T I PPC NOTES

Newsletter of
the TI Programmable Calculator Club.

v7N4/5, 1982

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

It is with great sadness that I have to tell you of the death of Richard Blayney last month in Misíon Viejo, California. Richard is best known for his many contributions to the pages of the SR-52 Notes. He also was the one who came up with the brilliant idea of filling a card with nothing but SST, code 41, commands and use it as a superb diagnostic check for the entire memory of the calculator. Richard lost a tough fight with cancer. The world lost a bright fellow and we all lost a good friend.

Highlights in this issue? Without a doubt, Ralph Snyder takes the cake with his RATE OF RETURN program. It is a superb tool for all you people interested in business and finance.

Then there is SUPERPLOT, a hardware modification you can easily do to your PC100. It permits you to control EVERY SINGLE DOT, be it in an indirect way.

And if you think there isn't anything new under the sun, given that our calculator has been around that long, think again. Just read NEUTRALIZATION. It shows you how you can fool your calculator into believing that you are using, legitimately, a digit as a label. And your calculator believes you!

I was happily surprised the other evening to find in my mail box a complementary copy of L'Ordinateur de Poche (The Pocket Computer) from France. It is a nicely printed magazine à la Popular Science, dealing with the TI-58/59, the HP-41C and all the Japanese made pocket computers. Lots of very interesting news and pictures, but, unfortunately the sophistication of programming, at least from what I can judge from the 59 programs, is not much above high school level. The high point in this issue was an article and program on how to produce factorials. The low point? Why not: a long article on how to record TI-59 programs on a magnetic card. This was prompted by inquieries from two readers. One complained that he had been unable to record programs longer than 239 steps. Any attempt to record bank 2 resulted in a wipe-out of bank 1.

I received many orders for the AF-1. Some of the uses proposed were door stops and boat anchors. Besides telling their members about the availability of the AF-1, the Swedish newsletter Programbiten ran a program and instructions on how to place special hex codes to make the printer print the ä, ö, and å, all letters in the Swedish alphabet , but sadly missing in the PC100 ROM. All the victims got for their trouble executing those most-user-unfriendly instructions was a nice printout of the words APRIL FOOL, followed by your calculator departing to the great digital beyond.

Maurice.

IN THIS ISSUE:

Dear Maurice,

In reference to your newcomers corner in v7n1/2p15:

No, I wouldn't take isue on which coordinates you choose to plot the data although I have little sympathy for shotgun curve fitting. What prompted this letter was your leaving the neophyte with the impression that polynomials may be used for wholesale interpolation. In general they are suitable only for interpolation and may bear no relation to the fitted function beyond the limits over which the fit was performed. In general, the better the fit (and the higher the order) the greater the departure from the fitted function byond the limits of the fit.

Yours Truly,
Elmer B. C.

Dear Maurice,

April fool yourself! Model AF-1 from the Okefenochee swamp, indeed!

I read page 1 and then 7 and 8, bug-eyed. Remembering your words about secret code, I set each digit's triplet of letters vertically under the telephone number, which brought FOOL below 3665. Then that weight of 129 lbs., and all those antiques on page 7 !

With all best regards and be on the watch for banana peels!

Ralph W.S. IN

No comment on that one either.Ed.

Dear Maurice,

I thought you would be interested to know that recently a judge in a Federal Tax Court case ruled that the cost of a hand-held calculator purchased for business purposes by an engineer constituted a capital expenditure rather than a currently deductible business expense because a calculator has a useful life of over one year. The case is known as TC MEMO 1982-42.

Regards,
Ben K.

Dear Maurice,

Perhaps a subject for future discussion in the newsletter would be the speed of certain operations. It would be probably help some of us develop better programming style if we knew more about how long it takes to do various steps.

It would help me also if you occasionally summarize how and when to use new techniques that more active members discover. When the dust settles, I need to know how I can use new ideas in practical programs.

You do a very good job with the newsletter. Hope you stay with it

Sincerely,
Ed. W. IL

Dear Ed,

For your first suggestion, a sequence such as LBL A 1000 STO 00 LBL B PI DSZ 0 B works rather nicely to check execution times of different functions. Suppose you want to check how much time it takes to do a STO. First run the sequence above and note the exact time in seconds it takes. Then insert STO 01 after PI. Now run it again and note the time. The diference between the two execution times divided by 1000 will give you the number of milliseconds execution time for that particular function.

Anybody wants to make up a list? I will publish it.

With respect to your second suggestion: Things often move so fast I hardly have time to absorb it all myself. Palmer Hansen has done several rehashes of recent discoveries, only to discover that something new has been added already while he was writing. But I will try anyway and so will Palmer, I am sure. ED.

Dear Maurice,

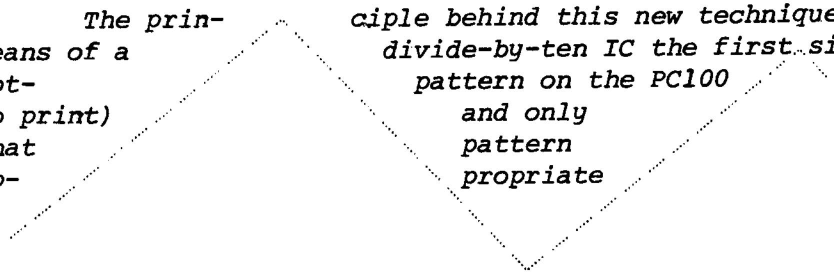
I should really start (this letter) with my satisfaction of being "one of the family"!!! It was really nice to see these drawings from Programbiten too.

Lars Hedlund, Sweden

What Lars is referring to is his picture on the last page of v7n1/2. Sorry that I forgot to mention that all those nice drawings came from the excellent Swedish newsletter Programbiten, of which Lars is the editor. By the way, to the people who asked me what Björn Gustavsson is saying in Swedish, as far as I am able to translate,it means "This is the last time you beat me at NIM."

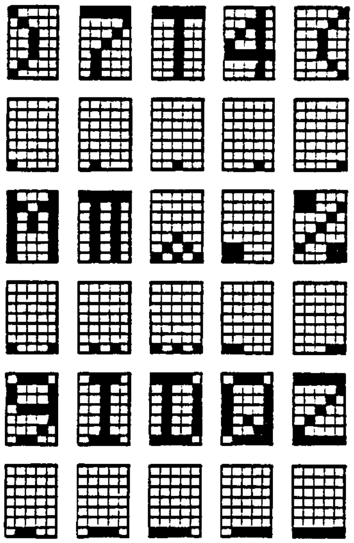
SUPERPLOT,- A month ago Peter Poloczek in Frankfurt, West Germany, sent me a kit to modify the PC100. He obtained it from a Mr. Szpiro, who developed it, somewhere in Switzerland. This hardware modification permits you to have either SUPERPLOT or normal printing/plotting at the flick of a switch. On the bottom of this page is shown a sample of this capability: on the left a pattern in normal printing mode and on the right the resulting equivalent in SUPERPLOT.

And what about the triangular wave here on the top? It took about two minutes to plot that one, with a program of about 50 steps and the plotting subroutine, a program of about 65 steps and 30 registers loaded with print code.



The principle behind this new technique is simple but ingenious: by means of a divide-by-ten IC the first six rows of dots in the seven-row dot-pattern on the PC100 are suspended (not allowed to print) and only the seventh row of dots in that pattern will be printed. By choosing appropriate characters (left parenthesis, 7, T, 4 and right parenthesis) one can print a single dot now in any of the five posible positions.(leftmost, left, center, right and rightmost) How this is done is shown graphically on the left. Observe the top row of characters "(,7,T,4,and)" and see what is printed of them on the row of characters just below them.

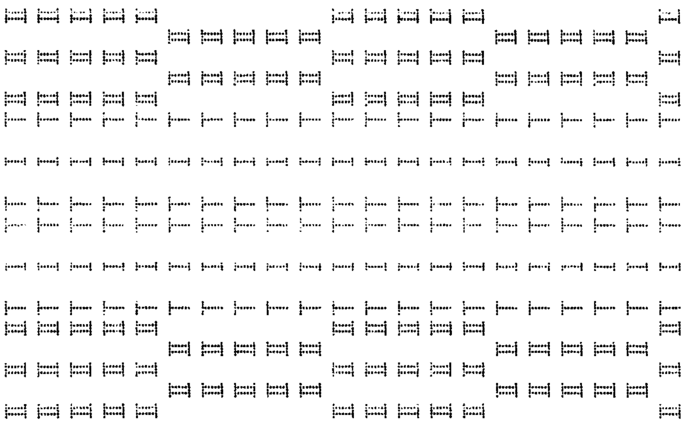
Other possibilities of printing multiple-dot patterns are shown in character rows 3 through 6. The Z character, for example, will result in a solid row of five dots.

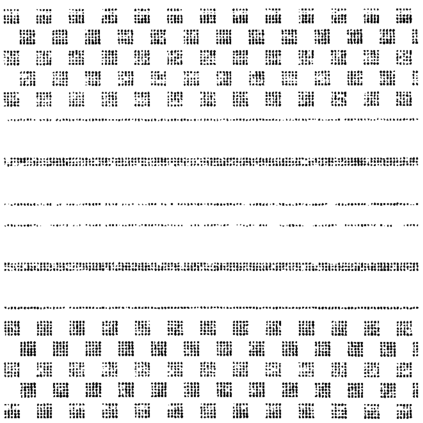


Unfortunately, the kit came with horrendous documentation written in a cauterwelsh German.(I don't mean that German sounds like cauterwelsh to me. On the contrary, German is a beautiful language. But this Szpiro kid knows how to kill a language. Of the three genders in German he used only the Neuter and the four possible declensions he reduced to the simplest common denominator and employed only the Nominativ. The result is the most efficient encipherment of German you have ever encountered.)

Furthermore, the kit came with outdated ICs, holes in the printed circuit that didn't match the solder tabs on the switch (they were off by 1/8th of an inch; 3 mm to Mr. Szpiro.) and so on, such that it would be rather unwise to release this kit, as is, to an unsuspecting membership.

I will make some sort of arrangement with either Peter Poloczek or Mr. Szpiro to have the kit reworked and I will write better instructions for it. Or, if that one falls through, I will reverse-engineer this thing to get the schematic diagram out of it, (I did already, of course) and will publish it, to allow anybody handy with a soldering iron to make their own.





LOAN REPAYMENT, USING DAILY INTEREST WITH NO COMPOUNDING.- Bill Carpenter in Bakersfield, California wrote this program which uses the ML-module. The various label functions are:

LBL B. Enter the date and a statement is written showing the interest due since last receipt.

LBL A.Enters the amount of the payment and also causes LBL A to produce receipt, uodating receipt number, date since last receipt, and total interest paid.

LBL D. Produces full record of payments and dates of payments in chronological order.

User instructions:

1. Enter DATE as MMDD.YYYY. Press B. See accrued interest due and current balance.
2. Enter AMOUNT of payment. Press A. Enter DATE as MMDD.YYYY. Press B. See receipt with complete information.
3. After receipts have been written, press D. See record od DATES and PAYMENTS. The date and amount are stored in one register. (packed)

NOTE: DON'T FORGET TO REWRITE CARDS AFTER ANY RECEIPT WRITING. THIS IN ORDER TO UPDATE RECORDED INFORMATION.

Note also that for each account you will need two mag cards. The name (here MET SWINNEN) is stored in registers 87 through 89.

FIRST TIME INITIALIZATION:(has to be done for each account)

1. Enter 76. Press STO 00.(this is the basis for Count and Records)
2. Enter DATE of note and press 2nd PGM 20 A RST. (This inializes R01 through R04)
3. Press CLR STO 06 STO 07 STO 10.
4. Enter AMOUNT of note and press STO 08.
5. Enter INTEREST rate and press DIV 36500 (alternatively use 36000) = STO 09. This means you are using either 365 or 360 days in a year.

The ML-module is used in this program only for purposes of computing number of days.

KEY IN THE PROGRAM AND LOAD THE ALPHA REGISTERS IN 9 OP 17.
THEN RECORD BOTH CARD SIDES, 1 AND 2, IN 6 OP 17.
SIDES 3 AND 4 OF AN ADDITIONAL CARD ARE USED TO STORE THE DATA YOU ENTER AND THE RESULTS COMPUTED BY THIS PROGRAM.

See the program, alpha listings and sample run on next page.

MODULO 210 SPEEDY FACTOR FINDER.- Re-v6n4/5p13. Palmer Hanson Jr. revised this program by Patrick Acosta and told us we could use it in Normal Mode as well as in Fast Mode. Most of the members didn't need instructions for Normal Mode (so I thought) and therefore I published only the Fast Mode instructions. Since then, several newcomers have asked me how to run the thing in Normal Mode. Palmer Hanson gives the following instructions:

1. With the calculator in the turn-on partitioning, load the four card sides. I am assuming that you recorded the cards in the turn-on partitioning, if you are getting Fast Mode to work already.
2. Press SBR 022. See a zero in the display. This changes the partitioning to 1 OP 17 and positions the program to location 027, ready for entry of an integer to be tested.
3. Enter your integer and press R/S. The solution will proceed the same as in Fast Mode, but at a normal speed. The calculator will stop with a either a 1 or the highest factor in the display. The 1 will not be printed. At this point the program counter will be at step 104.
4. To get ready to test another factor, press R/S. See a zero in the display. The program counter will be at step 025. You may now iterate steps 3 and 4 as needed to test additional numbers.

There are, of course, many other ways to accomplish the same thing. For example, after loading the cards, you could press 1 OP 17 to change to the desired partitioning for running this program. Then you may enter the number to be tested and press SBR 027.

