STO
430  13   13
431  22  INV
432  97  DSZ
433  13   13
434  05   05
435  46   46
436  43  RCL
437  12   12
438  85   +
439  01   1
440  95   =
441  32  X:T
442  43  RCL
443  10   10
444  15   E
445  42  STO
446  15   15
447  67   EQ
448  05   05
449  28   28
450  32  X:T
451  43  RCL
452  00   00
453  67   EQ
454  05   05
455  46   46
456  19  D'
457  45  YX
458  53   (
459  53   (
460  73  RC*
461  15   15
462  85   +
463  73  RC*
464  00   00
465  54   )
466  55   ÷
467  02   2
468  00   0
469  95   =
470  32  X:T
471  93   .
472  05   5
473  22  INV
474  87  IFF
475  02   02
476  04   04
477  79   79
478  01   1
479  22  INV
480  77   GE
481  05   05
482  46   46
483  43  RCL
484  00   00
485  32  X:T
486  43  RCL
487  15   15
488  77   GE
489  04   04
490  93   93
491  48  EXC
492  00   00
493  65   ×
494  93   .
495  01   1
496  49  PRD
497  00   00
498  95   =
499  32  X:T
500  43  RCL
501  10   10
502  71  SBR
503  01   01
504  60   60

505  42  STO
506  10   10
507  43  RCL
508  14   14
509  71  SBR
510  01   01
511  60   60
512  42  STO
513  14   14
514  32  X:T
515  42  STO
516  15   15
517  01   1
518  00   0
519  49  PRD
520  00   00
521  49  PRD
522  15   15
523  00   0
524  63  EX*
525  15   15
526  74  SM*
527  00   00
528  43  RCL
529  12   12
530  42  STO
531  15   15
532  10  E'
533  01   1
534  44  SUM
535  12   12
536  43  RCL
537  12   12
538  32  X:T
539  08   8
540  77   GE
541  04   04
542  31   31
543  61  GTO
544  05   05
545  74   74
546  93   .
547  01   1
548  65   ×
549  73  RC*
550  00   00
551  59  INT
552  69  OP
553  10   10
554  95   =
555  74  SM*
556  00   00
557  01   1
558  44  SUM
559  12   12
560  43  RCL
561  12   12
562  32  X:T
563  08   8
564  77   GE
565  04   04
566  03   03
567  09   9
568  32  X:T
569  43  RCL
570  10   10
571  15   E
572  42  STO
573  00   00
574  93   .
575  01   1
576  74  SM*
577  00   00
578  65   ×
579  44  SUM
580  14   14
581  43  RCL
582  14   14
583  59  INT
584  42  STO
585  14   14
586  95   =
587  59  INT
588  42  STO
589  11   11
590  86  STF
591  01   01
592  18  C'
593  92  RTN
594  76  LBL
595  11   A
596  02   2
597  00   0
598  42  STO
599  17   17
600  22  INV
601  86  STF
602  02   02
603  71  SBR
604  01   01
605  88   88
606  01   1
607  44  SUM
608  17   17
609  68  NOP
610  73  RC*
611  17   17
612  68  NOP

613  68  NOP
614  68  NOP
615  42  STO
616  18   18
617  98  ADV
618  02   2
619  07   7
620  01   1
621  03   3
622  01   1
623  04   4
624  69  OP
625  04   04
626  43  RCL
627  17   17
628  75   -
629  02   2
630  00   0
631  95   =
632  69  OP
633  06   06
634  04   4
635  06   6
636  02   2
637  07   7
638  03   3
639  01   1
640  69  OP
641  04   04
642  43  RCL
643  18   18
644  69  OP
645  06   06
646  02   2
647  22  INV
648  44  SUM
649  18   18
650  22  INV
651  49  PRD
652  18   18
653  03   3
654  05   5
655  03   3
656  01   1
657  01   1
658  06   6
659  69  OP
660  04   04
661  43  RCL
662  19   19
663  69  OP
664  06   06
665  98  ADV
666  71  SBR
667  02   02
668  00   00
669  08   8
670  06   6
671  42  STO
672  01   01
673  17  B'
674  16  A'
675  97  DSZ
676  18   18
677  06   06
678  69   69
679  86  STF
680  02   02
681  17  B'
682  71  SBR
683  02   02
684  00   00
685  97  DSZ
686  20   20
687  06   06
688  00   00
689  92  RTN
690  76  LBL
691  12   B
692  42  STO
693  19   19
694  92  RTN
695  76  LBL
696  13   C
697  42  STO
698  20   20
699  02   2
700  01   1
701  42  STO
702  17   17
703  43  RCL
704  17   17
705  75   -
706  02   2
707  00   0
708  95   =
709  91  R/S
710  72  ST*
711  17   17
712  01   1
713  44  SUM
714  17   17
715  61  GTO
716  07   07
717  03   03
718  68  NOP
719  68  NOP
```

407 10 10

333 93 .

RESTRICTIONS & LIMITATIONS OF FAST MODE USING THE PGM-02-SBR-239-9 SEQUENCE.

By Palmer O. Hanson.

1. Entry to fast mode clears all memory, the equivalent of a combined CP and CMs.
2. Entry to fast mode returns the calculator to the turn-on partitioning 6 OP 17. You can re-partition under program control.
3. Entry to fast mode resets all flags if there are zeroes in locations 011 through 015. Use of other digits or codes in those locations has not been researched fully. For example, an OP 00 at locations 014-015 will set flags 1 and 7.
4. Once in fast mode the only allowable keystrokes from the keyboard are +/-, the decimal point, CE, CLR, and EE. Other keystrokes may cause a crash, which will require you to turn off the calculator. But other erratic results have been observed. Sometimes the calculator will simply drop out of fast mode.
   See also paragraphs 12 and 13 below for the use of R/S and RST from the keyboard.
5. Entry to fast mode leaves the calculator in FIX-0 mode. You cannot change the FIX mode from the keyboard, but you may do so under program control.
6. While running in fast mode, the calculator will not recognize an R/S, unless preceded by either CLR, 2nd CLR, PAUse, PRInT, or the sequence EE INV EE.
7. The RTN (code 92) (entered by INV SBR) cannot be used as an alternative to R/S, even when preceded by any of the above alternatives named in paragraph 6.
8. Subroutines may not be used in fast mode. This includes the call up of subroutines from the library modules with any of the options listed in V-62 of the manual.
9. Neither User-defined labels nor Common labels can be used with transfer instructions such as DSZ, t-register comparisons, GTO or the flag tests. These have to be direct addresses. But all of the transfer instructions can be used, if followed by direct addresses.
10. Some operations do not run at fast mode, such as the trigonometric ones, the statistical functions and the conversion functions. The reason for this is that they already run at fast mode when in normal mode.(in the calculator firmware)
11. Once you are in fast mode, you cannot stop the program from the keyboard with R/S or RST. If it doesn't stop under program control, you will have turn the calculator off.
12. An RST from the keyboard, once the program has stopped, drops you out of fast mode.
13. An R/S may be used to start a fast mode program from the keyboard. SBR nnn may be used to start a fast mode program at the location of your choice. An attempt to reposition the program pointer from the keyboard with a GTO nnn causes erratic results as described in paragraph 4, above.
14. Once fast mode has been established, magnetic cards may be used to load a program with a "load-and-go" sort of mode. Program control must be used to provoide one of the sequences in paragraph 6 to stop the calculator after entry of each one of the mag cards, including a double entry of card 1 to provide for the fast mode entry.(see sample program on next page)

USE OF LOAD AND GO TO ENTER A PROGRAM IN FAST MODE.

The command sequence shown provides a safe entry for up to four mag cards, including a double entry for card 1, to provide for the fast mode entry. The sequence of operator actions is based on easy use with the printer. For operations without the printer, change the PRT commands in locations 023, 026, 029 and 032 to PAUse commands, so that an indication of which card has been entered becomes available.

Note: The program tape on the next page includes the instructions for entry into fast mode (steps 000 through 015) for load-and-go entry of four mag cards (steps 016 through 034) and a short fast-mode demonstration program.(steps 035 through 052)

1. Enter card side 1, which includes instructions from locations 000 through 034 as shown on the tape, plus whatever other additional program instructions you intend to run. The display will show you a 1.
2. Press RST R/S. The display will show a zero. You are now in fast mode.
3. Re-enter card side 1. The printer will print 1. The display will show a zero, ready to accept card side 1 again. This procedure is necessary, because the fast mode entry wiped out your program in bank 1.

```
000  0U   0
001  00   0
002  00   0
003  00   0
004  00   0
005  36  PGM
006  02   02
007  71  SBR
008  02   02
009  39   39
010  09   9
011  00   0
012  00   0
013  00   0
014  00   0
015  00   0
016  69  OP
017  00   00
018  22  INV
019  58  FIX
020  22  INV
021  57  ENG
022  01   1
023  99  PRT
024  25  CLR
025  91  R/S
026  99  PRT
027  25  CLR
028  91  R/S
029  99  PRT
030  25  CLR
031  91  R/S
032  99  PRT
033  25  CLR
034  91  R/S
035  76  LBL
036  11   A
037  47  CMS
038  29  CP
039  02   2
040  00   0
041  00   0
042  32  X:T
043  69  OP
044  20   20
045  43  RCL
046  00   00
047  22  INV
048  67   EQ
049  00   00
050  43   43
051  99  PRT
052  91  R/S
```

4. Enter a card to load one of three remaining banks. It is not necessary to load the banks in order, but for the demonatration we will assume that you load in the order 2, 3, 4. So, since you entered the card for bank 2, the program will verify the entry ny printing a 2 and the display will conatin a zero, ready for the next card.
5. Enter the card for bank 3 of your program. The printer verifies the entry by printing a 3, and the display will again contain a zero, to facilitate the entry of the next bank.
6. Enter the card for bank 4 of yor program. The printer will again verify the netry by printing a 4. The display will show a zero. You are now ready to operate your program in fast mode.

To run the sample program, the one Martin Neef used in his original demonstration of fast mode, simply press R/S at the completion of step 6.
Note that the fast mode simply runs past LBL A at 035-036, just as would happen in normal mode. This means that you can write fast mode programs such that you can default to normal mode if you wish.

While the program is running, press either R/S or RST, to demonstrate the inability to stop the program this way.

The calculator will stop after about 37 seconds with 200 in the display and 200 printed. Now press RST to take the calculator out of fast mode and back into normal mode. Then press (user-defined key) A.

This time the calculator will stop after about 70 seconds with 200 in the dsiplay and 200 printed, thus demonstrating the difference in speed between fast mode and normal mode.

If your program does not include four cards, you may delete the appropriate number of PRT-CLR-R/S sequences. Or you may simply press R/S instead of loading a card, in which case the printer shows a zero, to indicate that no card (bank) has been loaded.

FIRMWARE,- Palmer O. Hanson has the following comments on this subject:
-------- "v5n3p6 Reported that assumptions were running amok as to how many firmware program locations exist. There was speculation that the firmaware code may even be repeated as many as nine times for a total of 8000 steps. Table VI of Patent No. 4,153,937 (v6n2p4) would seem to put an end to such speculation. The table shows 512 program locations which are exactly the number needed to support the *__unique__* *code in the various downloadings of the firmware; that is, 380 steps of firmware code, 4 waste steps at locations 380 through 383, and the equivalent 128 steps needed for the 16 constants identified in v3n12p5 by Bill Skillman. The table also supports Patrick Acosta's contention that there is some hexadecimal code at the end of some of the constants; see the OC code at the end of the first constant in table VI of the patent.*

Our downloading of the firmware by various methods (v5n4p6) had interpreted the location 384 of that constant (ln 10) as a 12, equivalent to the keycode B."

HEXADECIMAL CODES FOR NEWCOMERS,- Maurice Swinnen. So that newcomers also might experiment with this new and wonderful discovery, here are some "cook book recipes" you might try out:

1. Make sure the ML library is in place. Also connect your printer.
2. Go into RAM mode by pressing from the keyboard 9 OP 17 PGM 12 SBR 444 R/S DMs LRN
3. See 82. Press INS LRN. Press RST LRN. See 24. This is the famous h24 created at step 000.
4. SST to step 001. Write any program here. For example: X 5 = PAU + 3 = PAU R/S
5. LRN (go out of LRN mode)
6. Press RST R/S
7. The program does not execute properly. Try to trace it. Same result.
8. Enter , say, 3 and execute the program by pressing SBR 001. Now it works. Trace it.
9. Press RST LRN SST and write a new program, say, LBL A X 7 = PRT - 8 = PRT R/S
10. LRN and try to execute the program by entering any digit and pressing A. It won't work. But execute the program with SBR 001 and it works.
11. Turn your calculator off and on again.
12. Go into ROM mode by 9 OP 17 PGM 12 SBR 444 R/S DMs LRN
13. See 82 in the display at step 000.
14. SST to step 144, where there is another 82. Don't be smart now and try GTO 144. It will take you out of ROM mode. Only SST will do, please.
15. At step 144, press INS LRN RST LRN
16. At step 000 now, write another one of your simple programs, say, LBL A X 22 = PRT + 103 = PRT R/S (press LRN to go out of LRN mode)
17. This time say GTO 145(that is just one step beyond where you created h24)
18. LRN and write another simple program: LBL B X 203 = PRT - 18 = PRT R/S
19. Enter any number and press A. Works just fine, prints two results.
20. Enter anything and press B. Nothing doing. Program doesn't work.
21. Enter anything and press SBR 145. It works.
22. Now go back to Patrick Acosta's article on hexadecimal codes and see if it doesn't start making sense.

TRUTH IN LENDING,-Glen Ellis, Memphis TN. Given the amount of the loan, the projected inflation rate, the annual interest and the number of payment periods either in months or years, this program computes and prints the required payment per period, the sum of the payments and the sum of the interest portions. What is unique about this program is, that it also calculates the effect of inflation on the three output categories. It does this by an iterative process, which, admittedly, is rather slow, especially if many periods, such as 240 months of a 30-year loan, are involved. But it is accurate. The program takes about 4 seconds per period, which translates into nearly 24 minutes for 240 periods.

Using the program is rather straightforward and user-friendly:

1. *Enter the amount of the loan and press A.*
2. *Enter the annual interest rate and press B.(12.6 % is entered as 12.6)*
3. *Enter the projected inflation rate and press C.*
4. *Now either enter the number of months and press D OR; enter the number of years and press E.*
5. *The program starts executing at this point and the results will be printed. The payment per period appears relatively fast, but the sum of payments, the sum of the interest portions and the sum of the principal and the effect of inflation on each of them, printed immediately after each output, takes some time.*

You don't have a printer? Don't despair. This user-friendly program can be used without the printer as well. It displays the payment period. If you want to know also the three output values and the effect of inflation on them, press C'. The sum of payments will be shown. Press R/S and the display will show the inflation-effect on this one. Press R/S again and out comes the sum of the interest portions. Another R/S will again show the effect of inflation on the last one. The next R/S produces the sum of the principal and a last R/S shows the inflation-effect on that one too.

If you forgot to write everything down, never mind. Keep pressing R/S. It's a loop

TRUTH IN LENDING, Glen Ellis, Program listing and sample.