Loan repayment- Bill Carpenter

```
 MET SWINNEN      RECT

          1.      NO.
   101.1982       DATE
          0.      DAYS
       0.00       INT
   10000.00       +BAL
   10000.00       =TOT
       0.00       -PMT
   10000.00       NBAL
       0.00       ΣINT


 MET SWINNEN      STMT

          1.      NO.
   102.1982       DATE
          1.      DAYS
       3.29       INT
   10000.00       +BAL
   10003.29       =TOT
       0.00       -PMT
   10003.29       NBAL
       0.00       ΣINT


 MET SWINNEN      STMT

          1.      NO.
   201.1982       DATE
         31.      DAYS
     101.92       INT
   10000.00       +BAL
   10101.92       =TOT
       0.00       -PMT
   10101.92       NBAL
       0.00       ΣINT


 MET SWINNEN      RECT

          2.      NO.
   301.1982       DATE
         59.      DAYS
     193.97       INT
   10000.00       +BAL
   10193.97       =TOT
    1500.00       -PMT
    8693.97       NBAL
     193.97       ΣINT


 MET SWINNEN      RECT

          3.      NO.
   401.1982       DATE
         31.      DAYS
      88.61       INT
    8693.97       +BAL
    8782.58       =TOT
    1500.00       -PMT
    7282.58       NBAL
     282.58       ΣINT
```

```
        0.00
    101.1982


     1500.00
    301.1980


     1500.00
    401.1980
```

```
000  22 INV
001  58 FIX
002  69 OP
003  00  00
004  25 CLR
005  76 LBL
006  11  A
007  42 STO
008  06  06
009  06  6
010  76 LBL
011  16 A'
012  69 OP
013  17  17
014  25 CLR
015  29 CP
016  98 ADV
017  92 RTN
-----------
018  76 LBL
019  19 D'
020  69 OP
021  04  04
022  32 X:T
023  92 RTN
-----------
024  76 LBL
025  15  E
026  58 FIX
027  02  02
028  52 EE
029  22 INV
030  52 EE
031  76 LBL
032  10 E'
033  69 OP
034  06  06
035  32 X:T
036  22 INV
037  58 FIX
038  92 RTN
-----------
039  76 LBL
040  12  B
041  67  EQ
042  00  00
043  46  46
044  42 STO
045  07  07
046  43 RCL
047  06  06
048  22 INV
049  67  EQ
050  00  00
051  54  54
052  86 STF
053  00  00
054  09  9
055  16 A'
056  43 RCL
057  89  89
058  69 OP
059  01  01
```

```
060  43 RCL
061  88  88
062  69 OP
063  02  02
064  43 RCL
065  87  87
066  69 OP
067  03  03
068  87 IFF
069  00  00
070  00  00
071  79  79
072  43 RCL
073  86  86
074  69 OP
075  30  30
076  61 GTO
077  00  00
078  81  81
-----------
079  43 RCL
080  85  85
081  69 OP
082  04  04
083  69 OP
084  05  05
085  98 ADV
086  43 RCL
087  84  84
088  19 D'
089  07  7
090  06  6
091  75  -
092  43 RCL
093  00  00
094  95  =
095  10 E'
096  43 RCL
097  83  83
098  19 D'
099  43 RCL
100  07  07
101  58 FIX
102  04  04
103  10 E'
104  32 X:T
105  36 PGM
106  20  20
107  12  B
108  43 RCL
109  82  82
110  19 D'
111  36 PGM
112  20  20
113  13  C
114  10 E'
115  43 RCL
116  81  81
117  19 D'
118  65  ×
119  43 RCL
```

```
120  08  08
121  65  ×
122  43 RCL
123  09  09
124  95  =
125  15  E
126  43 RCL
127  80  80
128  19 D'
129  87 IFF
130  00  00
131  01  01
132  35  35
133  44 SUM
134  10  10
135  85  +
136  43 RCL
137  08  08
138  15  E
139  43 RCL
140  79  79
141  19 D'
142  95  =
143  15  E
144  43 RCL
145  78  78
146  19 D'
147  75  -
148  43 RCL
149  06  06
150  15  E
151  43 RCL
152  77  77
153  19 D'
154  95  =
155  15  E
156  32 X:T
157  87 IFF
158  00  00
159  01  01
160  80  80
161  42 STO
162  08  08
163  43 RCL
164  05  05
165  42 STO
166  04  04
167  43 RCL
168  06  06
169  85  +
170  43 RCL
171  07  07
172  52 EE
173  06  6
174  94 +/-
175  95  =
176  22 INV
177  52 EE
178  72 ST*
179  00  00
```

```
180  43 RCL
181  76  76
182  19 D'
183  43 RCL
184  10  10
185  15  E
186  81 RST
-----------
187  76 LBL
188  13  C
189  75  -
190  07  7
191  06  6
192  95  =
193  94 +/-
194  42 STO
195  11  11
196  09  9
197  16 A'
198  73 RC*
199  11  11
200  67  EQ
201  01  01
202  86  86
203  75  -
204  15  E
205  32 X:T
206  95  =
207  58 FIX
208  04  04
209  65  ×
210  01  1
211  52 EE
212  06  6
213  22 INV
214  52 EE
215  95  =
216  99 PRT
217  22 INV
218  58 FIX
219  61 GTO
220  00  00
221  09  09
-----------
222  76 LBL
223  14  D
224  07  7
225  05  5
226  42 STO
227  11  11
228  09  9
229  16 A'
230  71 SBR
231  01  01
232  96  96
233  97 DSZ
234  11  11
235  02  02
236  30  30
237  81 RST
-----------
```

```
   77243137.          76
   31141327.          77
   20333037.          78
   64373237.          79
   47141327.          80
     243137.          81
   16134536.          82
   16133717.          83
     313240.          84
   36373037.          85
   35171537.          86
 3100000000.}         87
 4324313117.}NAME     88
 3017370036.}         89
```

CURVE FITTING BY THE CUBIC SPLINE METHOD.- Robert K. Leaman, Slidell, Louisiana.

Surveyors and draftsmen, please note. If you want smooth curves to fit to a number of points, here is your chance. This program will fit 2 to 9 points tc a spiral. The smooth curves it produces look very natural.

TITLE CUBIC SPLINE CURVE FIT
PROGRAMMER R. K. LEAMAN DATE 1/21/82
Partitioning (Op 17) 4,7,9,5,9 Library Module ANY Printer Option Cards 1

PROGRAM DESCRIPTION

This program fits a cubic spline interpolating curve through 2 to 9 equally spaced points. When more than 9 points or an interval of zero is entered, the program defaults to error. The cubic spline represents the shape of the curve that would be generated if a clock spring were to be threaded through the points. Attempts to project the curve for points beyond the given data points will result in indeterminate data. R/S must not be used after the completion of a LBL routine.

USER INSTRUCTIONS

STEP	PROCEDURE	ENTER	PRESS			DISPLAY
1.	Load number of data points	# of pts.	A			# of pts. *
2.	Load h, the x interval	h	B			h *
3.	Initialize for y data		C			1st data *
	a. Load 1st y data value	1st y	C			2nd data *
	b. " 2nd y data value	2nd y	C			3rd data *
	c. Load 3rd y data value	3rd y	C			4th data *
	Continue to load until y_n	y_n	C			y_nth(stop)*
	LBL C will not accept more than the points loaded at LBL A or 9!					
4.	Calculate cubic spline		D			1.
5.	Load 1st x point	1st x pt.	E'			1st x pt.
6.	Calculate y"	x^*	E			y" *
	Step 6 may be used any number of times.					
7.	Load x interval for sweep output	Δx	A'			Δx
8.	Start sweep		B'			See tape *
	R20 and R21 must remain cleared to zero at all times.					
	*Printed when PC-100A is used.					

USER DEFINED KEYS		DATA REGISTERS (INV List)					
A	# data pts.	0	used	10	used = 0		
B	h	1	x for sweep	11	y_1	21	0
C	y data	2	Δx	12	y_2	22	y2"
D	Cal. spline	3	x_1	13	y_3	23	y3"
E	x to y"	4	# of data pts.	14	y_4	24	y4"
A'	Δx	5	h	15	y_5	25	y5"
B'	Start sweep	6	used	16	y_6	26	y6"
C'		7	index	17	y_7	27	y7"
D'		8	used	18	y_8	28	y8"
E'	first x pt.	9	index	19	y_9	29	y_n"=0

LABELS (Op 08)

INV _ lnx _ CE _ CLR _ x⇌t _ x² _
√x _ 1/x _ STO _ RCL _ SUM _ y^x _
EE _ (_) _ ÷ _ GTO _ × _
SBR _ − _ RST _ + _ R/S _ . _
+/− _ = _ CLR _ INV _ log _ CP _
tan _ Pgm _ P→R _ sin _ cos _ CMs _
Exc _ Prd _ |x| _ Eng _ Fix _ Int _
Deg _ Pause _ x=t _ Nop _ Op _ Rad _
Lbl _ x≥t _ Σ+ _ x̄ _ Grad _ St flg _
If flg _ D.MS _ π _ List _ Write _ Dsz _
Adv _ Prt _

FLAGS 0	1	2	3	4	5	6	7	8	9

CURVE FITTING BY THE CUBIC SPLINE METHOD

The program fits a cubic spline interpolating curve through 2 to 9 equally spaced points. The cubic spline represents the shape of the curve that would be generated if a clock spring were threaded through the data points. This technique is often used by draftsmen to draw a smooth curve through given points. The shape of such a curve looks natural, and is generally the shape one would attempt to draw by hand.

Let the ordinates, y_i, be given at $x_i = x_1 + (i-1)\cdot h$, where i = 1, 2, 3, ..., n, and h is the point spacing. Furthermore, let y(x) be the interpolating curve that is fitted to these points, and let y_i' and y_i'' represent the first and second derivatives of y(x) evaluated at $x = x_i$. y(x) may be represented piecewise where the function and its first and second derivatives are matched at the boundaries. The first and last segments of the interpolating curve may have their first and second derivatives specified by the user. The individual cubic interpolating polynomial $f_i(x)$ can be expressed in terms of the ordinates y_i and y_{i+1}, and either their first or second derivatives. Both forms will provide the same y(x), but the second derivative form requires simpler calculations.

Assume the third derivative, y'"(x), is constant in each interval, h. This assumption implies that y"(x) is linear in x, i.e.,

$$f_i''(x) = y_i''\left\{\frac{x-x_i}{h}\right\} + y_{i+1}''\left\{\frac{x-x_i}{h}\right\} \qquad (1)$$

Equation 1 is integrated twice with respect to x, and the constants of integration chosen so the boundary conditions are met to the extent that $f_i(x_i) = y_i$ (i = 1,2,...,n-1), and $f_{i-1}(x_i) - y_i$ (i = 2, 3,...,n). The results of this integration yield:

$$\begin{aligned} f_i(x) &= y_i(1 - (x-x_i/h) + y_{i+1}(x-x_i/h) \\ &- (h^2/6)(y_i'')\left[1 -(x-x_i)/h - (1 -(x-x_i)/h)^3\right] \\ &- (h^2/6)(y_{i+1}'')\left[(x-x_i)/h - ((x-x_i)/h)^3\right] \end{aligned} \qquad (2)$$

Since the first and second derivatives of the function must also match at the boundaries, Eq. 2 is differentiated with respect to x and evaluated at x_i:

$$f_i'(x_i) = (y_{i+1} - y_i)/h - (h/6)(2y_i'' + y_{i+1}'') \qquad (3)$$

and

$$f_{i-1}'(x_i) = (y_i - y_{i-1})/h + (h/6)(y_{i-1}'' + 2y_i'') \qquad (4)$$

Equating Eqs. 3 and 4 implying boundary match yields:

$$h\cdot y_{i-1}'' = 4h\cdot y_i'' + h\cdot y_{i-1}'' = (6/h)(y_{i-1} - 2y_i + y_{i+1}) \qquad (5)$$

where

i = 2, 3, ... , n-1.

This equation set represents n-2 equations in n unknowns. If the starting and ending second derivatives are specified (y_1'' and y_n''), then the number of unknowns is reduced by 2, and a solution exists to the equation set. This equation set may be expressed in matrix notation:

$$\begin{bmatrix} 4 & 1 & 0 & \ldots\ldots & 0 \\ 1 & 4 & 1 & 0 \ldots\ldots & 0 \\ 0 & 1 & 4 & 1 & 0 \ldots & 0 \\ \cdot & & & & \cdot \\ \cdot & & & & \cdot \\ 0 & \ldots\ldots & 0 & 1 & 4 & 1 \\ 0 & 0 & \ldots & 0 & 0 & 1 & 4 \end{bmatrix} \cdot \begin{bmatrix} y_2'' \\ y_3'' \\ y_4'' \\ \cdot \\ \cdot \\ y_{n-2}'' \\ y_{n-1}'' \end{bmatrix} = \begin{bmatrix} (6/h^2)(y_1 - 2y_2 + y_3) - y_1'' \\ (6/h^2)(y_2 - 2y_3 + y_4) \\ (6/h^2)(y_3 - 2y_4 + y_5) \\ \cdot \\ \cdot \\ \cdot \\ (6/h^2)(y_{n-2} - 2y_{n-1} + y_n) - y_n'' \end{bmatrix} \qquad (6)$$

Because of the tridiagonal characteristic of Eq. 6, a Gauss reduction is an effective method for finding the values of the various second derivatives. Let,

$$d_i = (6/h^2)(y_{i-1} - 2y_i + y_{i+1}) \qquad (7)$$

and select $y_1'' = y_n'' = 0$ (another common selection is $y_1'' = y_2''/2$ and $y_n'' = y_{n-1}''/2$). If a recursion relationship is defined thus:

$$i^4 = 1/(4 - {}_{i-1}{}^4) \text{ for } i = 0, 1, \ldots , n-1 \qquad (8)$$

i.e.,

$$\begin{aligned} 0^4 &= 1/4 = 0.25 \\ 1^4 &= 1/(4 - 0.25) = 0.266666\text{—} \\ 2^4 &= 1/(4 - {}_1{}^4) = 0.267857143 \end{aligned}$$

then the Gauss reduced matrix becomes:

$$\begin{bmatrix} 1/_0{}^4 & 1 & 0 & \ldots\ldots & 0 \\ 0 & (1/_1{}^4) & 1 & \ldots\ldots & 0 \\ 0 & 0 & (1/_2{}^4) & \ldots\ldots & 0 \\ & & & \cdot & \cdot \\ & & & \cdot & \cdot \\ & & & \cdot & \cdot \\ 0 & \ldots\ldots\ldots\ldots & 0 & 0 & (1/_{n-3}{}^4) \end{bmatrix} \cdot \begin{bmatrix} y_2'' \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ y_{n-1}'' \end{bmatrix} = \begin{bmatrix} d_2 \\ d_3 - 0^4\cdot d_2 \\ d_4 - 1^4(d_3 - 0^4\cdot d_2) \\ \cdot \\ \cdot \\ \cdot \\ \cdot \end{bmatrix} \qquad \text{(Eq. 9)}$$

Equation 9 is evaluated by the program.

Example: Fit a cubic spline interpolating curve to the data given in the following table.

x	1	2	3	4	5	6	7	8	9
y	0	5	9	7	4	3	5	8	9

Data is loaded as per the program use instructions.

CUBIC SPLINE FIT, Bob Leaman.

```
000  76  LBL
001  11  A
002  59  INT
003  32  X:T
004  09  9
005  77  GE
006  00  00
007  11  11
008  25  CLR
009  35  1/X
010  92  RTN
011  32  X:T
012  42  STO
013  04  04
014  22  INV
015  86  STF
016  00  00
017  71  SBR
018  00  00
019  22  22
020  99  PRT
021  92  RTN
022  43  RCL
023  04  04
024  85  +
025  02  2
026  00  0
027  95  =
028  42  STO
029  00  00
030  00  0
031  72  ST*
032  00  00
033  43  RCL
034  04  04
035  92  RTN
036  76  LBL
037  12  B
038  29  CP
039  67  EQ
040  00  00
041  08  08
042  42  STO
043  05  05
044  71  SBR
045  00  00
046  22  22
047  43  RCL
048  05  05
049  99  PRT
050  92  RTN
051  76  LBL
052  13  C
053  87  IFF
054  00  00
055  00  00
056  81  81
057  98  ADV
058  86  STF
059  00  00
060  01  1
061  00  0
062  42  STO
063  09  09
064  43  RCL
065  04  04
066  65  ×
067  02  2
068  95  =
069  94  +/-
070  42  STO
071  00  00
072  69  OP
073  29  29
074  43  RCL
075  09  09
076  75  -
077  01  1
078  00  0
079  95  =
080  92  RTN
081  22  INV
082  97  DSZ
083  00  00
084  00  00
085  73  73
086  72  ST*
087  09  09
088  99  PRT
089  22  INV
090  97  DSZ
091  00  00
092  00  00
093  73  73
094  61  GTO
095  00  00
096  71  71
097  76  LBL
098  14  D
099  98  ADV
100  71  SBR
101  00  00
102  22  22
103  75  -
104  01  1
105  95  =
106  42  STO
107  07  07
108  00  0
109  42  STO
110  06  06
111  01  1
112  42  STO
113  00  00
114  69  OP
115  20  20
116  43  RCL
117  00  00
118  85  +
119  09  9
120  95  =
121  42  STO
122  09  09
123  73  RC*
124  09  09
125  75  -
126  53  (
127  02  2
128  65  ×
129  69  OP
130  29  29
131  73  RC*
132  09  09
133  54  )
134  85  +
135  69  OP
136  29  29
137  73  RC*
138  09  09
139  54  )
140  65  ×
141  06  6
142  55  ÷
143  43  RCL
144  05  05
145  33  X²
146  75  -
147  43  RCL
148  06  06
149  95  =
150  42  STO
151  06  06
152  29  CP
153  43  RCL
154  00  00
155  75  -
156  02  2
157  95  =
158  67  EQ
159  01  01
160  76  76
161  42  STO
162  09  09
163  04  4
164  35  1/X
165  94  +/-
166  85  +
167  04  4
168  95  =
169  97  DSZ
170  09  09
171  01  01
172  64  64
173  61  GTO
174  01  01
175  77  77
176  04  4
177  35  1/X
178  42  STO
179  08  08
180  43  RCL
181  00  00
182  32  X:T
183  43  RCL
184  07  07
185  67  EQ
186  01  01
187  99  99
188  43  RCL
189  06  06
190  65  ×
191  43  RCL
192  08  08
193  95  =
194  42  STO
195  06  06
196  61  GTO
197  01  01
198  14  14
199  32  X:T
200  85  +
201  02  2
202  01  1
203  95  =
204  42  STO
205  09  09
206  43  RCL
207  06  06
208  75  -
209  73  RC*
210  09  09
211  54  )
212  65  ×
213  43  RCL
214  08  08
215  95  =
216  48  EXC
217  09  09
218  32  X:T
219  85  +
220  02  2
221  00  0
222  95  =
223  48  EXC
224  09  09
225  72  ST*
226  09  09
227  66  PAU
228  69  OP
229  37  37
230  43  RCL
231  07  07
232  32  X:T
233  01  1
234  22  INV
235  77  GE
236  01  01
237  08  08
238  92  RTN
239  76  LBL
240  10  E'
241  42  STO
242  03  03
243  92  RTN
244  76  LBL
245  15  E
246  32  X:T
247  04  4
248  04  4
249  00  0
250  00  0
251  69  OP
252  04  04
253  32  X:T
254  69  OP
255  06  06
256  75  -
257  43  RCL
258  03  03
259  54  )
260  55  ÷
261  43  RCL
262  05  05
263  95  =
264  42  STO
265  06  06
266  59  INT
267  85  +
268  01  1
269  00  0
270  42  STO
271  09  09
272  01  1
273  95  =
274  44  SUM
275  09  09
276  75  -
277  43  RCL
278  06  06
279  95  =
280  42  STO
281  07  07
282  65  ×
283  73  RC*
284  09  09
285  54  )
286  42  STO
287  08  08
288  43  RCL
289  06  06
290  75  -
291  43  RCL
292  09  09
293  85  +
294  01  1
295  01  1
296  95  =
297  48  EXC
298  08  08
299  85  +
300  69  OP
301  29  29
302  73  RC*
303  09  09
304  65  ×
305  43  RCL
306  08  08
307  95  =
308  42  STO
309  06  06
310  43  RCL
311  08  08
312  75  -
313  24  CE
314  65  ×
315  33  X²
316  54  )
317  65  ×
318  01  1
319  00  0
320  44  SUM
321  09  09
322  73  RC*
323  09  09
324  95  =
325  42  STO
326  08  08
327  43  RCL
328  07  07
329  75  -
330  24  CE
331  65  ×
332  33  X²
333  54  )
334  69  OP
335  39  39
336  65  ×
337  73  RC*
338  09  09
339  95  =
340  42  STO
341  07  07
342  85  +
343  43  RCL
344  08  08
345  54  )
346  55  ÷
347  06  6
348  65  ×
349  43  RCL
350  05  05
351  33  X²
352  54  )
353  94  +/-
354  85  +
355  43  RCL
356  06  06
357  95  =
358  32  X:T
359  04  4
360  05  5
361  06  6
362  05  5
363  69  OP
364  04  04
365  32  X:T
366  69  OP
367  06  06
368  92  RTN
369  76  LBL
370  16  A'
371  42  STO
372  02  02
373  92  RTN
374  76  LBL
375  17  B'
376  98  ADV
377  43  RCL
378  03  03
379  75  -
380  43  RCL
381  02  02
382  95  =
383  42  STO
384  01  01
385  43  RCL
386  02  02
387  44  SUM
388  01  01
389  43  RCL
390  01  01
391  32  X:T
392  43  RCL
393  04  04
394  75  -
395  01  1
396  54  )
397  65  ×
398  43  RCL
399  05  05
400  85  +
401  43  RCL
402  03  03
403  85  +
404  43  RCL
405  02  02
406  95  =
407  32  X:T
408  77  GE
409  04  04
410  16  16
411  15  E
412  98  ADV
413  61  GTO
414  03  03
415  85  85
416  98  ADV
417  98  ADV
418  98  ADV
419  25  CLR
420  92  RTN
```

See sample run on next page.

CUBIC SPLINE FIT, Bob Leaman.

```
          9. # of PTS
          1. = h

          0. y1
          5. y2
          9. y3
          7. y4
          4. y5
          3. y6
          5. y7
          8. y8
          9. y9

        0.2 Δx

Start Sweep
          1.         X
          0.         Y'

        1.2          X
.9737201767          Y'

        1.4          X
1.954010309          Y'

        1.6          X
2.947440353          Y'

        1.8          X
3.960580265          Y'

          2.         X
          5.         Y'

        2.2          X
6.057699558          Y'

        2.4          X
7.067399116          Y'

        2.6          X
7.948248895          Y'
```

```
        2.8          X
8.619399116          Y'

          3.         X
          9.         Y'

        3.2          X
9.035481591          Y'

        3.4          X
8.776393225          Y'

        3.6          X
8.299564065          Y'

        3.8          X
 7.68182327          Y'

          4.         X
          7.         Y'

        4.2          X
 6.32037408          Y'

        4.4          X
5.667027982          Y'

        4.6          X
5.053494845          Y'

        4.8          X
4.493307806          Y'

          5.         X
          4.         Y'

        5.2          X
3.587022091          Y'

        5.4          X
3.267494845          Y'
```

```
        5.6          X
3.054456554          Y'

        5.8          X
2.960945508          Y'

          6.         X
          3.         Y'

        6.2          X
3.179537555          Y'

        6.4          X
3.486992636          Y'

        6.6          X
 3.90467894          Y'

        6.8          X
4.414910162          Y'

          7.         X
          5.         Y'

        7.2          X
5.638827688          Y'

        7.4          X
 6.29653461          Y'

        7.6          X
6.934827688          Y'

        7.8          X
7.515413844          Y'

          8.         X
          8.         Y'

        8.2          X
8.361151694          Y'

        8.4          X
8.614868925          Y'
```

```
        8.6          X
8.788010309          Y'

        8.8          X
8.907434462          Y'

          9.         X
          9.         Y'

Data Memories
after Sweep Run

          2.         00
        9.2          01
        0.2          02
          1.         03
          9.         04
          1.         05
          9.         06
          0.         07
          0.         08
         29.         09
          0.         10
          0.         11
          5.         12
          9.         13
          7.         14
          4.         15
          3.         16
          5.         17
          8.         18
          9.         19
          0.         20
          0.         21
 .8212444772         22
-9.284977909         23
 .3186671576         24
 2.010309278         25
 3.640095729         26
 1.429307806         27
-3.357326951         28
          0. y0''    29
          0.         30
```

SOLAR ENERGY,- A program called PREDICTING SUN OUTAGES appeared in CED. It is written by Raymond Bostick, Jr. If you are unfamiliar with CED, it is the magazine for cable distribution engineers. It is the official organ of the Society of Cable Television Engineers.(SCTE) The publisher is Titsch Publishing Inc. 2500 Curtis Street, P.O.BOX 5400 TA. Denver. Colorado, 80217. Subscription $ 20.00 a year. (US)

The program computes : 1) From prompted entries azimuth and elevation for satellite longitudes at five-degree intervals from 140 West to 70 West.
2) From entry of longitude only, azimuth and elevation for that longitude only.
3) Computes two numbers: the first one is the number of days before the vernal or after the autumnal equinox for peak sun outage. The second number is Greenwich time for the outage.

A well written program with printing and descriptors in the margin. The documentation is complete, with all pertinent equations.

MODULE SWITCHER.- The famous manual and automatic module selectors made by American Microproducts, which were priced at $ 119.95 and $ 199.95 respectively are now available from Systems 7, 1720 West Belt, Houston TX 77043, USA. For information call Mr. Harvey Sperber at 713-468-4394. The prices have been greatly reduced to $ 98.00 and $ 169.00 respectively.

CONVERTING A DECIMAL TO A FRACTION.- Stanley Becker, Long Beach, NY writes that he noticed in v6n8p15 that Bob Fruit didn't like John Worthington's and Emil Regelman's program to do the above. Now, Stan in turn says he did not like either Bob's fractions produced by his extremely fast program. So, Stan wrote a new program that always takes less than ten minutes to produce likable fractions to any degree of accuracy. And furthermore, instead of producing a single fraction to an entered degree of accuracy, it produces a series of fractions, each more accurate than the preceding one, ending with a fraction accurate to 13 digits. Simply enter the number ($\geq$ 0, ≠1) and press A. The number will be printed. Then for each fraction a space, the numerator, the denominator and the difference between the fraction and the entered number will be printed.

The CMS in program location 004 is needed to clear R34, which is needed to produce a correct answer for certian entries such as 0.2 or any other reciprocal of an integer. The 33 in locations 021 and 022 was determined by trial and error to be needed in certain cases for 13-digit accuracy. For example, the ancient Greek' golden ratio of $(\sqrt{5} - 1)/ 2$ is one of the few numbers needing all 33 registers and as much as 10 minutes execution time. Locations 066 through 078 and locations 084 through 096 are identical and thus could have been a subroutine. However, it would have increased execution time, even with direct addressing.

```
000  76  LBL
001  11   A
002  98  ADV
003  99  PRT
004  47  CMS
005  42  STO
006  36   36
007  32  X:T
008  01   1
009  32  X:T
010  22  INV
011  77   GE
012  00   00
013  16   16
014  35  1/X
015  22  INV
016  86  STF
017  00   00
018  42  STO
019  37   37
020  29  CP
021  03   3
022  03   3
023  42  STO
024  00   00
025  42  STO
026  35   35
027  43  RCL
028  37   37
029  22  INV
030  59  INT
031  67   EQ
032  00   00
033  44   44
034  35  1/X
035  42  STO
036  37   37
037  59  INT
038  72  ST*
039  00   00
040  97  DSZ
041  00   00
042  00   00
043  27   27
044  43  RCL
045  35   35
046  98  ADV
047  42  STO
048  00   00
049  03   3
050  03   3
051  32  X:T
052  73  RC*
053  00   00
054  42  STO
055  38   38
056  65   ×
057  69  OP
058  20   20
059  73  RC*
060  00   00
061  85   +
062  01   1
063  95   =
064  42  STO
065  37   37
066  48  EXC
067  00   00
068  77   GE
069  01   01
070  05   05
071  48  EXC
072  00   00
073  69  OP
074  20   20
075  65   ×
076  73  RC*
077  00   00
078  85   +
079  43  RCL
080  38   38
081  95   =
082  42  STO
083  38   38
084  48  EXC
085  00   00
086  77   GE
087  01   01
088  05   05
089  48  EXC
090  00   00
091  69  OP
092  20   20
093  65   ×
094  73  RC*
095  00   00
096  85   +
097  43  RCL
098  37   37
099  95   =
100  42  STO
101  37   37
102  61  GTO
103  00   00
104  66   66
105  43  RCL
106  38   38
107  32  X:T
108  43  RCL
109  37   37
110  22  INV
111  77   GE
112  01   01
113  15   15
114  32  X:T
115  87  IFF
116  00   00
117  01   01
118  20   20
119  32  X:T
120  99  PRT
121  55   ÷
122  32  X:T
123  99  PRT
124  75   -
125  43  RCL
126  36   36
127  95   =
128  99  PRT
129  29  CP
130  67   EQ
131  01   01
132  37   37
133  97  DSZ
134  35   35
135  00   00
136  44   44
137  92  RTN
```

ANALYZE COMPLEX CIRCUITS WITH A MATRIX-INVERSION PROGRAM.- William N. Waggener, Sangoma Weston, EDN, March 17, 1982, pp 131-134. This TI-59 program is very handy when you want to do steady-state analysis of linear circuits. This matrix-inversion program can invert matrices containing as many as 5 X 5 complex elements, allowing you to solve the sets of linear equations that describe such circuits. This well written program uses the ML module.