```
1000.00    AMT
  18.00    I %
  10.00    INFL
  24.00    MOS
  49.92    PMT

1198.18    ΣPMT
1096.44    EFF

 198.18    ΣINT
 186.31    EFF

1000.00    ΣPRC
 910.13    EFF


1000.00    AMT
  17.50    I %
  12.00    INFL
   1.00    YRS
  12.00    MOS
  91.44    PMT

1097.31    ΣPMT
1057.50    EFF

  97.31    ΣINT
  94.89    EFF

1000.00    ΣPRC
 962.61    EFF
```

```
000  58 FIX
001  02  02
002  43 RCL
003  00  00
004  65  ×
005  43 RCL
006  01  01
007  95  =
008  42 STO
009  07  07
010  94 +/-
011  85  +
012  43 RCL
013  02  02
014  95  =
015  22 INV
016  44 SUM
017  00  00
018  42 STO
019  04  04
020  43 RCL

021  00  00
022  32 X:T
023  69 OP
024  23  23
025  25 CLR
026  77 GE
027  17 B'
028  43 RCL
029  09  09
030  55  ÷
031  53  (
032  43 RCL
033  03  03
034  75  -
035  01  1
036  85  +
037  43 RCL
038  09  09
039  95  =
040  42 STO
041  06  06
042  43 RCL
043  02  02
044  44 SUM
045  13  13
046  65  ×
047  43 RCL
048  06  06
049  95  =
050  44 SUM
051  15  15
052  43 RCL
053  07  07
054  44 SUM
055  08  08
056  65  ×
057  43 RCL
058  06  06
059  95  =
060  44 SUM
061  16  16
062  43 RCL
063  04  04
064  44 SUM
065  10  10
066  65  ×
067  43 RCL
068  06  06
069  95  =
070  44 SUM
071  17  17
072  81 RST
073  76 LBL
074  10 E'
075  22 INV
076  58 FIX
077  01  1
078  07  7
079  02  2

080  01  1
081  02  2
082  01  1
083  00  0
084  00  0
085  69 OP
086  04  04
087  58 FIX
088  02  02
089  92 RTN
090  76 LBL
091  11  A
092  58 FIX
093  02  02
094  47 CMS
095  42 STO
096  00  00
097  42 STO
098  12  12
099  01  1
100  42 STO
101  06  06
102  01  1
103  03  3
104  03  3
105  00  0
106  03  3
107  07  7
108  69 OP
109  04  04
110  43 RCL
111  00  00
112  69 OP
113  06  06
114  91 R/S
115  76 LBL
116  12  B
117  42 STO
118  01  01
119  32 X:T
120  02  2
121  04  4
122  00  0
123  00  0
124  06  6
125  01  1
126  69 OP
127  04  04
128  43 RCL
129  01  01
130  69 OP
131  06  06
132  55  ÷
133  01  1
134  02  2
135  65  ×
136  93  .
137  00  0
138  01  1

139  95  =
140  42 STO
141  01  01
142  32 X:T
143  95  =
144  91 R/S
145  76 LBL
146  13  C
147  22 INV
148  58 FIX
149  32 X:T
150  25 CLR
151  02  2
152  04  4
153  03  3
154  01  1
155  02  2
156  01  1
157  02  2
158  07  7
159  69 OP
160  04  04
161  32 X:T
162  58 FIX
163  02  02
164  42 STO
165  09  09
166  65  ×
167  93  .
168  00  0
169  01  1
170  94 +/-
171  85  +
172  01  1
173  95  =
174  42 STO
175  14  14
176  43 RCL
177  09  09
178  69 OP
179  06  06
180  91 R/S
181  76 LBL
182  14  D
183  42 STO
184  05  05
185  03  3
186  00  0
187  03  3
188  02  2
189  03  3
190  06  6
191  69 OP
192  04  04
193  43 RCL
194  05  05
195  69 OP
196  06  06
197  43 RCL

198  09  09
199  65  ×
200  01  1
201  02  2
202  95  =
203  42 STO
204  09  09
205  76 LBL
206  16 A'
207  43 RCL
208  00  00
209  55  ÷
210  53  (
211  53  (
212  01  1
213  75  -
214  53  (
215  01  1
216  85  +
217  43 RCL
218  01  01
219  54  )
220  45 Yˣ
221  43 RCL
222  05  05
223  94 +/-
224  54  )
225  55  ÷
226  43 RCL
227  01  01
228  95  =
229  42 STO
230  02  02
231  03  3
232  03  3
233  03  3
234  00  0
235  03  3
236  07  7
237  69 OP
238  04  04
239  43 RCL
240  02  02
241  69 OP
242  06  06
243  98 ADV
244  81 RST
245  76 LBL
246  15  E
247  22 INV
248  58 FIX
249  42 STO
250  05  05
251  04  4
252  05  5
253  03  3
254  05  5
255  03  3
256  06  6

257  00  0
258  00  0
259  69 OP
260  04  04
261  43 RCL
262  05  05
263  58 FIX
264  02  02
265  69 OP
266  06  06
267  65  ×
268  01  1
269  02  2
270  95  =
271  61 GTO
272  14  D
273  76 LBL
274  17 B'
275  22 INV
276  58 FIX
277  07  7
278  07  7
279  03  3
280  03  3
281  03  3
282  00  0
283  03  3
284  07  7
285  69 OP
286  04  04
287  58 FIX
288  02  02
289  43 RCL
290  13  13
291  69 OP
292  06  06
293  10 E'
294  43 RCL
295  15  15
296  69 OP
297  06  06
298  98 ADV
299  22 INV
300  58 FIX
301  07  7
302  07  7
303  02  2
304  04  4
305  03  3
306  01  1
307  03  3
308  07  7
309  69 OP
310  04  04
311  58 FIX
312  02  02
313  43 RCL
314  08  08
315  69 OP
316  06  06