13-DIGIT PRINTER - Charlie Williamson, Sacramento, CA. This program will print ANY number in any format. The printing is always in EE format, as shown in the examples. The execution time is from 33 to 36 sec, depending on the number. As with Charlie's Guard Digit Printer, this is the first program I encounter that will do exactly as it states it will do in this category. Anybody wants to improve upon it? Only a much shorter and/or much speedier program will do. Instructions? Just enter the number and press A.

For our newcomers: A long number, that is a number with more than 10 digits, is entered as follows: For example you want to enter 1.000000000002 EE -32; first you enter 1 and press STO 00; then you enter 2 EE 12 +/- and press SUM 00; then you press RCL 00; then you say X 1 EE 32 +/- = and you press A. Just wait about half a minute and see your number printed.

```
 2.718281828459 E 00
 3.141592653590 E 00
-3.141592653590 E 00
-3.141592653590 E-99
 9.999999999099 E-99
 9.999999999999 E-99
 1.000000000000 E 13
-1.000000000000 E-13
```

```
000  76 LBL
001  14  D
002  53  (
003  50 I×I
004  85  +
005  01  1
006  85  +
007  53  (
008  24 CE
009  65  ×
010  08  8
011  35 1/X
012  54  )
013  59 INT
014  65  ×
015  02  2
016  54  )
017  92 RTN
018  76 LBL
019  13  C
020  53  (
021  43 RCL
022  00  00
023  65  ×
024  01  1
025  00  0
026  85  +
027  22 INV
028  59 INT
029  42 STO
030  00  00
031  00  0
032  54  )
033  59 INT
```

```
034  92 RTN
035  76 LBL
036  12  B
037  53  (
038  24 CE
039  65  ×
040  01  1
041  00  0
042  00  0
043  85  +
044  13  C
045  14  D
046  54  )
047  92 RTN
048  76 LBL
049  52 EE
050  53  (
051  24 CE
052  55  ÷
053  52 EE
054  53  (
055  52 EE
056  55  ÷
057  52 EE
058  00  0
059  00  0
060  54  )
061  53  (
062  29 CP
063  85  +
064  32 X:T
065  54  )
066  54  )
067  53  (
```

```
068  59 INT
069  32 X:T
070  55  ÷
071  00  0
072  22 INV
073  52 EE
074  67  EQ
075  55  ÷
076  01  1
077  54  )
078  92 RTN
079  76 LBL
080  55  ÷
081  01  1
082  00  0
083  54  )
084  92 RTN
085  76 LBL
086  15  E
087  53  (
088  32 X:T
089  28 LOG
090  52 EE
091  22 INV
092  52 EE
093  55  ÷
094  01  1
095  00  0
096  54  )
097  53  (
098  29 CP
099  85  +
100  32 X:T
101  54  )
102  53  (
103  59 INT
104  14  D
105  65  ×
106  01  1
107  00  0
108  00  0
109  85  +
110  32 X:T
```

```
111  53  (
112  29 CP
113  85  +
114  32 X:T
115  54  )
116  53  (
117  22 INV
118  59 INT
119  65  ×
120  01  1
121  00  0
122  54  )
123  14  D
124  54  )
125  53  (
126  24 CE
127  85  +
128  32 X:T
129  53  (
130  29 CP
131  85  +
132  32 X:T
133  54  )
134  10 E'
135  65  ×
136  01  1
137  00  0
138  00  0
139  00  0
140  00  0
141  85  +
142  01  1
143  07  7
144  65  ×
145  01  1
146  00  0
147  00  0
148  00  0
149  33 X²
150  54  )
151  92 RTN
152  76 LBL
153  10 E'
```

```
154  53  (
155  69 OP
156  10  10
157  65  ×
158  53  (
159  24 CE
160  75  -
161  01  1
162  54  )
163  65  ×
164  01  1
165  00  0
166  54  )
167  92 RTN
138  76 LBL
169  11  A
170  29 CP
171  67  EQ
172  16 A'
173  53  (
174  24 CE
175  55  ÷
176  71 SBR
177  52 EE
178  53  (
179  29 CP
180  85  +
181  32 X:T
182  54  )
183  55  ÷
184  01  1
185  00  0
186  54  )
187  42 STO
188  00  00
189  15  E
190  69 OP
191  04  04
192  43 RCL
193  00  00
194  10 E'
195  12  B
196  53  (
```

```
197  24 CE
198  65  ×
199  01  1
200  00  0
201  00  0
202  85  +
203  04  4
204  00  0
205  54  )
206  12  B
207  12  B
208  69 OP
209  01  01
210  00  0
211  12  B
212  12  B
213  12  B
214  12  B
215  12  B
216  69 OP
217  02  02
218  00  0
219  12  B
220  12  B
221  12  B
222  12  B
223  12  B
224  69 OP
225  03  03
226  69 OP
227  05  05
228  92 RTN
229  76 LBL
230  16 A'
231  69 OP
232  00  00
233  03  3
234  22 INV
235  28 LOG
236  33 X²
237  69 OP
238  01  01
239  92 RTN
```

NAVIGATION MODULE- From several sources in Europe I learned that the famous "Seefahrt (navigation) Modul" by Bobby Schenck is available from:

TISCO, Haggertys Strasse 1, D-8050, FREISING, West Germany.

The module is available with English documentation. Don't underestimate this fact. Translating these highly-technical texts is a nightmare. Unless you are thouroughly familiar AND with both languages AND with the technical aspect (here navigation) you will get nowhere with your two years of high school German.

Please write to the address above for prices and availability.

PROGRAMMING PUZZLE,- Bill Zimmerman, who, by the way, has the distiction of being the very last member on the mailing list, sent me this programming puzzle for which I have been unable to come up with a solution. Bill doesn't have one either, but we hope some of our smarter members will tell us how it can be done.

Part of Bill's work consists of writing programs for the TI-59. One of his programs generates a number between 0 and 99, which in turn has to produce a control digit 0 through 9 according to the table below.

So, Bill simply loaded each control digit in a register, 00 through 99 and recalls the contents (the control digit) by means of an indirect recall scheme, as:

```
LBL E  STO 00    X:T  CLR X=T R/S  RCL IND 00 LBL R/S R/S
```

It works fine, fast, but has the disadvantage that it uses up ALL of your registers.

CAN YOU DEVISE A ROUTINE THAT WILL PRODUCE THE SAME RESULTS, SUCH THAT THE ONE-DIGIT NUMBER MAY BE COMPUTED FROM AN ENTRY OF THE TWO-DIGIT NUMBER, RATHER THAN BE FOUND IN A REGISTER AND RECALLED INDIRECTLY.

DIGIT GENERATION TABLE.

00-0	10-9	20-8	30-7	40-6	50-5	60-4	70-3	80-2	90-1
01-8	11-7	21-6	31-5	41-4	51-3	61-2	71-1	81-0	91-9
02-6	12-5	22-4	32-3	42-2	52-1	62-0	72-9	82-8	92-7
03-4	13-3	23-2	33-1	43-0	53-9	63-8	73-7	83-6	93-5
04-2	14-1	24-0	34-9	44-8	54-7	64-6	74-5	84-4	94-3
05-9	15-8	25-7	35-6	45-5	55-4	65-3	75-2	85-1	95-0
06-7	16-6	26-5	36-4	46-3	56-2	66-1	76-0	86-9	96-8
07-5	17-4	27-3	37-2	47-1	57-0	69-9	77-8	87-7	97-6
08-3	18-2	28-1	38-0	48-9	58-8	68-7	78-6	88-5	98-4
09-1	19-0	29-9	39-8	49-7	59-6	69-5	79-4	89-3	99-2

OP2n-OP3n PECULIARITY.- Michael Sperber, Fuerth, West Germany sent me this one. He says he got the original report on it in the newsletter from the IGPA club (IGPA Trick-Buch, Seite III-30, Absatz n.) But Michael thinks he has an explanation for it.

If you put your calculator in 0 OP 17 partitioning and then press one of the OP2n or OP3n commands, you will see a curious display. To make it more visible, put your calculator also in the EE-mode or better still in the ENG-mode.

Michael tabulated all the OP2n and OP3n results in a table:

OP20	&	OP30	produces	119-12	and in the TI-58	59-12
OP21	&	OP31	p	118-12	a	58-12
OP22	&	OP32	p	117-12	a	57-12
OP23	&	OP33	p	116-12	a	56-12
OP24	&	OP24	p	115-12	a	55-12
OP25	&	OP35	p	114-12	a	54-12
OP26	&	OP36	p	113-12	a	53-12
OP27	&	OP37	p	112-12	a	52-12
OP28	&	OP38	p	111-12	a	51-12
OP29	&	OP39	p	110-12	a	50-12

Michaels says that it is clear that the calculator displays the register number of register n, multiplied by 10 to the power of -12.

For example, says Michael, program steps 952-959 correspond to register 119. (Of course, the calculator does not allows us to access register 119. It reserves that part of memory for program steps only.) The same happens in the TI-58, where program steps 472-479 correspond to register 59.

Does anybody see a practical use for this quirk?

---

100 RANDOM INTEGERS,- Sidney Hack, Columbia, South Carolina. In v6n1p9 we had two programs dealing with that subject: one by Richard Snow and one by Jeff Rosedale. Richard's program ran 2 min and 35 sec in Fast Mode. Sidney's program runs 2 min and 14 sec in Fast Mode.

The program will produce 100, non repeated, integers. The last two numbers, i.e. 99 and 100, take about one minute of the program, in order to randomize them as to position, sequence and to provide for their selection in random order, if one or neither one are hit in the first 98 cycles. So, if we were to be satisfied with 98 random integers, the program would be considerably shorter and simpler. This is what Sidney also has done: besides this program he sent me a simplified one for only 98 integers. In a future issue I might publish that one also, if there is interest for it.

There is one quirk in the program that Sidney hasn't been able to eliminate so far: if one, or neither one of the numbers 99 and 100 appear in the first 98 numbers, the program will provide the proper ending, but will also print a zero after the 100th number.

User Instructions:

1. Record program, through step 178, in power-up partitioning, 6 OP 17.
2. Record on side 1 of a mag card.
3. From turn-on, enter side 1 of the mag card. Press A LRN LRN and enter side 1 again.
4. Enter a seed, any positive number up to 10 to the 9th power, and press R/S.
   The list will be printed.
5. To print a new list, still in Fast Mode, press R/S, enter a seed and press R/S again.

```
000  22 INV
001  58 FIX
002  01  1
003  00  0
004  69 OP
005  17  17
006  43 RCL
007  99  99
008  72 ST*
009  99  99
010  97 DSZ
011  99  99
012  00  00
013  06  06
014  02  2
015  32 X:T
016  66 PAU
017  91 R/S
018  99 PRT
019  98 ADV
020  82 HIR
021  05  05
022  82 HIR
023  15  15
024  65  ×
025  89  π
026  95  =
027  22 INV
028  59 INT
029  65  ×
030  43 RCL
031  98  98
032  95  =
033  82 HIR
034  05  05
035  34 √X
036  22 INV
037  59 INT
038  65  ×
039  43 RCL
040  98  98
041  95  =
042  59 INT
043  75  -
044  82 HIR
045  15  15
046  59 INT
047  42 STO
048  99  99
049  95  =
050  50 I×I
051  22 INV
052  77  GE
053  01  01
054  02  02
055  73 RC*
056  99  99
057  85  +
058  01  1
059  95  =
060  99 PRT
061  97 DSZ
062  98  98
063  00  00
064  68  68
065  61 GTO
066  00  00
067  77  77
068  73 RC*
069  98  98
070  63 EX*
071  99  99
072  63 EX*
073  98  98
074  61 GTO
075  00  00
076  22  22
077  01  1
078  94 +/-
079  72 ST*
080  99  99
081  25 CLR
082  22 INV
083  77  GE
084  00  00
085  22  22
086  98 ADV
087  98 ADV
088  98 ADV
089  66 PAU
090  91 R/S
091  47 CMS
092  09  9
093  09  9
094  42 STO
095  99  99
096  22 INV
097  86 STF
098  01  01
099  61 GTO
100  00  00
101  00  00
102  01  1
103  32 X:T
104  87 IFF
105  01  01
106  01  01
107  53  53
108  86 STF
109  01  01
110  43 RCL
111  99  99
112  85  +
113  43 RCL
114  65  65
115  95  =
116  55  ÷
117  02  2
118  95  =
119  22 INV
120  59 INT
121  85  +
122  01  1
123  95  =
124  67  EQ
125  01  01
126  40  40
127  01  1
128  00  0
129  00  0
130  99 PRT
131  09  9
132  09  9
133  82 HIR
134  08  08
135  01  1
136  32 X:T
137  61 GTO
138  00  00
139  22  22
140  09  9
141  09  9
142  99 PRT
143  01  1
144  00  0
145  00  0
146  82 HIR
147  08  08
148  01  1
149  32 X:T
150  61 GTO
151  00  00
152  22  22
153  82 HIR
154  18  18
155  99 PRT
156  29 CP
157  61 GTO
158  00  00
159  22  22
160  10  E'
161  00  0
162  00  0
163  00  0
164  00  0
165  00  0
166  00  0
167  99 PRT
168  00  0
169  00  0
170  00  0
171  76 LBL
172  11  A
173  36 PGM
174  02  02
175  71 SBR
176  02  02
177  40  40
178  09  9
```

```
   45623.

      87.
      43.
      32.
      46.
      47.
      85.
      68.
      99.
      90.
      41.
       3.
       4.
      31.
      51.
      23.
      84.
      72.
      75.
      45.
      54.
      57.
      83.
      77.
      89.
      38.
      80.
      50.
       •
       •
       •
       •
       •
     100.
      36.
      62.
      37.
      22.
      40.
      30.
      25.
      44.
      10.
      17.
      52.
      95.
      19.
      94.
      79.
      97.
      55.
      81.
      78.
      88.
       8.
      98.
      59.
      60.
       7.
      34.
      66.
      35.
      93.
      33.
      42.
      27.
      29.
      92.
      82.
      53.
      56.
       5.
```

INTERNAL RATE OF RETURN ON UNEQUAL CASH FLOWS- Ralph Snyder, Indianapolis, Indiana.

Here is a program I am giving an unusual amount of space. It would have been even more if I hadn't resorted to some judicious reducing. I hope I don't get too many complaints about it. I had the text tested for legibility by twenty (rather grouchy I must say) men and women over fifty. Only one claimed he couldn't read it at all, which is the sort of percentage I am willing to go with.