317  10 E'
318  43 RCL
319  16  16
320  69 OP
321  06  06
322  98 ADV
323  22 INV
324  58 FIX
325  07  7
326  07  7
327  03  3
328  03  3
329  03  3
330  05  5
331  01  1
332  05  5
333  69 OP
334  04  04
335  58 FIX
336  02  02
337  43 RCL
338  10  10
339  69 OP
340  06  06
341  10 E'
342  43 RCL
343  17  17
344  69 OP
345  06  06
346  22 INV
347  58 FIX
348  98 ADV
349  98 ADV
350  98 ADV
351  98 ADV
352  25 CLR
353  91 R/S
354  76 LBL
355  18 C'
356  43 RCL
357  13  13
358  91 R/S
359  43 RCL
360  15  15
361  91 R/S
362  43 RCL
363  08  08
364  91 R/S
365  43 RCL
366  16  16
367  91 R/S
368  43 RCL
369  10  10
370  91 R/S
371  43 RCL
372  17  17
373  91 R/S
374  61 GTO
375  18 C'
```

PRINTED WIRING BOARD DESIGN ESTIMATE,- Wallace E. Agy, Phoenix, Arizona. As with the Printed Wiring Board Design Hours program elsewhere in this issue, I doubt that many of our members will ever have an oportunity to use it in their professional field. Even among the more than 70 % of our members-electronic engineers and technicians there are precious few who design printed circuits, although all of them have used one time or another. The main reason I present these programs is, because they have very unusual constructs in them, which could be used in many other programs. They are also also perfect examples of how to design an interactive program.

This program will compute the number of layers required in a printed circuit board. This is an important factor in the cost of the board, and if known early in the design, it can help avoid much lost time backtracking and redoing to add additional layers.

The factors which are used are: Digital: 6, Analog: 4 and Digital/Analog: 5. Line density factors are: Low= 3, Medium= 5 and High= 7.

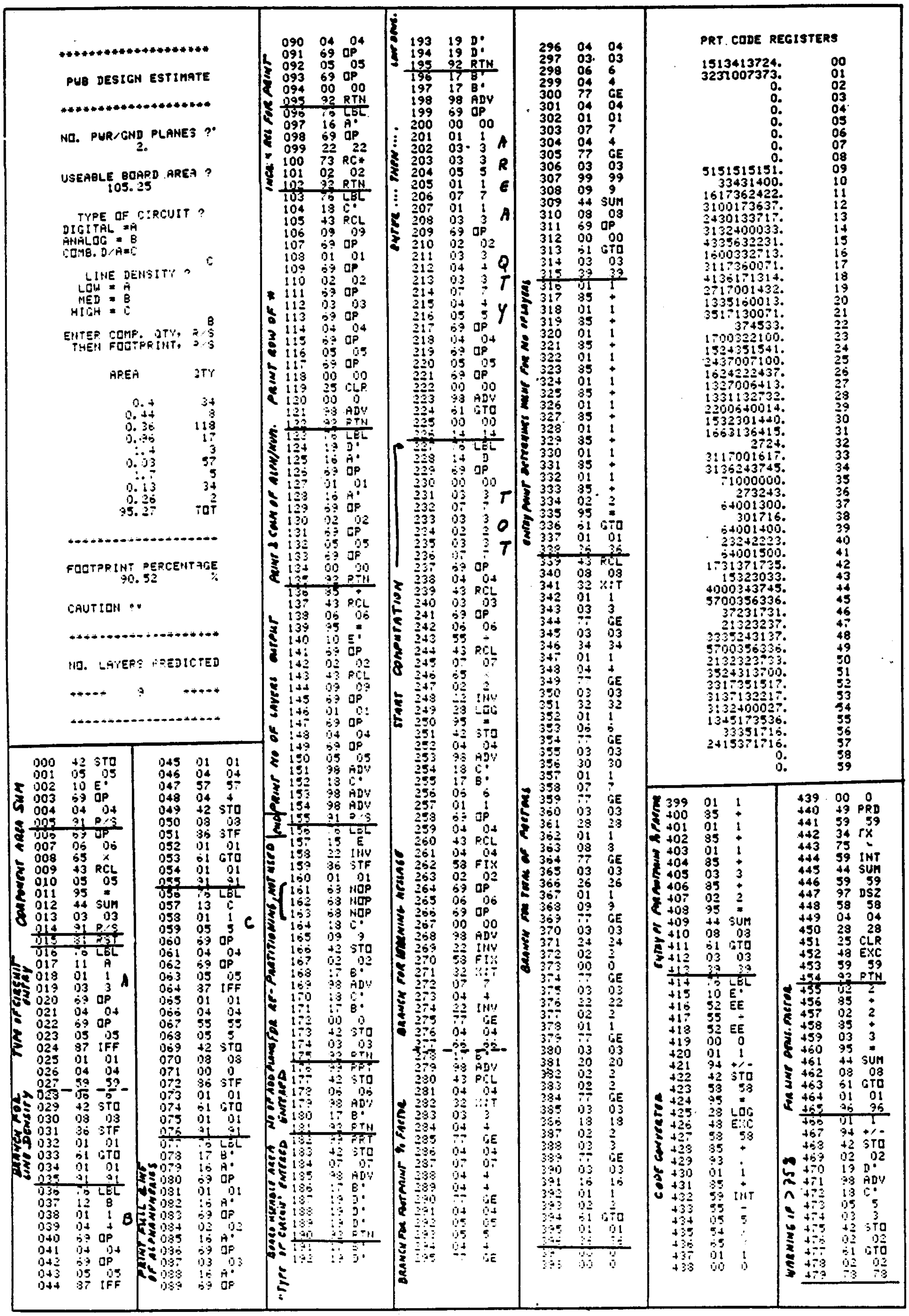
Footprint area:	*15 to 34 % = 2*	*55 to 64 % = 7*
	35 to 44 % = 5	*65 to 74 % = 8*
	45 to 54 % = 6	*75 to 99 % = 9*

A caution message is issued if this factor exceeds 75 %. After summing all factors, total determins number of layers as follows:

TOTAL =	*24*	*23*	*22*	*21*	*20*	*19*	*18*	*17*	*16*	*15*	*14*	*13*	*12*	*11*	*10*	*9*	*8*
LAYERS =	*12*	*11*	*10*	*9*	*8*	*7*	*6*	*5*	*4*	*4*	*3*	*2*	*2*	*2*	*2*	*2*	*2*

Enter cards and press E. Prompts will be printed. Enter number of power and ground planes, will be added to total number of layers. Press A, B or C to enter type of circuit. Press A, B or C to enter line density. Enter quantity of components of a type, then area of it's footprint including leads and pads. When all components have been entered, press D.

Printed Wiring Board, W.E. Agy, program listing and sample calculation.



```
********************

  PWB DESIGN ESTIMATE

********************

NO. PWR/GND PLANES ?'
           2.

USEABLE BOARD AREA ?
        105.25

  TYPE OF CIRCUIT ?
DIGITAL =A
ANALOG = B
COMB.D/A=C
                   C
   LINE DENSITY ?
  LOW = A
  MED = B
 HIGH = C
                    B
ENTER COMP. QTY, R/S
 THEN FOOTPRINT, R/S

       AREA        QTY

        0.4         34
        0.44         8
        0.36       118
        0.96        17
        1.4          3
        0.03        57
        1.7          5
        0.13        34
        0.26         2
       95.27       TOT

********************

FOOTPRINT PERCENTAGE
      90.52          %

CAUTION **

********************

NO. LAYERS PREDICTED

*****     9     *****

********************
```

COMPONENT AREA SUM

TYPE OF CIRCUIT ENTRY

BRANCH FOR LINE DENSITY

A

B

```
000  42 STO
001  05  05
002  10 E'
003  69 OP
004  04  04
005  91 R/S
006  69 OP
007  06  06
008  65  ×
009  43 RCL
010  05  05
011  95  =
012  44 SUM
013  03  03
014  91 R/S
015  81 RST
016  76 LBL
017  11  A
018  01  1
019  03  3
020  69 OP
021  04  04
022  69 OP
023  05  05
024  87 IFF
025  01  01
026  04  04
027  59  59
028  06  6
029  42 STO
030  08  08
031  86 STF
032  01  01
033  61 GTO
034  01  01
035  91  91
036  76 LBL
037  12  B
038  01  1
039  04  4
040  69 OP
041  04  04
042  69 OP
043  05  05
044  87 IFF
```

C

PRINT FULL LINE OF ALPHANUMERICS

```
045  01  01
046  04  04
047  57  57
048  04  4
049  42 STO
050  08  08
051  86 STF
052  01  01
053  61 GTO
054  01  01
055  91  91
056  76 LBL
057  13  C
058  01  1
059  05  5
060  69 OP
061  04  04
062  69 OP
063  05  05
064  87 IFF
065  01  01
066  04  04
067  55  55
068  05  5
069  42 STO
070  08  08
071  00  0
072  86 STF
073  01  01
074  61 GTO
075  01  01
076  91  91
077  76 LBL
078  17 B'
079  16 A'
080  69 OP
081  01  01
082  16 A'
083  69 OP
084  02  02
085  16 A'
086  69 OP
087  03  03
088  16 A'
089  69 OP
```

INCR. & RCL FOR PRINT

PRINT ROW OF *

PRINT & CALC OF ALPH/NUM.

OUTPUT

PRINT NO OF LAYERS AND

FOR RF-PARTITIONING, NOT USED

NO OF ADD PLANES ENTERED

BOARD USABLE AREA "TYPE OF CIRCUIT" ENTERED

```
090  04  04
091  69 OP
092  05  05
093  69 OP
094  00  00
095  92 RTN
096  76 LBL
097  16 A'
098  69 OP
099  22  22
100  73 RC*
101  02  02
102  92 RTN
103  76 LBL
104  18 C'
105  43 RCL
106  09  09
107  69 OP
108  01  01
109  69 OP
110  02  02
111  69 OP
112  03  03
113  69 OP
114  04  04
115  69 OP
116  05  05
117  69 OP
118  00  00
119  25 CLR
120  00  0
121  98 ADV
122  92 RTN
123  76 LBL
124  19 D'
125  16 A'
126  69 OP
127  01  01
128  16 A'
129  69 OP
130  02  02
131  69 OP
132  05  05
133  69 OP
134  00  00
135  92 RTN
136  85  +
137  43 RCL
138  06  06
139  95  =
140  10 E'
141  69 OP
142  02  02
143  43 RCL
144  09  09
145  69 OP
146  01  01
147  69 OP
148  04  04
149  69 OP
150  05  05
151  98 ADV
152  18 C'
153  98 ADV
154  98 ADV
155  91 R/S
156  76 LBL
157  15  E
158  22 INV
159  86 STF
160  01  01
161  68 NOP
162  68 NOP
163  68 NOP
164  18 C'
165  09  9
166  42 STO
167  02  02
168  17 B'
169  98 ADV
170  18 C'
171  17 B'
172  00  0
173  42 STO
174  03  03
175  92 RTN
176  99 PRT
177  42 STO
178  06  06
179  98 ADV
180  17 B'
181  92 RTN
182  99 PRT
183  42 STO
184  07  07
185  98 ADV
186  17 B'
187  19 D'
188  19 D'
189  19 D'
190  92 RTN
191  17 B'
192  19 D'
```

LINE DENS.

ENTER ... THEN ...

AREA QTY

TOT

START COMPUTATION

BRANCH FOR WARNING MESSAGE

BRANCH FOR FOOTPRINT % FACTOR

```
193  19 D'
194  19 D'
195  92 RTN
196  17 B'
197  17 B'
198  98 ADV
199  69 OP
200  00  00
201  01  1
202  03  3
203  03  3
204  05  5
205  01  1
206  07  7
207  01  1
208  03  3
209  69 OP
210  02  02
211  03  3
212  04  4
213  03  3
214  07  7
215  04  4
216  05  5
217  69 OP
218  04  04
219  69 OP
220  05  05
221  69 OP
222  00  00
223  98 ADV
224  61 GTO
225  00  00
226  14  14
227  76 LBL
228  14  D
229  69 OP
230  00  00
231  03  3
232  07  7
233  03  3
234  02  2
235  03  3
236  07  7
237  69 OP
238  04  04
239  43 RCL
240  03  03
241  69 OP
242  06  06
243  55  ÷
244  43 RCL
245  07  07
246  65  ×
247  02  2
248  22 INV
249  28 LOG
250  95  =
251  42 STO
252  04  04
253  98 ADV
254  18 C'
255  17 B'
256  06  6
257  01  1
258  69 OP
259  04  04
260  43 RCL
261  04  04
262  58 FIX
263  02  02
264  69 OP
265  06  06
266  69 OP
267  00  00
268  98 ADV
269  22 INV
270  58 FIX
271  32 X:T
272  07  7
273  04  4
274  22 INV
275  77 GE
276  04  04
277  66  66
278  10 E'
279  98 ADV
280  43 RCL
281  04  04
282  32 X:T
283  03  3
284  04  4
285  77 GE
286  04  04
287  07  07
288  04  4
289  04  4
290  77 GE
291  04  04
292  05  05
293  05  5
294  04  4
295  77 GE
```

ENTRY POINT DETERMINES VALUE FOR NO OF LAYERS

BRANCH FOR TOTAL OF FACTORS

```
296  04  04
297  03  03
298  06  6
299  04  4
300  77 GE
301  04  04
302  01  01
303  07  7
304  04  4
305  77 GE
306  03  03
307  99  99
308  09  9
309  44 SUM
310  08  08
311  69 OP
312  00  00
313  61 GTO
314  03  03
315  39  39
316  01  1
317  85  +
318  01  1
319  85  +
320  01  1
321  85  +
322  01  1
323  85  +
324  01  1
325  85  +
326  01  1
327  85  +
328  01  1
329  85  +
330  01  1
331  85  +
332  01  1
333  85  +
334  02  2
335  95  =
336  61 GTO
337  01  01
338  36  36
339  43 RCL
340  08  08
341  32 X:T
342  01  1
343  03  3
344  77 GE
345  03  03
346  34  34
347  01  1
348  04  4
349  77 GE
350  03  03
351  32  32
352  01  1
353  06  6
354  77 GE
355  03  03
356  30  30
357  01  1
358  07  7
359  77 GE
360  03  03
361  28  28
362  01  1
363  08  8
364  77 GE
365  03  03
366  26  26
367  01  1
368  09  9
369  77 GE
370  03  03
371  24  24
372  02  2
373  00  0
374  77 GE
375  03  03
376  22  22
377  02  2
378  01  1
379  77 GE
380  03  03
381  20  20
382  02  2
383  02  2
384  77 GE
385  03  03
386  18  18
387  02  2
388  03  3
389  77 GE
390  03  03
391  16  16
392  01  1
393  02  2
394  61 GTO
395  01  01
396  36  36
397  00  0
398  00  0
```

```
PRT CODE REGISTERS

1513413724.      00
3231007373.      01
         0.      02
         0.      03
         0.      04
         0.      05
         0.      06
         0.      07
         0.      08
5151515151.      09
  33431400.      10
1617362422.      11
3100173637.      12
2430133717.      13
3132400033.      14
4335632231.      15
1600332713.      16
3117360071.      17
4136171314.      18
2717001432.      19
1335160013.      20
3517130071.      21
    374533.      22
1700322100.      23
1524351541.      24
2437007100.      25
1624222437.      26
1327006413.      27
1331132732.      28
2200640014.      29
1532301440.      30
1663136415.      31
      2724.      32
3117001617.      33
3136243745.      34
  71000000.      35
    273243.      36
  64001300.      37
    301716.      38
  64001400.      39
  23242223.      40
  64001500.      41
1731371735.      42
  15323033.      43
4000343745.      44
5700356336.      45
  37231731.      46
  21323237.      47
3335243137.      48
5700356336.      49
2132323733.      50
3524313700.      51
3317351517.      52
3137132217.      53
3132400027.      54
1345173536.      55
  33351716.      56
2415371716.      57
         0.      58
         0.      59
```

ENTRY PT FOR ADDITION OF FACTORS

CODE CONVERTER

```
399  01  1
400  85  +
401  01  1
402  85  +
403  01  1
404  85  +
405  03  3
406  85  +
407  02  2
408  95  =
409  44 SUM
410  08  08
411  61 GTO
412  03  03
413  39  39
414  76 LBL
415  10 E'
416  52 EE
417  55  ÷
418  52 EE
419  00  0
420  01  1
421  94 +/-
422  42 STO
423  58  58
424  95  =
425  28 LOG
426  48 EXC
427  58  58
428  85  +
429  93  .
430  01  1
431  85  +
432  59 INT
433  75  -
434  05  5
435  54  )
436  65  ×
437  01  1
438  00  0
```

FOR LINE DENS. FACTOR

WARNING IF > 75%

```
439  00  0
440  49 PRD
441  59  59
442  34 ΓX
443  75  -
444  59 INT
445  44 SUM
446  59  59
447  97 DSZ
448  58  58
449  04  04
450  28  28
451  25 CLR
452  48 EXC
453  59  59
454  92 RTN
455  02  2
456  85  +
457  02  2
458  85  +
459  03  3
460  95  =
461  44 SUM
462  08  08
463  61 GTO
464  01  01
465  96  96
466  01  1
467  94 +/-
468  42 STO
469  02  02
470  19 D'
471  98 ADV
472  18 C'
473  05  5
474  03  3
475  42 STO
476  02  02
477  61 GTO
478  02  02
479  78  78
```

LOAN SCHEDULE,- Lem Matteson, Kansas City, MO. This is one of the most practical loan
------------- schedule programs I have ever seen. It uses Normal Mode for the entry of data and the prelimninary calculations. Then it allows you to use Fast Mode for the print out of the long and slow schedule print out. However, if you prefer life at a leisurely pace, you may continue in Normal Mode and wait twice as long.

Moreover, it computes a payment per period for you, based amount of loan, interest rate and number of pay periods. You may use this payment for computation of the schedule or you may enter anu other payment, more or less than the computed one and it will use <u>your</u> payment proposal as the basis for a schedule. If that isn't "user-friendliness.."

To record your program you will need two mag cards.

1. Put your calculator in 5 OP 17 and key in the program and the data registers.
2. Put your calculator in 6 OP 17 and record bank I on side 1 of a mag card.
   On the same card, record bank III, the print code data registers.
   On the next card, record bank II and bank III again.

To use the program:

1. Read in three card sides in 6 OP 17 partitioning. (the turn-on one)
2. Enter the amount of the loan and press A.
3. Enter the annual percent interest rate and press B.
4. Enter the starting month (1 through 12) and press 2nd A'. If you don't enter here, the program assumes January as the starting month.
5. Enter the number of years of the loan and press C.
   The program computes a monthly payment for you here. If you agree with this one as the basis for the loan schedule computation, go to 7. Otherwise do 6.
6. Enter any other payment amount and press D.
   The display will show, and the printer will print, the number of payments now needed with this new payment. It will be shown in the form of NNN.nn in which NNN is the number of full payments required and .nn is the fractional payment which will be computed as the last payment.
7. Again you will have a choice here. Either you can print the schedule in Normal Mode or in Fast mode. For Normal Mode, press E.
8. For Fast Mode, press 2nd E'. Display stops with a zero. Slide in card side bank I.
   Again the display stops with a zero. Slide in card side bank III, the data.
   See your paper tape zip out of the printer.
   Notice that at year's end the program prints out the amount of interest paid. This is, of course, very handy when you fill out your tax return.
   Notice also that, if you entered a different payment through D, the program computes a partial last payment. Otherwise, it prints a zero payment as the last one.

SEE PROGRAM LISTING ON NEXT PAGE.

ERRATA,- Lem Matteson sends me the following errata.
------ In v5n6p10, at the top of the page, the revised Snow program will work fine the first time around, but fouls up a second time. He found that register 01 did not get cleared and the next print code got added to the previou one. The 100 PRD 01 kept increasing the number when the register was full. Soon, the program got the calculator into EE mode and one digit was lost. The solution was to change the RCL at step 040 to EXC, then the CLR at o39 will clear R01 as it recalls R01 for display. Lem says "it had to be a typo, why would the CLR be there otherwise?"

In v5n9/10p11, the Guard Digits Printer doesn't work correctly. The example, pi, shows 3.141592654 as the display and 5359 as digits 10, 11, 12 and 13. Thse are, of course, digits 9, 19 11 and 12, the 13th being a zero. To make sure that the trailing zero was not the cause of the trouble, he used 1234.123456789 as the test number and, sure enough, it showed 5678 as the last digits, dropping the 9 erroneously.

He knew that FIX wasn't to blaim, because the last 8 didn't get rounded off. So, he checked the EE parts and tried 1 EE 8 instead of the 1 EE 7 at steps 022 to 024. Eureka, it did the trick. So, change the 7 at step 024 to an 8 and all will be well, as intended.

Jules Bell tells me that a few errors have crept into his program DIM.OPS v6n8p9. After the fifth line, add the words "press C."

Steps 126 through 146 should be changed as follows:

```
126: X:T  5 3 X=T  1 4 7  6 3  X=T  2 1 6  5 4  X=T  1 4 7  6 4 X=T  2 5 6
```

Loan Schedule, Lem Matteson, Listing & sample.

```
 56000.00     LOAN
     17.5     %INT
       1.      MTH

      30.      YRS
     360.      NO.
   821.14     PMT

   821.14     PMT
   360.03      NO.

     SCHEDULE

 56000.00     LOAN
     17.5     %INT
   821.14      PMT

       1.      NO.
   816.67      INT
     4.47     PCPL
 55995.53      BAL

       2.      NO.
   816.60      INT
     4.54     PCPL
 55990.99      BAL

       3.      NO.
   816.54      INT
     4.60     PCPL
 55986.39      BAL

       4.      NO.
   816.47      INT
     4.67     PCPL
 55981.72      BAL

       5.      NO.
   816.40      INT
     4.74     PCPL
 55976.98      BAL

       6.      NO.
   816.33      INT
     4.81     PCPL
 55972.17      BAL

       7.      NO.
   816.26      INT
     4.88     PCPL
 55967.29      BAL

       8.      NO.
   816.19      INT
     4.95     PCPL
 55962.34      BAL

       9.      NO.
   816.12      INT
     5.02     PCPL
 55957.32      BAL

      10.      NO.
   816.04      INT
     5.10     PCPL
 55952.22      BAL

      11.      NO.
   815.97      INT
     5.17     PCPL
 55947.05      BAL

      12.      NO.
   815.89      INT
     5.25     PCPL
 55941.80      BAL

   YEARS  INTEREST
  9795.48

      13.      NO.
   815.82      INT
     5.32     PCPL
 55936.48      BAL
```

```
     453526.     30
     303723.     31
    61243137.    32
    27321331.    33
  4517133536.    34
  3732371327.    35
      243137.    36
  1735173637.    37
      361523.    38
  1716412717.    39
          0.     40
          0.     41
          0.     42
     313240.     43
     333037.     44
    13153327.    45
      141327.    46
```

```
000  00  0
001  00  0
002  00  0
003  00  0
004  00  0
005  36 PGM
006  02  02
007  71 SBR
008  02  02
009  39  39
010  00  0
011  00  0
012  00  0
013  00  0
014  00  0
015  00  0
016  00  0
017  25 CLR
018  91 R/S
019  22 INV
020  58 FIX
021  22 INV
022  86 STF
023  02  02
024  05  5
025  69 OP
026  17  17
027  82 HIR
028  16  16
029  42 STO
030  38  38
031  82 HIR
032  17  17
033  42 STO
034  26  26
035  82 HIR
036  12  12
037  42 STO
038  02  02
039  00  0
040  42 STO
041  01  01
042  42 STO
043  23  23
044  42 STO
045  24  24
046  43 RCL
047  33  33
048  69 OP
049  04  04
050  58 FIX
051  02  02
052  82 HIR
053  14  14
054  69 OP
055  06  06
056  22 INV
057  58 FIX
058  43 RCL
059  32  32
060  69 OP
061  04  04
062  43 RCL
063  26  26
064  69 OP
065  06  06
066  43 RCL
067  44  44
068  69 OP
069  04  04
070  58 FIX
071  02  02
072  82 HIR
073  13  13
074  69 OP
075  06  06
076  22 INV
077  58 FIX
078  98 ADV
079  82 HIR
080  15  15
081  59 INT
082  42 STO
083  00  00
084  43 RCL
085  43  43
086  69 OP
087  04  04
088  69 OP
089  21  21
090  43 RCL
091  01  01
092  69 OP
093  06  06
094  43 RCL
095  36  36
096  69 OP
097  04  04
098  82 HIR
099  14  14
100  65  ×
101  43 RCL
102  28  28
103  95  =
104  58 FIX
105  02  02
106  52 EE
107  22 INV
108  52 EE
109  69 OP
```

```
110  06  06
111  22 INV
112  58 FIX
113  44 SUM
114  23  23
115  44 SUM
116  24  24
117  32 X:T
118  43 RCL
119  45  45
120  69 OP
121  04  04
122  87 IFF
123  01  01
124  01  01
125  80  80
126  82 HIR
127  13  13
128  75  -
129  32 X:T
130  95  =
131  58 FIX
132  02  02
133  69 OP
134  06  06
135  22 INV
136  58 FIX
137  82 HIR
138  54  54
139  43 RCL
140  46  46
141  69 OP
142  04  04
143  58 FIX
144  02  02
145  82 HIR
146  14  14
147  69 OP
148  06  06
149  22 INV
150  58 FIX
151  98 ADV
152  87 IFF
153  01  01
154  04  04
155  88  88
156  22 INV
157  97 DSZ
158  02  02
159  04  04
160  88  88
161  97 DSZ
162  00  00
163  00  00
164  84  84
165  86 STF
166  01  01
167  82 HIR
168  14  14
169  32 X:T
170  00  0
171  67 EQ
172  04  04
173  88  88
174  77 GE
175  04  04
176  88  88
177  61 GTO
178  00  00
179  84  84
180  82 HIR
181  14  14
182  85  +
183  69 OP
184  06  06
185  32 X:T
186  95  =
187  82 HIR
188  03  03
189  43 RCL
190  44  44
191  69 OP
192  04  04
193  58 FIX
194  02  02
195  82 HIR
196  13  13
197  69 OP
198  06  06
199  22 INV
200  58 FIX
201  00  0
202  82 HIR
203  04  04
204  61 GTO
205  01  01
206  39  39
207  43 RCL
208  35  35
209  69 OP
210  02  02
211  43 RCL
212  36  36
213  69 OP
214  03  03
215  43 RCL
216  37  37
217  69 OP
218  04  04
```