Anyone who can spell "Compound Interest" should take a look at this program. Ralph analyzed a similar program in TI's Sourcebook, and by riding off in two directions, he arrived at a remarkably fast rate-finding routine. In one direction he cut the TI valuing and summing loop from 47 to 31 steps, while keeping the basic TI concept,which Ralph rates "excellent." In the other he hauled out his own 1954 paper on an ultra-close direct first-approximation to a rate i for "irregular installment payments"(this was, of course, before "unequal cash flows" had ever been heard of.) The combined results are really impressive: e.g. for a Sourcebook problem with five cash flows, down from 54 sec.(4 runs) to 10.5 sec.(1 run). His program, by the way, churns out answers faster than SA-5 (the Security Analysis module) which does it in Fast Mode, of course.

Ralph says he is on old man (his own words; he claims 82) and after viewing the rate-finding recipes in programs written by so-called "experts" worries that he seems to be the only person around who knows were to find certain goodies published sporadically since around 1900 on special aspects of Compound Interest. Ralph has published some other very good programs on the subject, such as PPX # 198054, Solving All Types of Annuities by the Bond Formula, which is faster than ML-19 in module.

This program is also the best documented one I have seen in a long time, a fact which some of you, younger fellows, will dismiss with a snide "Small wonder, this guy has time on his hands. He is retired, isn't he?" As it may be, I highly recommend you to study this one. It contains the experience of one man working a life-time in the field.

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

This program finds an internal rate of return for a series of variable periodic cash flows, which form annuity-like situations. It also computes a present value (PV) and a future value (FV).

There are two general types of cash flows: Payments occurring at the <u>end</u> of the period, and payments occurring at the <u>beginning</u> of the period.

Definitions of symbols used:

C = general symbol for a series of cash flows $C_1, C_2, C_3, \cdots, C_k$
F = FV (Future Value) of a series of cash flows
P = PV (Present Value) of a series of cash flows
i = periodic interest rate (rate of return) expressed as a decimal
$1+i$ = annuity-due factor, used with payments due at beginning of period
n = general symbol for number of compounding periods 1, 2, 3, ···, k
n = the kth (final) n when used outside the summation sign
v = $(1+i)^{-1}$ = present value of 1 for 1 period
v^n = $(1+i)^{-n}$ = present value of 1 for n periods (n = 1, 2, 3, ···, k)
$\sum C$ = Sum of the C's, that is, of the payments $C_1, C_2, C_3, \cdots, C_k$
$\sum Cn$ = Sum of each C times its number of periods $n_1, n_2, n_3, \cdots, n_k$
$\sum C[n]^2 = \sum Cn(n+1)$
$\sum C[n]^3 = \sum Cn(n+1)(n+2)$
$\sum Cv^n$ = Sum of each C times $v^1, v^2, v^3, \cdots, v^k$

(A) Basic Formulas:

(a) For payments occurring at end of period:

(1) $P = \sum Cv^n$; (2) $F = \left[\sum Cv^n\right](1+i)^n$;

(b) For payments occurring at beginning of period:

(3) $P = \left[\sum Cv^n\right](1+i)$; (4) $F = \left[\sum Cv^n\right](1+i)^{n+1}$.

(B) Formulas for finding a rate of return i:

(a) First approximation:

$$(5)\quad i = \left[-\frac{\sum C[n]^2}{2\sum Cn} + \frac{\sum Cn}{\sum C - P} - i_0\left(\left[\frac{\sum C[n]^2}{2\sum Cn}\right]^2 - \frac{\sum C[n]^3}{6\sum Cn}\right)\right]^{-1},$$

$$\text{where } i_0 = \left[-\frac{\sum C[n]^2}{2\sum Cn} + \frac{\sum Cn}{\sum C - P}\right]^{-1}.$$

(b) Newton h-correction:

$$(6)\quad h = \frac{-F(i)}{F'(i)} = \frac{\sum Cvn - P}{\sum Cnv^{n+1}} = \frac{\sum Cv^n - P}{v\sum Cnv^n}.$$

<u>Accuracy of the rate formulas:</u>

The program calls for an iteration control for h of ‹<.0001, based on a rule of thumb which says that an h-correction should be good for about twice as many correct places as it has leading zeros. The h-correction used here provides a minimum of 7 or 8 correct places, ordinarily sufficient for internal rate of return problems.

For visible evidence of the extent of accuracy of a computed i, the user may replace 2nd Nop at Steps 064 and 097 by 2nd Prt or 2nd Pause. The printout or display on Pause will show the number of leading zeros for each correction.

Pressing SBR SBR will produce an additional h-correction following the normal run, if desired. This may refine i; but not always, depending on accuracy before the additional run, due to rounding errors in the guard digits.

<u>Running time for rate finding:</u>

The time to process one iteration in finding a rate of return varies with the number of cash flows, but the number of iterations is affected by the closeness of the first approximation, which in turn is affected by the magnitude of the actual rate. A few examples give an idea of run times:

Rate found	Number of Cash Flows	Number of Iterations	Running Time
12.9899%	4	1	9½ sec.
7.0000	5	1	10½
14.7548	8	2	29
14.4722	10	2	35
1.0000	12	1	20

These times are substantially better than by the Sourcebook program (see Reference 2), or by the Securities Analysis Module SA-05 "Uneven Cash Flows" program even in the module ("Fast Mode") which is faster than general program mode by nearly 50 percent to begin with.

<u>References:</u>

1. R. W. Snyder, "A Yield Formula for Irregular Installment Payments", The Accounting Review, July, 1954, pp. 457-464. The principal formula (4) derived there is the formula (5) of the present program.

2. Texas Instruments Incorporated, Sourcebook for Programmable Calculators, 1978, Section 6, pp. 6-5 to 6-11, "Variable Periodic Cash Flows and Internal Rate of Return".

<u>Note on derivation of first approximation to i:</u>

Suppose future payments $C_1, C_2, C_3, \cdots, C_k$, due at intervals $n_1, n_2, n_3, \cdots, n_k$. Then the present value of each payment is

$$P_1 = C_1v^1,\quad P_2 = C_2v^2,\quad P_3 = C_3v^3,\quad \cdots,\quad P_k = C_kv^k,$$

and the sum of all the P's is the present value of all the payments. By expanding $v^1, v^2, v^3, \ldots, v^k$, we have:

$$P_1 = C_1 - C_1n_1i + \frac{C_1n_1(n_1+1)}{2}i^2 - \frac{C_1n_1(n_1+1)(n_1+2)}{6}i^3 + \cdots$$

$$P_2 = C_2 - C_2n_2i + \frac{C_2n_2(n_2+1)}{2}i^2 - \frac{C_2n_2(n_2+1)(n_2+2)}{6}i^3 + \cdots$$

$$P_3 = C_3 - C_3n_3i + \frac{C_3n_3(n_3+1)}{2}i^2 - \frac{C_3n_3(n_3+1)(n_3+2)}{6}i^3 + \cdots$$

$$\cdot\quad\cdot\quad\cdot\quad\cdot\quad\cdot\quad\cdot\quad\cdot\quad\cdot\quad\cdot\quad\cdot\quad\cdot\quad\cdot$$

$$P_k = C_k - C_kn_ki + \frac{C_kn_k(n_k+1)}{2}i^2 - \frac{C_kn_k(n_k+1)(n_k+2)}{6}i^3 + \cdots.$$

Adding the terms of the foregoing set of series vertically, we arrive at the following general series for the present value P of all the future payments:

$$P = \sum C - i\sum Cn + i^2\frac{\sum C[n]^2}{2} - i^3\frac{\sum C[n]^3}{6} + \cdots.$$

Dividing by $i\sum Cn$, and dropping terms containing i^3 and higher powers of i after the division,we have

$$\frac{\sum C - P}{i\sum Cn} = 1 - i\frac{\sum C[n]^2}{2\sum Cn} + i^2\frac{\sum C[n]^3}{6\sum Cn}.$$

Inverting both members of this equation, and making the actual division of the right member into 1, we reach what may be called an "indirect" formula:

$$\frac{i\sum Cn}{\sum C - P} = 1 + i\frac{\sum C[n]^2}{2\sum Cn} + i^2\left(\left[\frac{\sum C[n]^2}{2\sum Cn}\right]^2 - \frac{\sum C[n]^3}{6\sum Cn}\right),$$

which we solve for i, following the principle of replacing a small unknown by its approximate value:

$$i = \left[\frac{\sum Cn}{\sum C - P} - \frac{\sum C[n]^2}{2\sum Cn} - i_0\left(\left[\frac{\sum C[n]^2}{2\sum Cn}\right]^2 - \frac{\sum C[n]^3}{6\sum Cn}\right)\right]^{-1},$$

$$\text{where } i_0 = \left[\frac{\sum Cn}{\sum C - P} - \frac{\sum C[n]^2}{2\sum Cn}\right]^{-1}.$$

(Adapted from the development in Reference 1.)

INTERNAL RATE OF RETURN....Ralph Snyder.(cont.)

User Instructions

Program Title INTERNAL RATE OF RETURN

%i	(1)	%i→PV**	%i→FV**	(1)
Pmts/FV	PV	%i→PV*	%i→FV*	INIT

Partition (OP 17) 479.59 Parentheses Levels 1 1 Register ■

Angular Mode (if applicable) SBR Levels 2 Absolute Addresses ☐

Library Module ID None Disturbs Pending Operations ■

*Pmts End of Per. **Pmts Beginning of Per.

USER DEFINED KEYS

A Enter Cash Flows
B Enter PV
C Enter %i to find PV*
D Enter %i to find FV*
E Initializer
A' Find %i
B' (1)
C' Enter %i to find PV**
D' Enter %i to find FV**
E' (1)

(1)Not called from keyboard

DATA REGISTERS

0	0	Dsz counter
	1	1 to n
	2	2 to n+1
	3	3 to n+2
	4	$\sum Cv^n$
	5	$\sum Cnv^n$
	6	Reg. nos. of C's
	7	P
	8	1+i
	9	$\sum C - P$
1	0	$\sum Cn$
	1	$(\sum C[n]^2)/2$
	2	$(\sum C[n]^3)/6$
1	3	successive C's as required
	4	
	5	
	6	
	7	
	8	
7	9	
	0	Repartition if needed
	1	
	2	
	3	
	4	
	5	
	6	
	7	
	8	
	9	
	0	
	1	
	2	
	3	
	4	

FLAGS 0	1	2	3	4	5	6	7	8	9

STEP	PROCEDURE	ENTER	PRESS			OUTPUT/MODE (see legend below)
	13 data registers (00-12) are used for data other than C's (cash flows). Power-up partition allows 47 registers (13-59) for C's. If more registers are needed for C's, repartition 8 OP 17 to 319.79. See Continuation Sheet					

Modes: x* — Printed only (x) — Displayed briefly (Pause) (x)* — Printed and displayed

See Continuation Sheet

User Instructions—continued

STEP	PROCEDURE	ENTER	PRESS			OUTPUT
	Enter program					
(A)	<u>For Payments at End of Period</u>					
1	To find PV or FV:					
a	Initialize		E			0.
b	Enter cash flows 1 through n	C_1	A			1. C_1
		...				...
		...				...
		C_n	A			n C_n
c	Enter %i and find PV	%i	C			P
	or					
d	Enter %i and find FV	%i	D			F
	To find internal rate of return:					
2	PV known:					
a	Initialize		E			0.
b	Enter cash flows 1 through n	C_1	A			1. C_1
		...				...
		...				...
		C_n	A			n C_n
c	Enter PV	P	B			P
d	Find rate of return %i		2nd	A'		%i
3	FV known:					
a	Initialize		E			0.
b	Enter cash flows 1 through n-1	C_1	A			1. C_1
		...				...
		...				...
		C_{n-1}	A			n-1 C_{n-1}
c	Enter $C_n - F$ as last cash flow	C_n	−			
		F	=	A		n C_n (net)
d	Find rate of return %i		2nd	A'		%i

NOTES: 1. 13+N registers are required where N is number of cash flows. Repartition if necessary to 8 OP 17 (319.79).
2. If any C is zero, enter its value as 0; if negative, press +/-.

See Continuation Sheet

STEP	PROCEDURE	ENTER	PRESS			OUTPUT
(B)	<u>For Payments at Beginning of Period</u>					
1	To find PV or FV:					
a	Initialize		E			0.
b	Enter cash flows 1 through n	C_1	A			1. C_1
		...				...
		...				...
		C_n	A			n C_n
c	Enter %i and find PV	%i	2nd	C'		P
	or					
d	Enter %i and find FV	%i	2nd	D'		F
	To find internal rate of return:					
2	PV known:					
a	Initialize		E			0.
b	Enter cash flows 2 through n	C_2	A			1. C_2
		...				...
		...				...
		C_n	A			n-1 C_n
c	Enter $P - C_1$ as net investment	P	−			
		C_1	=	B		P (net)
d	Find rate of return %i		2nd	A'		%i
3	FV known:					
a	Initialize		E			0.
b	Enter cash flows 1 through n	C_1	A			1. C_1
		...				...
		...				...
		C_n	A			n C_n
c	Enter FV as negative cash flow	F	+/-	A		-F
d	Find rate of return %i		2nd	A'		%i

NOTES: 1. 13+N registers are required where N is number of cash flows. Repartition if necessary to 8 OP 17 (319.79).
2. If any C is zero, enter its value as 0; if negative, press +/-.

INTERNAL RATE OF RETURN....Ralph Snyder.(cont.)

Sample Problems

Statement of Example

An investor is considering putting $750,000 into a venture for which the cash inflows over the next 8 years are estimated to be:
1 of $165,000; 1 of $195,000; next 3 of $191,000; final 3 of $97,500, with a residual value of $40,000 at the end of the 8th year.

1. Find the project's rate of return, assuming cash flows occur (a) at the end of the year, (b) at the beginning of the year.

2. If the investor demands a return of 15 percent, (c) how much would he be willing to invest? Or, putting in $750,000, (d) how much residual value must there be? Consider payments at end of year for (c) and (d).

For the rate solutions, PRT at Steps 064 and 097 is used, which shows 1st i and the h-corrections.

ENTER	PRESS	OUTPUT/MODE (see legend below)	COMMENT
		(a) Cash flows end of period	
	E	0.	Initialize
165000	A	1.*	n_1
		165000.00*	C_1
195000	A	2.*	n_2
		195000.00*	C_2
191000	A	3.*	n_3
		191000.00*	C_3
	A	4.*	n_4
		191000.00*	C_4
	A	5.*	n_5
		191000.00*	C_5
97500	A	6.*	n_6
		97500.00*	C_6
	A	7.*	n_7
		97500.00*	C_7
	+	97500.00	C_8
40000	= A	8.*	n_8
		137500.00*	C_8 + FV
750000	B	750000.00*	PV
	2nd A'	.1481723975*	1st i
		-.0006250041*	h_1
		.0000009674*	h_2
		14.7548* (Time: 29 sec.	%i
	INV 2nd Fix	14.75483608 w/o printout)	%i extended

ENTER	PRESS	OUTPUT/MODE (see legend below)	COMMENT
		Alternate solution for (a)	
	E	0.	Initialize
750000	+/- A	1.*	n_1
		-750000.00*	PV entered as -C
165000	A	2.*	n_2
		165000.00*	C_1
195000	A	3.*	n_3
		195000.00*	C_2
191000	A	4.*	n_4
		191000.00*	C_3
	A	5.*	n_5
		191000.00*	C_4
	A	6.*	n_6
		191000.00*	C_5
97500	A	7.*	n_7
		97500.00*	C_6
	A	8.*	n_8
		97500.00*	C_7
137500	A	9.*	n_9
		137500.00*	C_8 + FV
	2nd A'	.1475641288*	1st i
		-.0000157688*	h_1
		14.7548* (Time: 17 sec.	%i
	INV 2nd Fix	14.75483599 w/o printout)	%i extended
(Alternative method does not always result in fewer h-corrections and shorter run time)			
		(b) Cash flows beginning of period	
	E	0.	Initialize
195000	A	1.*	n_1
		195000.00*	C_2
191000	A	2.*	n_2
		191000.00*	C_3
	A	3.*	n_3
		191000.00*	C_4
	A	4.*	n_4
		191000.00*	C_5
97500	A	5.*	n_5
		97500.00*	C_6
	A	6.*	n_6
		97500.00*	C_7
	A	7.*	n_7
		97500.00*	C_8
40000	A	8.*	n_8
		40000.00*	FV
750000	-	750000.00	PV
165000	= B	585000.00*	PV - C_1
	2nd A'	.2166696119*	1st i
		-.0020288305*	h_1
		.0000084328*	h_2
		21.4649* (Time: 29 sec.	%i
	INV 2nd Fix	21.46492142 w/o printout)	%i extended

Sample Problems—continued

ENTER	PRESS	OUTPUT/MODE (see legend below)	COMMENT
		(c) Find PV (payments end of period)	
	E	0.	Initialize
165000	A	1.*	n_1
		165000.00*	C_1
195000	A	2.*	n_2
		195000.00*	C_2
191000	A	3.*	n_3
		191000.00*	C_3
	A	4.*	n_4
		191000.00*	C_4
	A	5.*	n_5
		191000.00*	C_5
97500	A	6.*	n_6
		97500.00*	C_6
	A	7.*	n_7
		97500.00*	C_7
	+	97500.00	C_8
40000	= A	8.*	n_8
		137500.00*	C_8 + FV
15	C	15.00*	%i
		744432.30* (Time: 14 sec.)	PV, to yield 15%
		(d) Find FV (payments end of period)	
	E	0.	Initialize
750000	+/- A	1.*	n_1
		-750000.00*	PV entered as -C
165000	A	2.*	n_2
		165000.00*	C_1
195000	A	3.*	n_3
		195000.00*	C_2
191000	A	4.*	n_4
		191000.00*	C_3
	A	5.*	n_5
		191000.00*	C_4
	A	6.*	n_6
		191000.00*	C_5
97500	A	7.*	n_7
		97500.00*	C_6
	A	8.*	n_8
		97500.00*	C_7
	A	9.*	n_9
		97500.00*	C_8
15	D	15.00*	%i
		57031.73* (Time: 16 sec.)	FV, to yield 15%

Statement of Example

(A) The X Company will sell a machine for $50,000, or lease it for 5 years, at scheduled payments of: $14,500; $13,500; $12,500; $11,000; $10,000, due at the beginning of each year. Find the rate of return from leasing the machine.

(B) Deposits are made in an investment account at the end of each year for 5 years as follows: $500; $800; $1,000; $1,200; $1,500, and the balance at the end of the 5th year is $5,564.33. Find the rate of return.

Note: Times for finding rate of return by three programs:

	(A)	(B)
This program	9½ sec.	10½ sec.
Sourcebook	45	54
SA-05—in module	15	25
—downloaded	31	51

ENTER	PRESS	OUTPUT/MODE (see legend below)	COMMENT
		(A)	
	E	0.	Initialize
13500	A	1.*	n_1
		13500.00*	C_2
12500	A	2.*	n_2
		12500.00*	C_3
11000	A	3.*	n_3
		11000.00*	C_4
10000	A	4.*	n_4
		10000.00*	C_5
35500	B	35500.00* (=50000 - 14500)	PV - C_1
	2nd A'	12.9899* (Time: 9½ sec.)	%i
		(B)	
	E	0.	Initialize
500	A	1.*	n_1
		500.00*	C_1
800	A	2.*	n_2
		800.00*	C_2
1000	A	3.*	n_3
		1000.00*	C_3
1200	A	4.*	n_4
		1200.00*	C_4
4064.33	+/- A	5.*	n_5
		-4064.33* (=1500 - 5564.33)	C_5 - FV
	2nd A'	7.0000* (Time: 10½ sec.)	%i

INTERNAL RATE OF RETURN.... Ralph Snyder.(cont.)

Internal Rate of Return on Unequal Cash Flows—listing

LOC	CODE	KEY	COMMENTS
000	91	R/S	
001	76	LBL	**Valuing and summing loop Subroutine**
002	17	B'	
003	73	RC*	
004	06	06	
005	65	×	
006	43	RCL	
007	08	08	
008	45	Y^X	
009	43	RCL	
010	00	00	
011	94	+/-	
012	65	×	
013	44	SUM	
014	04	04	
015	43	RCL	
016	00	00	
017	95	=	
018	44	SUM	
019	05	05	
020	69	OP	
021	36	36	
022	97	DSZ	
023	00	00	
024	17	B'	
025	43	RCL	
026	04	04	
027	75	-	
028	43	RCL	
029	07	07	
030	95	=	
031	92	RTN	
032	76	LBL	
033	16	A'	**Find %i (IRR)**
034	98	ADV	
035	22	INV	
036	58	FIX	
037	43	RCL	
038	11	11	
039	55	÷	
040	43	RCL	
041	10	10	
042	94	+/-	
043	85	+	
044	32	X:T	
045	43	RCL	
046	10	10	
047	55	÷	
048	43	RCL	
049	09	09	
050	75	-	
051	35	1/X	
052	65	×	
053	53	(	
054	32	X:T	
055	33	X²	
056	75	-	
057	43	RCL	
058	12	12	
059	55	÷	
060	43	RCL	
061	10	10	
062	95	=	
063	35	1/X	**1st i**
064	68	NOP	
065	44	SUM	
066	08	08	
067	04	4	**i-control**
068	94	+/-	
069	22	INV	
070	28	LOG	
071	32	X:T	
072	76	LBL	**h-correction**
073	71	SBR	
074	01	1	
075	02	2	
076	42	STO	
077	06	06	
078	43	RCL	
079	01	01	
080	44	SUM	
081	06	06	
082	42	STO	
083	00	00	
084	25	CLR	
085	42	STO	
086	04	04	
087	42	STO	
088	05	05	
089	17	B'	
090	55	÷	
091	43	RCL	
092	05	05	
093	65	×	
094	43	RCL	
095	08	08	
096	95	=	**h**
097	68	NOP	
098	44	SUM	
099	08	08	
100	50	I×I	
101	77	GE	
102	71	SBR	
103	69	OP	
104	38	38	
105	43	RCL	
106	08	08	
107	65	×	
108	01	1	
109	00	0	
110	00	0	
111	95	=	
112	58	FIX	
113	04	04	
114	99	PRT	
115	69	OP	
116	28	28	
117	91	R/S	
118	76	LBL	
119	11	A	**Enter C's (Cash Flows)**
120	22	INV	
121	58	FIX	
122	69	OP	
123	26	26	
124	72	ST*	
125	06	06	
126	44	SUM	
127	09	09	
128	69	OP	
129	21	21	
130	69	OP	
131	22	22	
132	69	OP	
133	23	23	
134	65	×	
135	32	X:T	
136	43	RCL	
137	01	01	
138	99	PRT	
139	65	×	
140	44	SUM	
141	10	10	
142	43	RCL	
143	02	02	
144	55	÷	
145	02	2	
146	65	×	
147	44	SUM	
148	11	11	
149	43	RCL	
150	03	03	
151	55	÷	
152	03	3	
153	95	=	
154	44	SUM	
155	12	12	
156	32	X:T	
157	58	FIX	
158	02	02	
159	99	PRT	
160	81	RST	
161	76	LBL	
162	12	B	**Enter P (PV)**
163	42	STO	
164	07	07	
165	22	INV	
166	44	SUM	
167	09	09	
168	99	PRT	
169	91	R/S	
170	76	LBL	**Subroutine for finding P (PV) and F (FV)**
171	10	E'	
172	58	FIX	
173	02	02	
174	99	PRT	
175	55	÷	
176	01	1	
177	00	0	
178	00	0	
179	95	=	
180	44	SUM	
181	08	08	
182	43	RCL	
183	01	01	
184	42	STO	
185	00	00	
186	17	B'	
187	92	RTN	
188	76	LBL	**Enter %i and find FV (payments end of period)**
189	13	C	
190	10	E'	
191	99	PRT	
192	91	R/S	
193	76	LBL	
194	18	C'	**Enter %i and find FV (payments beginning of period)**
195	10	E'	
196	65	×	
197	43	RCL	
198	08	08	
199	95	=	
200	99	PRT	
201	91	R/S	
202	76	LBL	
203	14	D	**Enter %i and find FV (payments end of period)**
204	10	E'	
205	65	×	
206	43	RCL	
207	08	08	
208	45	Y^X	
209	43	RCL	
210	01	01	
211	95	=	
212	50	I×I	
213	99	PRT	
214	91	R/S	
215	76	LBL	
216	19	D'	**Enter %i and find FV (payments beginning of period)**
217	10	E'	
218	65	×	
219	43	RCL	
220	08	08	
221	45	Y^X	
222	43	RCL	
223	02	02	
224	95	=	
225	50	I×I	
226	99	PRT	
227	91	R/S	
228	76	LBL	
229	15	E	**Initialise**
230	47	CMS	
231	01	1	
232	02	2	
233	42	STO	
234	06	06	
235	69	OP	
236	22	22	
237	02	2	
238	42	STO	
239	03	03	
240	69	OP	
241	28	28	
242	25	CLR	
243	81	RST	

MERGED CODES

62 72 83
63 73 84
64 74 92

"NEUTRALIZATION" between two consecutive digits.- Markús S. Markússon presents this very interesting twist in programming in 82-1p5 of Programbiten (Sweden): Suppose that you want to enter any digit between 0 and 9 and expect to obtain the correct print code for that digit. This means that, entering a digit between 0 and 8 will give you no trouble at all. Just add 1 to it to obtain the print code; 9, however,has to be sensed; if detected, you have to add 3 to it, to obtain 12 as the print code. So, the first program that comes to mind is PGM 1. Now, taken the label out and putting a direct address in it will result in PGM2. Can it still be shorter? Yes, says Markús. Look at PGM 3. Here the LBL separates the 1 and the 3 and "neutralizes" if you want, the effect of placing the 1 and the 3 so close together. It also "neutralizes" the 1 in case you want to add 3, by fooling the calculator into believing that you wrote a routine called "LBL 1".

PGM 1.

```
000  76 LBL
001  11  A
002  85  +
003  32 X:T
004  08  8
005  77  GE
006  44 SUM
007  03  3
008  95  =
009  91 R/S
010  76 LBL
011  44 SUM
012  01  1
013  95  =
014  91 R/S
```

PGM 2.

```
000  76 LBL
001  11  A
002  85  +
003  32 X:T
004  08  8
005  77  GE
006  00  00
007  11  11
008  03  3
009  95  =
010  91 R/S
011  01  1
012  95  =
013  91 R/S
```

PGM 3.

```
000  76 LBL
001  11  A
002  85  +
003  32 X:T
004  08  8
005  77  GE
006  00  00
007  10  10
008  03  3
009  76 LBL
010  01  1
011  95  =
012  92 RTN
```

PERIODIC TABLE OF THE ELEMENTS,- Author unknown.(Will the REAL author please stand up?)
------------------------------ This is another example of the letter getting separated from the program. If you submit a program or article, may I ask you to also put your name on the program or article. I think this is a great program and I will be happy to publish to name of the author in next issue, to put credit were credit is due.

PERIODIC TABLE OF THE ELEMENTS

The program contains data on 107 elements. The printout shows the Element Number, the Symbol, the Atomic Weight, the Valence, the Average number of Neutrons, the Number of Stable Isotopes, and the Percent Availability of the most common Isotope. It also prints the Mass; the number of Protons and the number of Neutrons of this isotope.

Radioactive elements, with 0% stable isotopes, list the number of naturally occurring radioactive isotopes. For man made elements the number of isotopes with half lives expressed in the same time unit as the one with the longest half life are listed. The half life along with the mass, protons and neutrons of this isotope is listed for all radioactive elements.

The data on each element comes from the 60th Edition (1980) of the CRC Handbook Of Chemistry And Physics. The Atomic Weights, which are based on the Carbon 12 scale, are the weight in grams of 6 x 10^{23} atoms of the normal mixture of all the stable isotopes of the element. These weights in some cases have been rounded off to four significant digits to fit the storage format of the program. The percent availability of the most common isotope, in many cases, has been rounded off to a whole number. A listing of 100% is actually true if the number of isotopes is one. Any thing 99.5% or higher has been rounded off. The Valence listed is the Number of electrons in an incomplete outer shell of an atom, the inert elements with a complete outer shell are listed as 0 valence.

There are two ways of finding an element, the fastest way is to enter the element number Indirect recall produces the required register at once. The othe way is to enter the symbol using its print code, this starts a check of each register in tund to find the register with the same print code stored as an integer. Checking takes about half a second per register and up to 54 registers may have to be checked even if the proper data card is in use.

USER INSTRUCTIONS

The program consists three cards, the main program in banks 1 & 2 has 479 steps. Two data cards, Card A and Card B each contain 54 registers of element data, one register for each element. Card A contains elements 1 through 54 and the heavier elements 55 through 107 are on Card B. It saves time if you can guess which card to enter first.

1 Read both sides of the program card and both sides of one data card.
2a Enter the element number and push key A. All data will be printed. OR
2b Enter the print code of the elements symbol and push B. Data printed.