```
219  69 OP
220  05  05
221  58 FIX
222  02  02
223  43 RCL
224  24  24
225  99 PRT
226  22 INV
227  58 FIX
228  22 INV
229  86 STF
230  00  00
231  06  6
232  69 OP
233  17  17
234  98 ADV
235  98 ADV
236  98 ADV
237  25 CLR
238  91 R/S
239  00  0
240  76 LBL
241  16 A'
242  22 INV
243  58 FIX
244  32 X:T
245  05  5
246  69 OP
247  17  17
248  61 GTO
249  05  05
250  34  34
251  76 LBL
252  15 E
253  86 STF
254  02  02
255  61 GTO
256  10 E'
257  76 LBL
258  11 A
259  42 STO
260  29  29
261  05  5
262  69 OP
263  17  17
264  43 RCL
265  33  33
266  69 OP
267  04  04
268  87 IFF
269  03  03
270  02  02
271  76  76
272  01  1
273  02  2
274  42 STO
275  02  02
276  58 FIX
277  02  02
278  43 RCL
279  29  29
280  69 OP
281  06  06
282  91 R/S
283  76 LBL
284  12 B
285  22 INV
286  58 FIX
287  42 STO
288  26  26
289  55  ÷
290  01  1
291  02  2
292  00  0
293  00  0
294  95  =
295  42 STO
296  28  28
297  43 RCL
298  32  32
299  69 OP
300  04  04
301  43 RCL
302  26  26
303  69 OP
304  06  06
305  91 R/S
306  76 LBL
307  13 C
308  98 ADV
309  32 X:T
310  43 RCL
311  30  30
312  69 OP
313  04  04
314  32 X:T
315  69 OP
316  06  06
317  65  ×
318  01  1
319  02  2
320  95  =
321  58 FIX
322  00  00
323  [illegible]
324  22 INV
325  58 FIX
326  42 STO
327  20  20
```

```
328  43 RCL
329  43  43
330  69 OP
331  04  04
332  43 RCL
333  20  20
334  69 OP
335  06  06
336  59 INT
337  82 HIR
338  05  05
339  43 RCL
340  29  29
341  65  ×
342  53  (
343  43 RCL
344  28  28
345  65  ×
346  53  (
347  01  1
348  75  -
349  53  (
350  01  1
351  85  +
352  43 RCL
353  28  28
354  54  )
355  45 YX
356  82 HIR
357  15  15
358  94 +/-
359  54  )
360  35 1/X
361  95  =
362  58 FIX
363  02  02
364  52 EE
365  22 INV
366  52 EE
367  42 STO
368  27  27
369  43 RCL
370  44  44
371  69 OP
372  04  04
373  43 RCL
374  27  27
375  69 OP
376  06  06
377  91 R/S
378  76 LBL
379  14 D
380  98 ADV
381  42 STO
382  27  27
383  43 RCL
384  44  44
385  69 OP
386  04  04
387  58 FIX
388  02  02
389  43 RCL
390  27  27
391  69 OP
392  06  06
393  22 INV
394  58 FIX
395  01  1
396  85  +
397  43 RCL
398  28  28
399  95  =
400  23 LNX
401  35 1/X
402  65  ×
403  53  (
404  01  1
405  75  -
406  43 RCL
407  29  29
408  65  ×
409  53  (
410  43 RCL
411  28  28
412  55  ÷
413  43 RCL
414  27  27
415  54  )
416  54  )
417  23 LNX
418  94 +/-
419  95  =
420  58 FIX
421  02  02
422  52 EE
423  22 INV
424  52 EE
425  42 STO
426  20  20
427  59 INT
428  82 HIR
429  05  05
430  22 INV
431  58 FIX
432  43 RCL
433  43  43
434  69 OP
435  04  04
436  43 RCL
```

```
437  20  20
438  69 OP
439  06  06
440  91 R/S
441  76 LBL
442  10 E'
443  22 INV
444  58 FIX
445  22 INV
446  86 STF
447  03  03
448  43 RCL
449  02  02
450  82 HIR
451  02  02
452  98 ADV
453  00  0
454  69 OP
455  04  04
456  43 RCL
457  38  38
458  69 OP
459  02  02
460  43 RCL
461  39  39
462  69 OP
463  03  03
464  69 OP
465  05  05
466  98 ADV
467  43 RCL
468  29  29
469  82 HIR
470  04  04
471  43 RCL
472  27  27
473  82 HIR
474  03  03
475  43 RCL
476  28  28
477  82 HIR
478  06  06
479  43 RCL
480  26  26
481  82 HIR
482  07  07
483  87 IFF
484  02  02
485  00  00
486  19  19
487  81 RST
488  43 RCL
489  34  34
490  69 OP
491  02  02
492  43 RCL
493  36  36
494  69 OP
495  03  03
496  43 RCL
497  37  37
498  69 OP
499  04  04
500  69 OP
501  05  05
502  58 FIX
503  02  02
504  43 RCL
505  23  23
506  99 PRT
507  22 INV
508  58 FIX
509  98 ADV
510  87 IFF
511  01  01
512  02  02
513  07  07
514  00  0
515  42 STO
516  23  23
517  01  1
518  02  2
519  42 STO
520  02  02
521  61 GTO
522  01  01
523  61  61
524  86 STF
525  03  03
526  43 RCL
527  31  31
528  69 OP
529  04  04
530  32 X:T
531  42 STO
532  03  03
533  69 OP
534  06  06
535  94 +/-
536  85  +
537  01  1
538  03  3
539  95  =
540  42 STO
541  02  02
542  43 RCL
543  03  03
544  91 R/S
545  25 CLR
546  91 R/S
```

REGISTER OPERATIONS.- Philip Brassine of Seattle, WA wrote this utility program. It will do a variety of operations on blocks of registers, from R02 through R99, provided you partition the calculator correctly.

1. LOAD DATA IN GROUPS OF REGISTERS. Load side 1-B by pressing CLR ans sliding the B side of the card into the slot.
   To load the same data in R(x) to and including R(y), enter x, press E, y, R/S, data, R/S. For example, if you want to store -234 in R40 to R43 and 567 in R45 to R48, enter 40 E 43 R/S 234 +/- R/S and 45 E 48 R/S 567 R/S
   To check if it really works, load side 1-A of the card,(press CLR and slide the A-side of the card into the slot) enter the low register-1 and press A, enter the high register+1 and press R/S. All zero-containing registers will be skipped.
   So, enter 39 A 49 R/S
2. TRANSFER OF A BLOCK OF REGISTERS. First of all, load side 1-B again.
   To transfer a block of registers R(v) to R(w) to a location starting with R(x), enter v, press B, enter w, press R/S, enter x, press R/S
   Here, suppose you want to transfer R42-R46 to a location starting at R32 and R45-R47 to a location starting at R57, enter 42 B 46 R/S 32 R/S and 45 B 47 R/S 57 R/S
   Check with 30 INV LIST.
3. EXCHANGE OF A BLOCK OF REGISTERS WITH ANOTHER ONE. This routine is located on the same 1-B side. To exchange block R(k)-R(l) with a block starting at R(m), enter k, press C, enter l, press R/S, enter m, press R/S.
   For example, you want to exchange R58-R59 with a block starting at R32, enter 58 C 59 R/S 32 R/S. Check again with 30 INV LIST.
4. TRANSFER AND CLR. Routine is on the same 1-B side of the card.
   To transfer a block of registers between R(v) and R(w) to a location starting at R(x) as is done in 2.above, but additionally CLR the block from R(v) through R(w), enter v, press B', enter w, press R/S, enter x, press R/S.
   Suppose you want to transfer and CLR R46-R48 to a location starting at R51, enter 46, B', 48, R/S, 51, R/S. You may check again with 30 INV LIST.
   Or you may load side 1-A again and press 30, A, 59, R/S.
5. SHIFT LEFT OF A BLOCK OF REGISTERS. This routine is located on side 1-A of the card.
   To shift to a location starting with R(r) a block of registers ending with R(q), enter r, press E', enter q, press R/S.
   To check how this works, wipe all your registers clean with CMs. Then store 1 in R41, 2 in R42, etc. up to 9 in R49. You now are going to shift the left this whole block ending in R49 to a location starting with R30. Enter 30, E', 49, R/S.
   Check with 30 INV LIST.
6. CLR A BLOCK OF REGISTERS. This routine is located on side 1-B of the card. Load side 1-B by press CLR and sliding side 1-B into the slot.
   If you want to CLR block R(f) through and including R(g) enter f, press D, enter g, press R/S. Suppose you want to CLR R32 through R35, enter 32, D, 35, R/S.

The TRANSFER routine executes in about 6 to 9 sec, the EXCHANGE routine needs 5 sec, the TRANSFER/CLR routine about 6 sec, the SHIFT LEFT routine in the order of 29 sec and the CLR about .5 sec/reg. The LIST ALL NON-ZERO REGISTERS routine is the slowest with 42 sec to 1 min+.

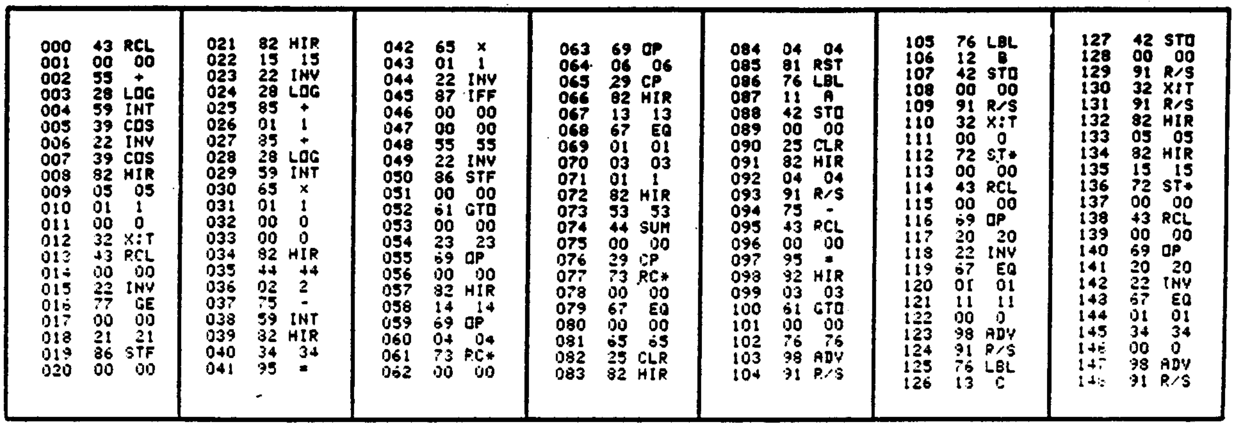
These routines could be good candidates for our own module. Comments invited.

SEE LISTINGS OF BOTH PROGRAMS NEXT PAGE.

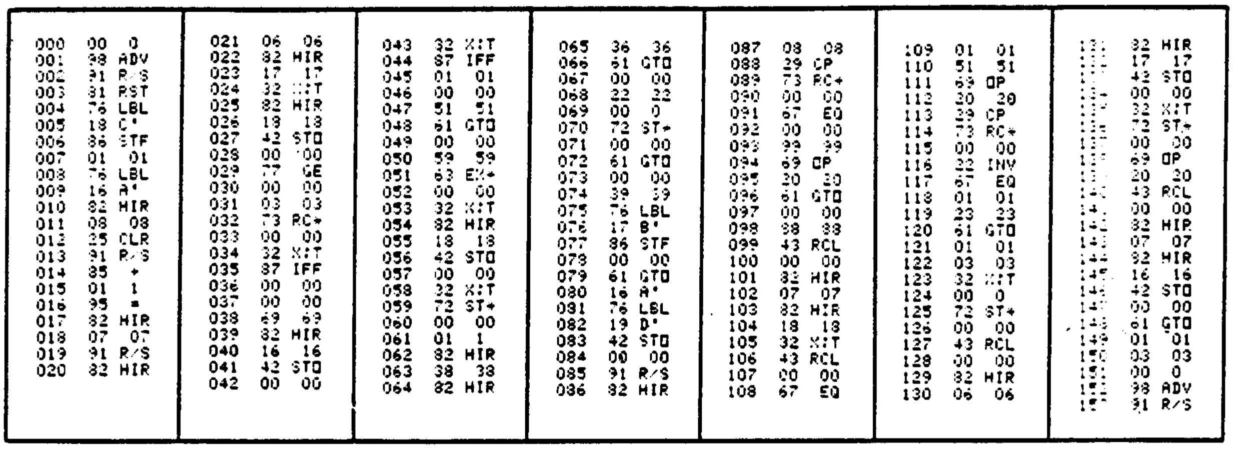
STATISTICS AND MATH LIBRARIES FOR THE SR-52.- Donald Lambert, of Technical Calculator Programs, 434 North Crescent Heights, Los Angeles, CA, 90048, USA, wrote greatly improved stat and math libraries for the SR-52. They either run faster and/or better than the original TI versions. If anyone cares, Don makes them available at $ 25.00 each or $ 40.00 for both of them. Please write to Don directly at the above address.

Don writes that these libraries come with listings, so I suppose they also come with recorded cards, as the original TI libraries did.

REGISTER OPS, Philip Brassine. Program listings. PGM 1-A is at the top, while 1-B is the bottom one.