If Card B is in use and the element is between 1 and 54, a note is printed. READ CARD A R/S The program halts with a 3 on display. Run bank 3 of Card A, push 0 or CLR and run the back side of the card. Then, as the note says, push R/S, The tape advances one step and the program stops with the number or code you entered on display. Push either A or B as required. This time you are sure of getting the data wanted. A similar note will call for Card B if card A in in use and the element number is 55 or higher.

Clearing is automatic after each printout, enter another code or number for as many elements as needed.

With radioactive elements whose half life is measured in years, the half life is expressed as a power of ten with a single digit and the exponent.

If you want to amaze your friends with the size of a number that can be stored along with all the other data, put a NOP at space 422 of the program listing. There is an EE there now.

PROGRAM DISCRIPTION

The data for each element is stored in a single register. Normal elements are stored as positive numbers and radioactive elements carry a minus sign. The data for radioactive elements is slightly different. Flag 8 is set when a minus register is detected.

For elements 1 to 54, the register number is the same as the element. The data is stored on Data Card A. For elements 55 to 107 the register number is the element number minus 54. Card B contains this data. Flag 1 is set when Card A is in use and Flag 2 when Card B has been entered.

The print code for the elements symbol is an integer in each register. This allows checking the symbol entered against the integer in each register to find the element when a symbol is entered. The program starts with register 54 and checks each register in decending order untill the matching register is found. DSZ 0 does the counting. It takes about ½ second to check each register so this is a slow way to find elements. Since you dont know the number, you may have the wrong card in use and after changing cards, more search time is needed.

The fast way is to enter the element number. The number is stored in R 00 and if it is 54 or less and Flag 1 is set, RCL/IND will recall the proper register. If it is 55 or higher and Flag 2 is set, 54 is subtracted from R 00 and the card B data is recalled. If the wrong Flag is set the program prints "READ CARD A R/S" if Flag 2 is set and "READ CARD B R/S" if Flag 1 is set. All the user has to do is read the card called for and push R/S as instructed. This resets the old Flag and recalls the number or symbol entered. The user pushes key A to run the program again. This time since the right data is available, the data is printed at once.

When a symbol is entered, and the program checks all 54 registers without finding the matching code the same note will be printed and after both sides of the requested card have been read, R/S and key B will restart the program.

Since up to four memory spaces are needed for the print code, only nine decimal digits can be stored with element data. With normal elements the four digits closest to the decimal point are the four most significant digits of the atomic weight. For #1 to 4 Flag 3 is set to causean integer and a three digit decimal to be orinted. For #5 to 42 Flag 4 sets the point between two integers and a two place decimal. Higher numbers all have only a single digit decimal.

Space #5 records the valence of the element. Space #6 is code for the amount that must be added or subtracted from the integer part of the atomic weight to give the mass of the most common isotope. 3 is added to the actual number to produce a digit that is always positive. Space 7 contains the number of stable isotopes minus 1. This allows 1 to 10 to be stored as a single digit. Spaces 8 and 9 contain the percent availability of the most common isotope. Since this can not possibly be 0%, zero in 8 & 9 is interpreted as 100%.

When card B is in use Flag 2 causes the program to skip the part of the program that sets Flags 3 and 4 when registers 1 through 42 are recalled. So all atomic weights have four digit interers and a one place decimal.

With radioactive elements the data packing is slightly different. Spaces 1, 2 & 3 contain the atomic weight rounded off to a whole number. This allows space #4 to be used to store a code number for the time unit for the half life. 0 is for Years, 3 is for Months, 4 is for Days, 5 is for Hours, 6 is for Minutes. and 7 is for Seconds. After being unpacked this number is put in R. 59 and STF IND 59 sets the flag that controls the printing of the time unit. There is no conflict in using Flags 3 & 4 here as the only radioactive element on card A has a half life in years and with card B in use Flags 3 & 4 never set set. Spaces 5, 6 & 7 are the same as before. Spaces 8 & 9 are the numerical value of the half life as an integer and a one place decimal, for seconds through months, With years #8 is the integer and #9 is the exponent for years in scientific notation or the number of zeros anned in normal notation.

PERIODIC TABLE.-

```
000  91 R/S
001  02   2
002  03   3
003  32 X:T
004  43 RCL
005  01  01
006  59 INT
007  67  EQ
008  00   0
009  13  13
010  86 STF
011  02   2
012  92 RTN
013  86 STF
014  01   1
015  92 RTN
016  76 LBL
017  12   B
018  82 HIR
019  03  03
020  71 SBR
021  00   0
022  01  01
023  05   5
024  04   4
025  42 STO
026  00  00
027  82 HIR
028  13  13
029  32 X:T
030  73 RC*
031  00  00
032  59 INT
033  50 I×I
034  67  EQ
035  00   0
036  83  83
037  97 DSZ
038  00   0
039  00   0
040  30  30
041  87 IFF
042  02   2
043  00   0
044  47  47
045  01   1
046  85   +
047  01   1
048  03   3
049  95   =
050  69  OP
051  03  03
052  03   3
053  05   5
054  06   6
055  03   3
056  03   3
057  06   6
058  69  OP
059  04  04
060  43 RCL
061  56  56
062  69  OP
063  01  01
064  43 RCL
065  57  57
066  69  OP
067  02  02
068  69  OP
069  05  05
070  03   3
071  91 R/S
072  82 HIR
073  13  13
074  98 ADV
075  81 RST
076  76 LBL
077  16  A'
078  65   ×
079  01   1
080  00   0
081  95   =
082  92 RTN
083  43 RCL
084  00  00
085  87 IFF
086  01   1
087  11   A
088  85   +
089  05   5
090  04   4
091  95   =
092  76 LBL
093  11   A
094  82 HIR
095  03  03
096  71 SBR
097  00   0
098  01  01
099  05   5
100  05   5
101  32 X:T
102  82 HIR
103  13  13
104  22 INV
105  77  GE
106  01   1
107  22  22
108  75   -
109  05   5
110  04   4
111  95   =
112  22 INV
113  87 IFF
114  02   2
115  00   0
116  45  45
117  42 STO
118  00  00
119  61 GTO
120  01   1
121  48  48
122  22 INV
123  87 IFF
124  01   1
125  00   0
126  47  47
127  42 STO
128  00  00
129  32 X:T
130  05   5
131  22 INV
132  77  GE
133  01   1
134  40  40
135  86 STF
136  03   3
137  61 GTO
138  01   1
139  48  48
140  04   4
141  04   4
142  22 INV
143  77  GE
144  01   1
145  48  48
146  86 STF
147  04  04
148  73 RC*
149  00  00
150  59 INT
151  69  OP
152  04  04
153  82 HIR
154  13  13
155  69  OP
156  06  06
157  98 ADV
158  73 RC*
159  00  00
160  29  CP
161  77  GE
162  01   1
163  67  67
164  86 STF
165  08  08
166  50 I×I
167  22 INV
168  59 INT
169  65   ×
170  04   4
171  22 INV
172  28 LOG
173  75   -
174  22 INV
175  59 INT
176  82 HIR
177  05  05
178  95   =
179  55   ÷
180  03   3
181  22 INV
182  28 LOG
183  95   =
184  87 IFF
185  03   3
186  01   1
187  94  94
188  16  A'
189  87 IFF
190  04   4
191  01   1
192  94  94
193  16  A'
194  82 HIR
195  04  04
196  75   -
197  22 INV
198  59 INT
199  42 STO
200  59  59
201  95   =
202  87 IFF
203  08   8
204  02   2
205  10  10
206  85   +
207  43 RCL
208  59  59
209  95   =
210  32 X:T
211  04   4
212  03   3
213  03   3
214  07   7
215  69  OP
216  04  04
217  32 X:T
218  69  OP
219  06  06
220  82 HIR
221  15  15
222  16  A'
223  75   -
224  22 INV
225  59 INT
226  82 HIR
227  05  05
228  95   =
229  32 X:T
230  04   4
231  02   2
232  01   1
233  03   3
234  02   2
235  07   7
236  69  OP
237  04  04
238  32 X:T
239  69  OP
240  06  06
241  22 INV
242  87 IFF
243  08   8
244  02   2
245  53  53
246  01   1
247  00   0
248  49 PRD
249  59  59
250  86 STF
251  40 IND
252  59  59
253  43 RCL
254  55  55
255  69  OP
256  04  04
257  82 HIR
258  14  14
259  58 FIX
260  00   0
261  52  EE
262  22 INV
263  52  EE
264  22 INV
265  58 FIX
266  75   -
267  82 HIR
268  13  13
269  95   =
270  69  OP
271  06  06
272  82 HIR
273  15  15
274  16  A'
275  75   -
276  59 INT
277  82 HIR
278  02  02
279  95   =
280  16  A'
281  75   -
282  22 INV
283  59 INT
284  82 HIR
285  05  05
286  85   +
287  01   1
288  95   =
289  32 X:T
290  02   2
291  04   4
292  03   3
293  06   6
294  03   3
295  02   2
296  69  OP
297  04  04
298  32 X:T
299  69  OP
300  06  06
301  06   6
302  01   1
303  69  OP
304  04  04
305  22 INV
306  87 IFF
307  08   8
308  03   3
309  37  37
310  00   0
311  69  OP
312  06  06
313  87 IFF
314  00   0
315  03   3
316  96  96
317  87 IFF
318  03   3
319  04   4
320  29  29
321  87 IFF
322  04   4
323  04   4
324  38  38
325  87 IFF
326  05   5
327  04   4
328  46  47
329  87 IFF
330  06   6
331  04   4
332  54  54
333  87 IFF
334  07   7
335  04   4
336  63  63
337  82 HIR
338  15  15
339  29  CP
340  22 INV
341  67  EQ
342  03   3
343  45  45
344  01   1
345  16  A'
346  16  A'
347  69  OP
348  06  06
349  03   3
350  82 HIR
351  52  52
352  43 RCL
353  58  58
354  69  OP
355  04  04
356  82 HIR
357  14  14
358  59 INT
359  75   -
360  82 HIR
361  12  12
362  95   =
363  42 STO
364  59  59
365  69  OP
366  06  06
367  03   3
368  03   3
369  03   3
370  05   5
371  03   3
372  02   2
373  69  OP
374  04  04
375  82 HIR
376  13  13
377  69  OP
378  06  06
379  03   3
380  01   1
381  04   4
382  01   1
383  69  OP
384  04  04
385  43 RCL
386  59  59
387  75   -
388  82 HIR
389  13  13
390  95   =
391  69  OP
392  06  06
393  98 ADV
394  25 CLR
395  81 RST
396  04   4
397  05   5
398  03   3
399  05   5
400  03   3
401  06   6
402  69  OP
403  04  04
404  82 HIR
405  15  15
406  16  A'
407  75   -
408  59 INT
409  82 HIR
410  05  05
411  95   =
412  16  A'
413  85   +
414  01   1
415  95   =
416  22 INV
417  28 LOG
418  65   ×
419  82 HIR
420  15  15
421  95   =
422  52  EE
423  69  OP
424  06  06
425  25 CLR
426  61 GTO
427  03   3
428  49  49
429  03   3
430  00   0
431  03   3
432  07   7
433  02   2
434  03   3
435  61 GTO
436  04   4
437  69  69
438  01   1
439  06   6
440  01   1
441  03   3
442  04   4
443  05   5
```

Program listing continues on next page.

PERIODIC TABLE ,-

```
444  61 GTO
445  04   4
446  69  69
447  02  2
448  03  3
449  03  3
```

```
450  05  5
451  61 GTO
452  04   4
453  69  69
454  03  3
455  00  0
```

```
456  02  2
457  04  4
458  03  3
459  01  1
460  61 GTO
461  04   4
```

```
462  69  69
463  03  3
464  06  6
465  01  1
466  07  7
467  01  1
```

```
468  05  5
469  69 OP
470  04  04
471  82 HIR
472  15  15
473  16 A'
```

```
474  69 OP
475  06  06
476  61 GTO
477  03   3
478  49  49
479  00  0
```

```
  CARD A
23.1008131          01
2317.4003031        02
2724.694112193      03
1417.901223         04
14.10813218         05
15.120143199        06
31.1401531          07
32.1600632          08
21.190073           09
3117.201803291      10
3113.229912         11
3022.243123279      12
1327.269832         13
3624.280943292      14
33.309752           15
36.320663395        16
1527.354573176      17
13.3995022          18
26.391013293        19
1513.400823597      20
3615.449622         21
3724.479022474      22
42.5094221          23
1535.520013384      24
3031.549422         25
2117.558522392      26
1532.589322         27
3124.587123468      28
```

```
1541.635413169      29
4631.653724449      30
2213.69723316       31
2217.725941437      32
1336.749252         33
3617.78966155       34
1435.799173151      35
2635.838002557      36
3514.854713172      37
3635.876222383      38
45.889122           39
4635.912224451      40
3114.929112         41
3032.959410624      42
3715.970013235      43
3541.101112632      44
3523.102912         45
3316.106403527      46
1322.107913152      47
1516.112421729      48
2431.114832196      49
3631.118741933      50
3614.121853157      51
3717.127660734      52
24.126972           53
4417.131302827      54
13424031.           55
35171316.           56
15133516.           57
30133636.           58
```

```
  CARD B
1536.132912         01
1413.137322672      02
2713.1389221        03
1517.141124388      04
3335.140922         05
3116.144225627      06
-3330.14502302      07
3630.150421627      08
1741.152022152      09
2216.157322625      10
3714.158922         11
1645.162521628      12
2332.164922         13
1735.167324534      14
3730.168922         15
4514.173022631      16
2741.175023197      17
2321.178521535      18
3713.1809221        19
43.183922431        20
3517.186222163      21
3236.19022164       22
2435.192222163      23
3337.195123534      24
1341.197013         25
2322.20062163       26
3727.204432171      27
3314.207242352      28
```

```
 1424.209053        29
-3332.209063611     30
-1337.210573183     31
-3531.222403238     32
-2135.223513004     33
-3513.226023322     34
-1315.227023121     35
-3723.232023519     36
-3313.231023133     37
-41.238023258       38
-3133.237023125     39
-3341.244023686     40
-1330.243023172     41
-1530.247023626     42
-1426.247023112     43
-1521.251023381     44
-3257.254323289     45
-2130.257323126     46
-3016.25752313      47
-3132.25562333      48
-2735.258623307     49
-2641.25772315      50
-2313.260723016     51
-71.000723009       52
-71.000723          53
-71.000723          54
 13424031.          55
 35171316.          56
 15133516.          57
 30133636.          58
```

```
        2.      HE

    4.003       WT
       0.      VAL
       2.     AV.N
       2.      ISO
     100.        %
       4.     MASS
       2.      PRO
       2.       NU

      42.       MO

    95.94       WT
       1.      VAL
      54.     AV.N
       7.      ISO
      24.        %
      98.     MASS
      42.      PRO
      56.       NU

READ CARD    B  R/S

      92.        U

     238.       WT
       2.      VAL
     146.     AV.N
       3.      ISO
       0.        %
       5. 09   YRS
     238.     MASS
      92.      PRO
     146.       NU
```

```
READ CARD     A  R/S

       50.        SN

     118.7        WT
        4.       VAL
       69.      AV.N
       10.       ISO
       33.         %
      120.      MASS
       50.       PRO
       70.        NU

       48.        CD

     112.4        WT
        2.       VAL
       64.      AV.N
        8.       ISO
       29.         %
      114.      MASS
       48.       PRO
       66.        NU

READ CARD     B  R/S
       80.        HG

     200.6        WT
        2.       VAL
      121.      AV.N
        7.       ISO
       30.         %
      202.      MASS
       80.       PRO
      122.        NU
```

```
        85.       AT

       210.       WT
         7.      VAL
       126.     AV.N
         2.      ISO
         0.        %
        8.3       HR
       210.     MASS
        85.      PRO
       125.       NU

       104.       KU

       257.       WT
         2.      VAL
       154.     AV.N
         2.      ISO
         0.        %
         5.      SEC
       257.     MASS
       104.      PRO
       153.       NU

        92.        U

       238.       WT
         2.      VAL
       146.     AV.N
         3.      ISO
         0.        %
5000000000.      YRS
       238.     MASS
        92.      PRO
       146.       NU
```

```
READ CARD     A  R/S

        1.         H

     1.008        WT
        1.       VAL
        0.      AV.N
        2.       ISO
      100.         %
        1.      MASS
        1.       PRO
        0.        NU

        8.         O

       16.        WT
        6.       VAL
        8.      AV.N
        3.       ISO
      100.         %
       16.      MASS
        8.       PRO
        8.        NU

READ CARD     B  R/S

       60.        ND

     144.2        WT
        2.       VAL
       84.      AV.N
        7.       ISO
       27.         %
      142.      MASS
```