```
000  43 RCL
001  00  00
002  55  ÷
003  28 LOG
004  59 INT
005  39 COS
006  22 INV
007  39 COS
008  82 HIR
009  05  05
010  01  1
011  00  0
012  32 X:T
013  43 RCL
014  00  00
015  22 INV
016  77  GE
017  00  00
018  21  21
019  86 STF
020  00  00

021  82 HIR
022  15  15
023  22 INV
024  28 LOG
025  85  +
026  01  1
027  85  +
028  28 LOG
029  59 INT
030  65  ×
031  01  1
032  00  0
033  00  0
034  82 HIR
035  44  44
036  02  2
037  75  -
038  59 INT
039  82 HIR
040  34  34
041  95  =

042  65  ×
043  01  1
044  22 INV
045  87 IFF
046  00  00
047  00  00
048  55  55
049  22 INV
050  86 STF
051  00  00
052  61 GTO
053  00  00
054  23  23
055  69 OP
056  00  00
057  82 HIR
058  14  14
059  69 OP
060  04  04
061  73 RC*
062  00  00

063  69 OP
064  06  06
065  29 CP
066  82 HIR
067  13  13
068  67  EQ
069  01  01
070  03  03
071  01  1
072  82 HIR
073  53  53
074  44 SUM
075  00  00
076  29 CP
077  73 RC*
078  00  00
079  67  EQ
080  00  00
081  65  65
082  25 CLR
083  82 HIR

084  04  04
085  81 RST
086  76 LBL
087  11  A
088  42 STO
089  00  00
090  25 CLR
091  82 HIR
092  04  04
093  91 R/S
094  75  -
095  43 RCL
096  00  00
097  95  =
098  92 HIR
099  03  03
100  61 GTO
101  00  00
102  76  76
103  98 ADV
104  91 R/S

105  76 LBL
106  12  B
107  42 STO
108  00  00
109  91 R/S
110  32 X:T
111  00  0
112  72 ST*
113  00  00
114  43 RCL
115  00  00
116  69 OP
117  20  20
118  22 INV
119  67  EQ
120  01  01
121  11  11
122  00  0
123  98 ADV
124  91 R/S
125  76 LBL
126  13  C

127  42 STO
128  00  00
129  91 R/S
130  32 X:T
131  91 R/S
132  82 HIR
133  05  05
134  82 HIR
135  15  15
136  72 ST*
137  00  00
138  43 RCL
139  00  00
140  69 OP
141  20  20
142  22 INV
143  67  EQ
144  01  01
145  34  34
146  00  0
147  98 ADV
148  91 R/S
```



```
000  00  0
001  98 ADV
002  91 R/S
003  81 RST
004  76 LBL
005  18  C'
006  86 STF
007  01  01
008  76 LBL
009  16  A'
010  82 HIR
011  08  08
012  25 CLR
013  91 R/S
014  85  +
015  01  1
016  95  =
017  82 HIR
018  07  07
019  91 R/S
020  82 HIR

021  06  06
022  82 HIR
023  17  17
024  32 X:T
025  82 HIR
026  18  18
027  42 STO
028  00  00
029  77  GE
030  00  00
031  03  03
032  73 RC*
033  00  00
034  32 X:T
035  87 IFF
036  00  00
037  00  00
038  69  69
039  82 HIR
040  16  16
041  42 STO
042  00  00

043  32 X:T
044  87 IFF
045  01  01
046  00  00
047  51  51
048  61 GTO
049  00  00
050  59  59
051  63 EX*
052  00  00
053  32 X:T
054  82 HIR
055  18  18
056  42 STO
057  00  00
058  32 X:T
059  72 ST*
060  00  00
061  01  1
062  82 HIR
063  38  38
064  82 HIR

065  36  36
066  61 GTO
067  00  00
068  22  22
069  00  0
070  72 ST*
071  00  00
072  61 GTO
073  00  00
074  39  39
075  76 LBL
076  17  B'
077  86 STF
078  00  00
079  61 GTO
080  16  A'
081  76 LBL
082  19  D'
083  42 STO
084  00  00
085  91 R/S
086  82 HIR

087  08  08
088  29 CP
089  73 RC*
090  00  00
091  67  EQ
092  00  00
093  99  99
094  69 OP
095  20  20
096  61 GTO
097  00  00
098  88  88
099  43 RCL
100  00  00
101  82 HIR
102  07  07
103  82 HIR
104  18  18
105  32 X:T
106  43 RCL
107  00  00
108  67  EQ

109  01  01
110  51  51
111  69 OP
112  20  20
113  29 CP
114  73 RC*
115  00  00
116  22 INV
117  67  EQ
118  01  01
119  23  23
120  61 GTO
121  01  01
122  03  03
123  32 X:T
124  00  0
125  72 ST*
126  00  00
127  43 RCL
128  00  00
129  82 HIR
130  06  06

131  82 HIR
132  17  17
133  42 STO
134  00  00
135  32 X:T
136  72 ST*
137  00  00
138  69 OP
139  20  20
140  43 RCL
141  00  00
142  82 HIR
143  07  07
144  82 HIR
145  16  16
146  42 STO
147  00  00
148  61 GTO
149  01  01
150  03  03
151  00  0
152  98 ADV
153  91 R/S
```

FAST MODE,- Palmer O. Hanson. The following are some random comments by Palmer on this subject that seems to fascinate a lot of members, judging by the amount of letters I receive dealing with this subject:

" I have expended some time and effort on a compilation of fast mode information both for use in PXX (see the July-August 1981 issue for Palmer's article) and for TI PPC Notes use. (see this issue LIMITATIONS ON THE USE OF THE PGM 02 SBR 239 9 SEQUENCE)

I intend to expand in the near future the list of restrictions and to show more examples of alternate methods of entry, such as the 2 4 STO 00 PGM 02 SBR IND 00 DSZ 99 99 method which leaves you in FIX 9, etc.

I am now reasonably convinced that Patrick Acosta's Fast Mode entry (see v6n8p3and a follow up article in this issue) will manage to do all he has claimed for it:

1. It will work with ANY module. So far I have tested the ML, LE and the Real Estate/ Investment modules.
2. It doesn't destroy memory, so we can go back to normal card reading procedures.
3. It will tie up substantially less memory than the load-and-go technique with the Neef sequence.

If it continues to perform as expected, then we may expect the use of the Neef technique to become obsolete.

To show how limited the Neef technique in reality is with other modules, I have searched all the modules for inadvertent code 31's except the Securities Analysis and Agriculture modules, and have found 49 code 31's. But testing showed that only the ones at ML-02-240 and LE-11-423 provide Fast Mode entry."

SHOPPING LIST,-Jules Bell, Baltimore MD. As promised in last issue, here is Jules' famous ------------- shopping list program. As Jules wants to be known for other programs than simple-minded shopping lists, he thinks "it hardly merits the dignity of the word 'program' but, here it is anyway." I beg to differ. It is a rather nice example how anybody can set up a list of things to do or to remember, be it grocery items, shop inventory, what have you, so long as you are able to identify each item by means of maximum five letters.

Jules advices the following to set up your shopping list:

1. Save your shopping lists for several weeks to determine what you buy most. Identify each item with maximum five letters. If you are trying to decipher Jules' original list, some are easy, but some are a little involved. COTCZ, for example means COTTAGE CHEEZE, PRCHZ is PARMESANO CHEEZE, W/WHT means WHOLE WHEAT (bread) and STTOM, you guessed it, are STEWED TOMATOES.
2. Use the MU module for sorting. You might consider using a program in user memory, but the MU sorting routine will beat them all in speed, simply because it is ROM-based.
3. Press 10 OP 17 and load all your print code. Codes must be entered as a decimal fraction Thus, AAAAA should be stored as " .1313131313 ".
   Do all this with the TI-59 attached to the printer.
4. When finished entering codes, sort by means of 99 STO 00 PGM 06 B. When sorting is complete, record banks 1 through 4 on two mag cards for data back up. Mark those cards with SHOPPING LIST DATA. Press RST to de-access the MU module.
5. Key in your program, still in 10 OP 17 and record four banks on a new set of two mag cards. Press A' and get an alphabetically sorted list of your items printed out.
   Take this set of cards and the TI-59 shopping with you.
   As you go through the aisles in the supermarket, write down a two-digit integer for each item matching your list. The first digit should represent the aisle number, the second digit the position of the item in that aisle, odd digits left and even digits to the right, in sequence. For example, 12 should be the first item to the right in aisle 1.
6. After you have put the groceries away *(members in other cultures might not fully appreciate this, but, viewed from an anthropological standpoint, this is a very important ritual, generally assigned to the non-dominant partner, which in the majority of cases turns out to be the male in contemporary Western society.Ed.)* load the second set of mag cards the ones you took to the supermarket (*and didn't use.Ed)* and SUM the locator code to the appropriate data register. Thus the item name was recorded as ".AAAAA" and the locator code was, say, 12. After SUMming you will have in that register 12.1313131313. If you list this by means of INV LIST, don't be alarmed if you don't see all 12 digits. They are there, but your printer will print only 10 of them. (*unless you use one of our patented 13-digit listers. Ed.)*
7. Now sort again by means of 99 STO 00 PGM 06 B. After sorting press RST A', and you shoulc get a print-out of your items in store location order.
8. If everything looks OK, record yet another set of two mag cards. This will give you:
   A set with supermarket locations on
   A set of "unlocationized" cards (in case you want to go to another supermarket) and
   A back up set.
9. Finally you are ready to use the program. Press A'. This will give you a numbered list of your items. Now, enter the number of each item you want, followed by pressing A; a -1 will be shown in the display each time. When you have indicated all of the items you want for this supermarket trip, enter 0 and press A. Items may be entered in any order, of course. If you want to cancel an item after you have it already entered, enter it again. This will take it out. A third entering will re-instate it, etc.
10. This is a "KAMIKAZI" program. Do not write (record) back to your mag cards after you have executed (run) the program. Your data registers have been altered by your selections. Just enter the program from mag cards each time you want to use (even at the same session).
    This program saves me about 35 to 40 minutes each week in shopping time. At the true value of my leisure time, it has paid off the value of the TI-59/PC100 already three times over.

Maurice, I have given you as much as I can on this. No doubt you'll edit it. *(Yes, I did to the tune of about 50 % of your words.Ed.)* But the <u>spelling</u> is right, s'il vous plait. *(There is no hair on my head doubting your <u>English</u> spelling, but abbreviating BOUILLON as BULLN is rather "odd" in my book. Are you sure you didn't have BULLION on your mind ? But I doubt you will buy this in any supermarket, in any of the two meanings of the word.E*

```
000  76 LBL        044  00  00
001  16 A'         045  53  53
002  00 .0         046  42 STO
003  42 STO        047  00  00
004  00  00        048  01  1
005  09  9         049  94 +/-
006  09  9         050  64 PD*
007  32 X:T        051  00  00
008  69 OP         052  91 R/S
009  20  20        053  00  0
010  43 RCL        054  42 STO
011  00  00        055  00  00
012  99 PRT        056  69 OP
013  73 RC*        057  20  20
014  00  00        058  09  9
015  22 INV        059  09  9
016  59 INT        060  32 X:T
017  65  ×         061  43 RCL
013  01  1         062  00  00
019  00  0         063  77  GE
020  22 INV        064  00  00
021  28 LOG        065  87  87
022  95  =         066  73 RC*
023  69 OP         067  00  00
024  02  02        063  29 CP
025  69 OP         069  77  GE
026  05  05        070  00  00
027  98 ADV        071  56  56
028  43 RCL        072  22 INV
029  00  00        073  59 INT
030  67  EQ        074  65  ×
031  00  00        075  01  1
032  39  39        076  00  0
033  61 GTO        077  22 INV
034  00  00        078  28 LOG
035  08  08        079  95  =
036  98 ADV        030  69 OP
037  98 ADV        081  02  02
038  98 ADV        082  69 OP
039  91 R/S        083  05  05
040  76 LBL        084  61 GTO
041  11  A         085  00  00
042  29 CP         086  56  56
843  67  EQ        087  91 R/S
```

```
          0.    00             .2363163222    50 H/DOG
    0.132727    01 ALL         .2637152333    51 KTCHP
 .1330152346    02 AMCHZ       .2713413116    52 LAUND
 .1413153231    03 BACON       .274144O027    53 LUX L
 .1413222417    04 BAGIE       .3013154715    54 MAC+C
  0.14171721    05 BEEF        .3013352231    55 MARGN
 .1435171316    06 BREAD        0.30134532    56 MAYO
 .1436321333    07 BSOAP        0.30242726    57 MILK
 .1441272731    08 BULLN       .3132321627    58 NOODL
 .1513353237    09 CAROT       .3231243231    59 ONION
 .1517273545    10 CELRY           0.33133    60 PAM
 .1523242631    11 CHIKN       .3324152627    61 PICKL
 .1530172431    12 CMEIN       .3324464613    62 PIZZA
 .1532212117    13 COFFE        0.33323526    63 PORK
 .1532301737    14 COMET       .3335152346    64 PRCHZ
 .1532322645    15 COOKY       .3340331331    65 P.PAN
 .1532371546    16 COTCZ       .3363311333    66 P/NAP
 .1535263536    17 CRKRS       .3363374327    67 P/TWL
 .1540154137    18 C.CUT       .3513243631    68 RAISN
 .1563152631    19 C/CKN        0.35241517    69 RICE
 .1563153531    20 C/CRN       .3532272736    70 ROLLS
 .1563213525    21 C/FRJ          0.354517    71 RYE
 .1563311627    22 C/NDL       .3613271316    72 SALAD
 .1563331713    23 C/PEA       .3613271635    73 SALDR
 .1563333237    24 C/POT       .3613273031    74 SALMN
 .1563363714    25 C/STB       .3613273731    75 SALTN
 .1563372645    26 C/TKY       .3613351331    76 SARAN
 .1563421722    27 C/VEG       .3613411517    77 SAUCE
 .1615173527    28 DCERL       .3615323737    78 SCOTT
 .1617363537    29 DESRT       .3623323537    79 SHORT
 .1630242726    30 DMILK       .3632161336    80 SODAS
  0.17222236    31 EGGS        .3632413336    81 SOUPS
 .2127324135    32 FLOUR       .3633132237    82 SPAGT
 .2127423531    33 FLVRN       .3633241517    83 SPICE
  0.21322427    34 FOIL        .3633323122    84 SPONG
 .2135412437    35 FRUIT      0.363737323    85 STTOM
 .2163142227    36 F/BGL      0.364015353    86 S.CRM
 .2163143515    37 F/BRC       .3640331316    87 S.PAD
 .2163143633    38 F/BSP       .3641221335    88 SUGAR
  0.21632121    39 F/FF        .3643152346    89 SWCHZ
 .2163213623    40 F/FSH       .3643374727    90 SWT+L
 .2163223531    41 F/GRN       .3645354133    91 SYRUP
  0.21633225    42 F/OJ           0.371713    92 TEA
 .2163363331    43 F/SPN       .3730333637    93 TMPST
 .2163363714    44 F/STB        0.37413113    94 TUNA
 .2163363726    45 F/STK       .4217222417    95 VEGIE
 .2163421722    46 F/VEG       .4224312235    96 VINGR
0.223532323     47 GROOM       .4323333237    97 WHPOT
 .2330142235    48 HMBGR       .4363413337    98 W/WHT
 .2335164335    49 HRDWR       .4532223537    99 YOGRT
```

HIDDEN DIGITS VIEWER,- Charlie Williamson of Sacramento, California finally has done what nobody else was able to do: write a true Hidden Digits Viewer. It is really that: It brings <u>only</u> the last three digits to the display, including the sign of the whole number. Furthermore, it brings <u>all three digits</u> to the display, even when some of them are trailing zeroes. For example, with pi his routine brings 590 to the display.

No other published routine accomplishes this, including those in v3n10p4/5 of 52-Notes, nor in v5n2p4, v6n3p7 of the TI PPC Notes, nor any of the many I have seen in Programbiten, Display or other German publications. My hat off to Charlie !

For example, Bill Beebe's routine in v6n3p7 of the TI PPC Notes returns .59265359 in response to pi and you have to count the number of digits to be sure that the hidden ones are NOT 359 but 59 and an assumed zero.

Charlie's routine is much longer than any I have seen, but it accomplishes what is advertised for the other routines, but not delivered by any.

```
000  76 LBL   009  09  9    018  54  )    027  55  ÷    036  53  (    045  55  ÷    053  92 RTN
001  11  A    010  54  )    019  22 INV   028  52 EE    037  29 CP    046  00  0    ___________
002  53  (    011  53  (    020  52 EE    029  53  (    038  85  +    047  22 INV   054  76 LBL
003  24 CE    012  22 INV   021  24 CE    030  52 EE    039  32 X:T   048  52 EE    055  55  ÷
004  55  ÷    013  59 INT   022  91 R/S   031  55  ÷    040  54  )    049  67  EQ   056  55  ÷
005  12  B    014  65  ×    ___________   032  52 EE    041  54  )    050  55  ÷    057  01  1
006  65  ×    015  01  1    023  76 LBL   033  00  0    042  53  (    051  01  1    058  00  0
007  01  1    016  52 EE    024  12  B    034  00  0    043  59 INT   052  54  )    059  54  )
008  52 EE    017  03  3    025  53  (    035  54  )    044  32 X:T                 060  92 RTN
                            026  24 CE                                              ___________
```

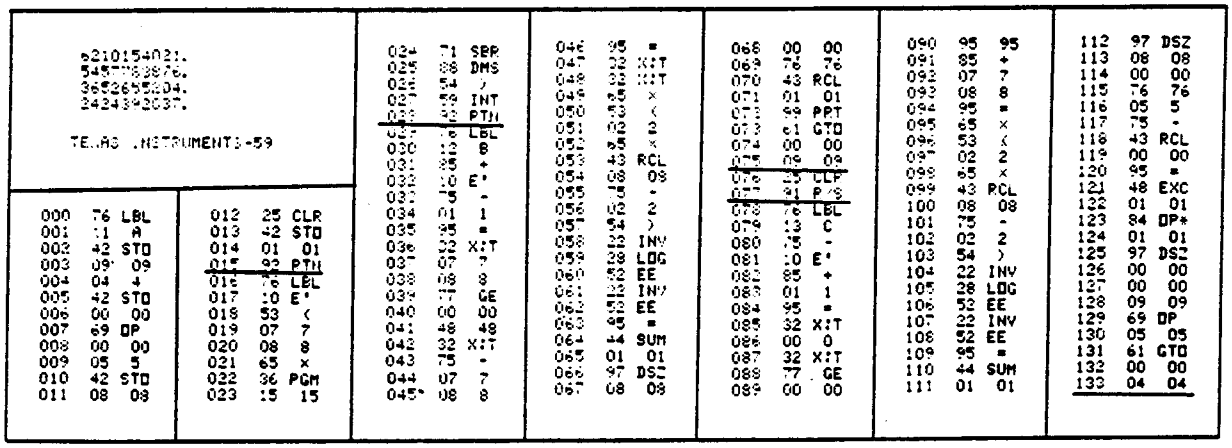
.PTOGRAPHY,- Dejan Ristanović, in Belgrade, Yugoslavia, sent me this one, which he
------------ called "Vizner's Code." Dejan reminds me that Blaise de Vigenère produced a code as early as 1586 with which he enciphered messages. I remember as a boy scout we used that famous table to encode our "secret" communications. This table, if I remember it correctly, is based upon the so-called polyalphabetic system, sometimes also called double-key system, because two or more cipher alphabets are used in the actual encipherment. As it may be, the table has not withstood the onslaught of code breakers, especially not the kind that have a modern computer at their disposal.

According to Dejan, a fellow named Vizner invented a much better code, which Dejan in turn improved upon by using the random generator in the ML library. He now even thinks that it would be virtually impossible to decipher his cryptogram without knowing the key. And the key is simply ANY seed which you and your correspondent know and use for both the encipherment and the decipherment of the message. But then, Dejan must not be "entirely" sure of himself, because he adds wistfully "do you think it is possible?".

I don't know, Dejan. Maybe there are some specialists among our club members who have more knowledge about such things and could enlighten us. I hope they do.

The program itself is easy to use. Be sure, though, that the ML module is in place. To encipher a message, first enter a key, a seed to the random generator. In the example we used 1604, and press A. Then enter letter by letter, in the form of a two-digit print code and follow up each time with B. After five letters the printer will print those first five letters or characters, encoded. Continue with the next five letters, each time followed by B. When you are through, send your friend the encoded message, in the form of your PC100 print outs. And don't forget to phone him and tell him the key or seed.

Your friend now enters that key or seed, 1604 in our case and presses A. Next he or she enters each letter or character from your encoded message, in the form of a two-digit code and presses C this time. When the whole meassage has been entered this way the PC100 prints out the decoded message this time,in the example TEXAS INSTRUMENTS-59.



```
6210154021.
5457763876.
3652655204.
2424392037.

TEXAS INSTRUMENTS-59

000  76 LBL
001  11  A
002  42 STO
003  09  09
004  04  4
005  42 STO
006  00  00
007  69 OP
008  00  00
009  05  5
010  42 STO
011  08  08

012  25 CLR
013  42 STO
014  01  01
015  92 RTN
016  76 LBL
017  10 E'
018  53  (
019  07  7
020  08  8
021  65  x
022  36 PGM
023  15  15

024  71 SBR
025  88 DMS
026  54  )
027  59 INT
028  92 RTN
029  76 LBL
030  12  B
031  85  +
032  10 E'
033  75  -
034  01  1
035  95  =
036  32 X:T
037  07  7
038  08  8
039  77  GE
040  00  00
041  48  48
042  32 X:T
043  75  -
044  07  7
045  08  8

046  95  =
047  32 X:T
048  32 X:T
049  65  x
050  53  (
051  02  2
052  65  x
053  43 RCL
054  08  08
055  75  -
056  02  2
057  54  )
058  22 INV
059  28 LOG
060  52 EE
061  22 INV
062  52 EE
063  95  =
064  44 SUM
065  01  01
066  97 DSZ
067  08  08

068  00  00
069  76  76
070  43 RCL
071  01  01
072  99 PRT
073  61 GTO
074  00  00
075  09  09
076  25 CLR
077  91 R/S
078  76 LBL
079  13  C
080  75  -
081  10 E'
082  85  +
083  01  1
084  95  =
085  32 X:T
086  00  0
087  32 X:T
088  77  GE
089  00  00

090  95  95
091  85  +
092  07  7
093  08  8
094  95  =
095  65  x
096  53  (
097  02  2
098  65  x
099  43 RCL
100  08  08
101  75  -
102  02  2
103  54  )
104  22 INV
105  28 LOG
106  52 EE
107  22 INV
108  52 EE
109  95  =
110  44 SUM
111  01  01

112  97 DSZ
113  08  08
114  00  00
115  76  76
116  05  5
117  75  -
118  43 RCL
119  00  00
120  95  =
121  48 EXC
122  01  01
123  84 OP*
124  01  01
125  97 DSZ
126  00  00
127  00  00
128  09  09
129  69 OP
130  05  05
131  61 GTO
132  00  00
133  04  04
```

?????,- Dave Leising called me the other day and, among other things, dictated a short
---- and sweet program on the phone:

000: OP 04 OP06 1 + RST

Dave claimed he used it all the time and finds it extremely practical. Why don't you try it. You might like it. Just start it with RST R/S.(use the printer, of course)

PERSONAL SCIENTIFIC CALCULATORS,- Jim Mc.Dermott, Special Features Editor, has written
------------------------------ an 11-page article by that name in EDN (Electronics Design News) September 16, 1981, pp 70-81. Jim covers mostly the newer arrivals on the personal calculator front this time: The Radio Shack, Sharp, Casio, Panasonic and Quasar hand-held computers. But he was so nice to tell his readers about our club and our two most interesting discoveries, the Fast Mode and the Graphics Mode. Thanks Jim.

BRANCHING FROM THE KEYBOARD DURING PROGRAM EXECUTION.- In v5n7p11 I talked about this discovery by Martin Neef of the ZEPRA club in West Germany. I also presented Richard Snow's application of this neat trick, in the form of a Time Bombs program. I asked the members at that time to investigate better, and especially longer sequences than PGM 01 SBR 098... PGM 01 BST. The trouble with that routine is that it is so short. Hold the R/S key a little too long and your program stops, without any possiblity of re-starting.

Well, Dejan Ristanović, of Belgrade, Yugoslavia, well-known in these pages, for a variety of programs, has found a better routine: PGM 01 SBR 012...PGM 01 BST.

It is also located in the ML-libray, a definite advantage, as every TI-58/59 owner has one, and it is also much longer and will not change any data register contents, except the first HIR. You still have to use RST, as in Richard's routine, because of the two levels of subroutine used.

Dejan has written a demonstration program to show his technique. It takes the form of one of those memory-reaction games, which he calls 5/10 START. The calculator flashes you five digits in a row and you are supposed to remember those. Then it flashes you one digit only. If it belongs to the first group of five, your decision should be to press R/S. If not, just let it slip by. After each decision (or non-decision) the calculator rewards you with a score up to that point, by flashing you a number in the form A.B in which A is the number of correct reactions so far and B is the number of boo-boos.

Be sure to press RST R/S to re-start the game when the calculator flashes "30" after a decision on your part. Simply R/S will NOT do.

To start the game from the beginning, enter a seed, between zero and 1, say .98765, and press A. After a few seconds you will see the calculator flash 99999....99 That is just to get your attention. Now watch for five single digits, displayed for a short period, one after another, after which the calculator signals the end of that by flashing again 9999...99. Now it is really going to test your memory ability by flashing a single digit. If it belongs to the group of five, press R/S. Otherwise, let it go by.

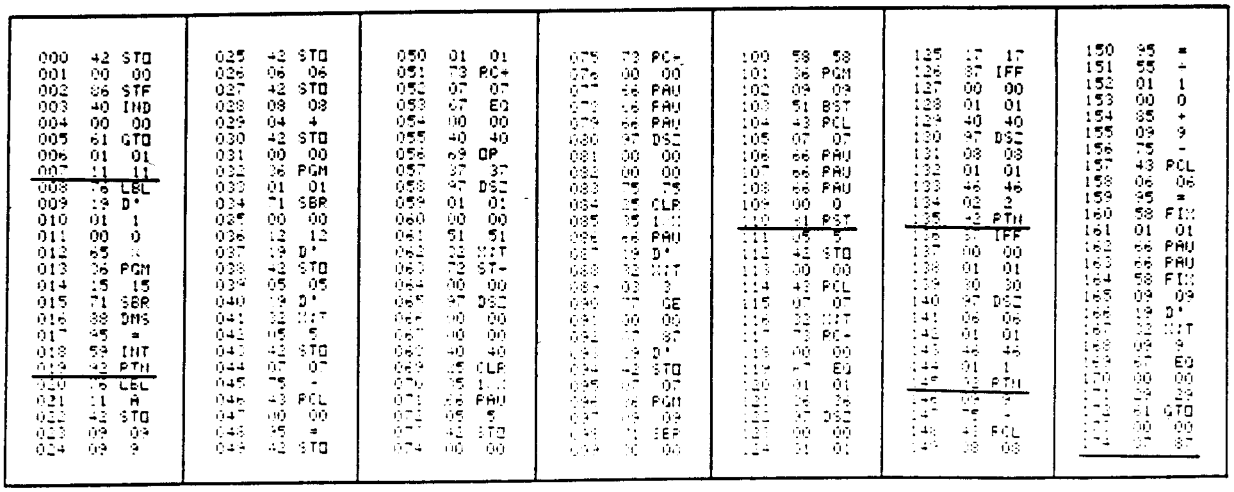
If the calculator flashes 30 (or any other two digit, non fractional number press RST R/S to re-start the game. Normally, at this point the calculator will give you your score so far, as explained above and flash a second digit, etc.

The game is over after you have identified nine digits in a row.

To start all over again, and in case you pressed the wrong key, just enter a new seed and press A.

Note tc newcomers: To enter the BST in step 103, key in STO 51 BST BST DEL SST. Also, if after a while you find the game a little tame, you might make it a little more difficult by overwriting step 108 with a NOP. And after that, maybe 107 too?

Remember that Dejan used direct addressing, so NO DELetes or INSerts, pleas‹ Only NOPs or direct, equal-number steps, overwrites.



```
000  42 STO    025  42 STO    050  01  01    075  73 RC*    100  58  58    125  17  17    150  95  =
001  00  00    026  06  06    051  73 RC*    076  00  00    101  36 PGM    126  87 IFF    151  55  ÷
002  86 STF    027  42 STO    052  07  07    077  66 PAU    102  09  09    127  00  00    152  01  1
003  40 IND    028  08  08    053  67  EQ    078  66 PAU    103  51 BST    128  01  01    153  00  0
004  00  00    029  04  4     054  00  00    079  66 PAU    104  43 RCL    129  40  40    154  85  +
005  61 GTO    030  42 STO    055  40  40    080  97 DSZ    105  07  07    130  97 DSZ    155  09  9
006  01  01    031  00  00    056  69 OP     081  00  00    106  66 PAU    131  08  08    156  75  -
007  11  11    032  36 PGM    057  37  37    082  00  00    107  66 PAU    132  01  01    157  43 RCL
008  76 LBL    033  01  01    058  97 DSZ    083  75  75    108  66 PAU    133  46  46    158  06  06
009  19 D'     034  71 SBR    059  01  01    084  25 CLR    109  00  0     134  02  2     159  95  =
010  01  1     035  00  00    060  00  00    085  35 1/X    110  81 RST    135  92 RTN    160  58 FIX
011  00  0     036  12  12    061  51  51    086  66 PAU    111  05  5     136  87 IFF    161  01  01
012  65  ×     037  19 D'     062  32 X:T    087  19 D'     112  42 STO    137  00  00    162  66 PAU
013  36 PGM    038  42 STO    063  72 ST*    088  32 X:T    113  00  00    138  01  01    163  66 PAU
014  15  15    039  05  05    064  00  00    089  03  3     114  43 RCL    139  30  30    164  58 FIX
015  71 SBR    040  19 D'     065  97 DSZ    090  77  GE    115  07  07    140  97 DSZ    165  09  09
016  88 DMS    041  32 X:T    066  00  00    091  00  00    116  32 X:T    141  06  06    166  19 D'
017  95  =     042  05  5     067  00  00    092  87  87    117  73 RC*    142  01  01    167  32 X:T
018  59 INT    043  42 STO    068  40  40    093  19 D'     118  00  00    143  46  46    168  09  9
019  92 RTN    044  07  07    069  25 CLR    094  42 STO    119  67  EQ    144  01  1     169  67  EQ
020  76 LBL    045  75  -     070  35 1/X    095  07  07    120  01  01    145  92 RTN    170  00  00
021  11  A     046  43 RCL    071  66 PAU    096  36 PGM    121  36  36    146  09  9     171  39  39
022  42 STO    047  00  00    072  05  5     097  09  09    122  97 DSZ    147  75  -     172  61 GTO
023  09  09    048  95  =     073  42 STO    098  71 SBR    123  00  00    148  43 RCL    173  00  00
024  09  9     049  42 STO    074  00  00    099  00  00    124  01  01    149  08  08    174  87  87
```

NEWCOMER'S CORNER. (We are getting tricky this time.) When the TI-58/59 executes a program which we stored in user memory, it doesn't do this step by step. But it rather fetches eight steps at a time, executes them step by step, fetches another eight steps, etc. You might say at this time: "Big deal, so who cares how the calculator does it. I am only interested in the results. I consider the calculator just to be a black box: garbage in, garbage out." Good, but knowing exactly how the calculator does its thing might enable us to take advantage of one or another peculiarity. To be more specific, the above fact that the calculator fetches eight steps at a time and executes them from that special accumulation register rather than from user memory enables us to have the calculator execute a sequence which has already been wiped clean in user memory!

As we all know, when we write the sequence 1 INV WRITE in a program, the calculator suspends execution momentarily and waits for you to slide a recorded card in the slot, to be read into user memory. After you do so, program execution continues.

Now, let us write such a sequence in the <u>first eight steps</u> (in these pages usually referred to as "<u>in the first octet</u>.")

000: LBL A 1 INV WRITE GTO B R/S (sequence 1)

With this type of anotation I mean " starting at step 000, write the following sequence in user memory; that is, in LRN mode."

Now, let us record this program on a magnetic card. Do it manually by pressing 1 2nd WRITE from the keyboard and sliding the card, right side up, in the slot. Next press CP and the calculator is ready to receive this next sequence:

000: LBL B 5 R/S 010: 10 + + R/S (sequence 2)

It is rather clear how to do this: Go into LRN mode at step 000 (press RST if in doubt) and key in the LBL B sequence. Then single-step(SST) to step 010 and continue keying in the rest. Go out of LRN mode. Press 1 2nd WRITE again and record this sequence on the same card, turned over this time.

Turn the calculator off and on again. Press CLR and read in sequence 1 by simply sliding the card in the slot, right side up. Execute LBL A by pressing A. The display goes blank, inviting us to read in side 2. Slide the card in the slot, upside down.

The display stops with a 5 showing. This could only happen if the calculator executed the GTO B in sequence 1 AFTER it had been overwritten by sequence 2. And in fact, that is what happened.

Let us now re-write sequence 1, such that the GTO B part falls in the second octet, that is starting from step 008. (octet one runs from steps 000 through 007). To do this, turn your calculator off and on again, press CLR and slide the mag card in the slot, right side up. Press RST LRN. Press 2nd INS three times. To assure ourselves that we did it right, SST to step 008. We see in fact "61" there, meaning the GTO step is located now at step 008. Press LRN again to go back to keyboard mode. Press A to execute sequence 1. Again the display goes blank, inviting us to read in sequence 2. Again we slide in the mag card, upside down. This time, however, the display shows a flashing 10. This can only mean that, after having read-in the mag card, the program continued execution at step 010 and found there "10 ++ R/S ". The two + signs caused a error, hence the flashing. And the GTO B in sequence 1 was not executed at all this time, because it was not located in the first octet (the only one to be saved) but in octet 2.

Ref. Goeth, Jungbauer & Mueller, Unbekannte Hardwareeigenschaften des TI-59, Display (West Germany) Editor Heinrich Schnepf, N6/7, p14 (1978)

See you next year,

Maurice