T- and PI-PAD ATTENUATION.- Maurice Swinnen. In Electronics, January 13, 1982, p 176-177, I found a marvelous program written for the HP-41C by Albert E. Hayes JR, of Fullerton CA. His title is HP-41C CALCULATOR ANALYSES RESISTIVE ATTENUATORS. The program does just that: you enter the various resistive values for either a PI or a T-pad configuration, including the input source resistance and the output load resistance, and the result of the computation is the attenuation given in dB. The HP-41C program is for calculator use only, no printing. So, I translated the program for TI-59/PC100 use. To make the bells and whistles complete, I made the program draw the two schematic doagrams. The program is further completely interactive. That means it will prompt you for the resistive values it wants to know.

Mr. Hayes gives the following equations for the attenuation in dB. First for the T-pad attenuator:

$$20\log\left[1+\frac{R_3}{R_L}+\left(\frac{R_3}{R_2R_L}+\frac{1}{R_2}+\frac{1}{R_L}\right)(R_S+R_1)\right]\quad -20\log\left(\frac{R_L+R_S}{R_L}\right)$$

Next for the PI-pad attenuator:

$$20\log\left\{1+\frac{R_2}{R_3}+\frac{R_2}{R_L}+R_S\left[\frac{1}{R_1}+\frac{R_2}{R_1}\left(\frac{1}{R_3}+\frac{1}{R_L}\right)\right]\quad +R_S\left(\frac{1}{R_3}+\frac{1}{R_L}\right)\right\}-20\log\left(\frac{R_L+R_S}{R_L}\right)$$

The last term in each of the above equations represents the attenuation introduced by the voltage divider that is formed by the source and load resistance(Rs and Rl)

The user instructions for this TI-59/PC100 program are extremely simple: Just load the program, 3 card sides, and press E to initialize. The printer will draw the two schematic diagrams. Choose either A or B and press the corresponding user-defined key. All resistance values will be prompted. (see sample run)

This part of the listing is the alpha code registers (in bank 3) and the last few steps of the program which didn't fit on one full page. The rest of the program can be found on the next page.

```
366  54   )    374  03  03     2035362020.   30     3335173636.   40
367  85   +    375  35 1/X     2035022020.   31       13000000.   41
368  53   (    376  85   +     2035032020.   32       14000000.   42
369  43 RCL    377  43 RCL     2035042020.   33       35360071.   43
370  00  00    378  04  04     2035272020.   34       35020071.   44
371  65   ×    379  35 1/X     2020202020.   35       35030071.   45
372  53   (    380  61 GTO     6565656565.   36       35040071.   46
373  43 RCL    381  65   ×     3720331316.   37       35270071.   47
               ____________    5320331316.   38         356336.   48
                                        0.   39       13373764.   49
                                                      16140000.   50
```

NEW LOCAL CLUB.- Gary Michael Holton has formed a programmable calculator users' group which he calls the CHICAGO STUDENTS' ASSOCIATION OF PERSONAL PROGRAMMERS. They accept both RPN and AOS programmers, university and high school students in the Chicago area. Although they don't have a newsletter of their own, they have lots of fun at their meetings. Gary says that " something crazy has to happen when a bunch of Comp. Sci. majors with calculator fetishes get together." Gary further affirms that they strongly support the TI PPC NOTES. If you are interested write Gary at 496 Jefferson in Glencoe, IL 60022 or after 1 May 1982 at 405 North Wabash, Apt. 4103, Chicago IL, 60611.

SURVEYING- The SJC (Survey Calculations Journal, P.O. Box 6674, San Bernardino CA, 92412 USA, Vol. 2, 1982 contains two program for the TI-59: Curb Return Intersection and Least Squares Adjustment of a Triangular figure.(Quadrilateral)

Besides these two programs it contains several for the HP-41C. The editor is Joe Bell, a TI PPC Club member. From what I notice, Joe is slowly switching over to a word processor.

The newsletter also contains a remarkable HP-41C program called The 1981 Solar Ephimeris. Astronomy fans, please note.

PI and T-Pad Attenuation (listing)

```
-RS---R2------------
    :    :         :
   R1   R3        RL
    :    :         :
....................
π-PAD-----PRESS B

-RS---R1---R3-------
          :        :
         R2       RL
          :        :
....................
T-PAD-----PRESS A
```

```
        RS ?  R/S
   100.
        R1 ?  R/S
    75.
        R2 ?  R/S
    38.
        R3 ?  R/S
    19.
        RL ?  R/S
    50.

      T-PAD ATT=
  11.47     DB

        RS ?  R/S
   100.
        R1 ?  R/S
    75.
        R2 ?  R/S
    38.
        R3 ?  R/S
    19.
        RL ?  R/S
    50.

      π-PAD ATT=
  14.56     DB
```

```
000  91 R/S
001  76 LBL
002  68 NOP
003  06  6
004  02  2
005  69 OP
006  01  01
007  76 LBL
008  69 OP
009  06  6
010  02  2
011  69 OP
012  02  02
013  71 SBR
014  99 PRT
015  92 RTN
016  76 LBL
017  99 PRT
018  69 OP
019  04  04
020  76 LBL
021  98 ADV
022  69 OP
023  05  05
024  69 OP
025  00  00
026  92 RTN
027  76 LBL
028  91 R/S
029  43 RCL
030  30  30
031  69 OP
032  01  01
033  92 RTN
034  76 LBL
035  93  .
036  43 RCL
037  36  36
038  69 OP
039  01  01
040  69 OP
041  02  02
042  69 OP
043  03  03
044  69 OP
045  04  04
046  71 SBR
047  98 ADV
048  92 RTN
049  76 LBL
050  37 P/R
051  43 RCL
052  40  40
053  69 OP
054  03  03
055  92 RTN
056  76 LBL
057  42 STO
058  69 OP
059  03  03
060  43 RCL
061  48  48
062  71 SBR
063  99 PRT
064  92 RTN
065  76 LBL
066  15  E
067  71 SBR
068  91 R/S
069  43 RCL
070  32  32
071  69 OP
072  02  02
073  43 RCL
074  35  35
075  69 OP
076  03  03
077  71 SBR
078  99 PRT
079  71 SBR
080  68 NOP
081  03  3
082  05  5
083  00  0
084  02  2
085  69 OP
086  01  01
087  85  +
088  02  2
089  95  =
090  69 OP
091  02  02
092  85  +
093  02  2
094  03  3
095  95  =
096  71 SBR
097  99 PRT
098  71 SBR
099  68 NOP
100  71 SBR
101  93  .
102  43 RCL
103  38  38
104  69 OP
105  01  01
106  43 RCL
107  35  35
108  69 OP
109  02  02
110  71 SBR
111  37 P/R
112  43 RCL
113  42  42
114  69 OP
115  04  04
116  71 SBR
117  98 ADV
118  98 ADV
119  71 SBR
120  91 R/S
121  43 RCL
122  31  31
123  69 OP
124  02  02
125  43 RCL
126  33  33
127  69 OP
128  03  03
129  43 RCL
130  35  35
131  71 SBR
132  99 PRT
133  71 SBR
134  69 OP
135  03  3
136  05  5
137  00  0
138  03  3
139  69 OP
140  02  02
141  85  +
142  02  2
143  04  4
144  95.  =
145  71 SBR
146  99 PRT
147  71 SBR
148  69 OP
149  71 SBR
150  93  .
151  43 RCL
152  37  37
153  69 OP
154  01  01
155  43 RCL
156  35  35
157  69 OP
158  02  02
159  71 SBR
160  37 P/R
161  43 RCL
162  41  41
163  69 OP
164  04  04
165  71 SBR
166  98 ADV
167  98 ADV
168  98 ADV
169  98 ADV
170  91 R/S
171  76 LBL
172  44 SUM
173  53  (
174  43 RCL
175  04  04
176  85  +
177  43 RCL
178  00  00
179  54  )
180  55  ÷
181  43 RCL
182  04  04
183  95  =
184  28 LOG
185  65  ×
186  02  2
187  00  0
188  95  =
189  42 STO
190  05  05
191  92 RTN
192  76 LBL
193  11  A
194  86 STF
195  00  00
196  76 LBL
197  12  B
198  43 RCL
199  43  43
200  71 SBR
201  42 STO
202  91 R/S
203  42 STO
204  00  00
205  99 PRT
206  43 RCL
207  44  44
208  71 SBR
209  42 STO
210  91 R/S
211  42 STO
212  01  01
213  99 PRT
214  43 RCL
215  45  45
216  71 SBR
217  42 STO
218  91 R/S
219  42 STO
220  02  02
221  99 PRT
222  43 RCL
223  46  46
224  71 SBR
225  42 STO
226  91 R/S
227  42 STO
228  03  03
229  99 PRT
230  43 RCL
231  47  47
232  71 SBR
233  42 STO
234  91 R/S
235  42 STO
236  04  04
237  99 PRT
238  98 ADV
239  22 INV
240  87 IFF
241  00  00
242  87 IFF
243  43 RCL
244  37  37
245  69 OP
246  03  03
247  71 SBR
248  44 SUM
249  43 RCL
250  03  03
251  55  ÷
252  43 RCL
253  02  02
254  55  ÷
255  43 RCL
256  04  04
257  95  =
258  85  +
259  43 RCL
260  02  02
261  35 1/X
262  85  +
263  43 RCL
264  04  04
265  35 1/X
266  95  =
267  65  ×
268  53  (
269  43 RCL
270  00  00
271  85  +
272  43 RCL
273  01  01
274  95  =
275  85  +
276  53  (
277  43 RCL
278  03  03
279  55  ÷
280  43 RCL
281  04  04
282  54  )
283  85  +
284  01  1
285  76 LBL
286  65  ×
287  95  =
288  28 LOG
289  65  ×
290  02  2
291  00  0
292  95  =
293  75  -
294  43 RCL
295  05  05
296  95  =
297  42 STO
298  05  05
299  43 RCL
300  49  49
301  71 SBR
302  99 PRT
303  43 RCL
304  50  50
305  69 OP
306  04  04
307  43 RCL
308  05  05
309  58 FIX
310  02  02
311  69 OP
312  06  06
313  22 INV
314  58 FIX
315  98 ADV
316  81 RST
317  76 LBL
318  87 IFF
319  43 RCL
320  38  38
321  69 OP
322  03  03
323  71 SBR
324  44 SUM
325  43 RCL
326  03  03
327  35 1/X
328  85  +
329  43 RCL
330  04  04
331  35 1/X
332  95  =
333  65  ×
334  43 RCL
335  02  02
336  55  ÷
337  43 RCL
338  01  01
339  95  =
340  85  +
341  43 RCL
342  01  01
343  35 1/X
344  95  =
345  65  ×
346  43 RCL
347  00  00
348  95  =
349  85  +
350  01  1
351  85  +
352  53  (
353  43 RCL
354  02  02
355  55  ÷
356  43 RCL
357  03  03
358  54  )
359  85  +
360  53  (
361  43 RCL
362  02  02
363  55  ÷
364  43 RCL
365  04  04
```

PROGRAMMING PUZZLES.- Re-v7n1/2p9 and v7n3p12. I received several more solutions between the time I finished writing v7n3 and the time it arrived in your mail box. To the sorting program I got these:

Nello Coda, Erie PA.: LBL A STO 00 - X:T STO 01 = OP 10 +/- STO 03 RCL IND 03 X:T RTN

John Hamlin, Beaverton CO:

LBL X:T HIR 02 X:T - X:T = HIR 12 OP 18 IFF 07 = X:T LBL = CE INV STF 07 RTN

Dejan Ristanović and Saša Nick, Belgrade, Yugoslavia:

LBL A + (+/- + X:T) X (OP 10 + 1) OP 10 = R/S

Now, to the Powers of Minus 1 problem:

Dejan Ristanović and Saša Nick: LBL A X PI = COS R/S *and also this one:*

LBL A X 180 = COS R/S

J. Hamlin: LBL A DIV 2 = INV INT - .1 = OP 10 +/- R/S

And finally Nello Coda: LBL A RAD X PI = COS FIX 0 RTN

But then, Palmer Hanson, who proposed these Charlie Williamson puzzles in the first place, points out that:

" The min-max sorter somehow got misstated. The instructions should have been: PLACE a IN THE DISPLAY, AND b IN THE T-REGISTER. DEVISE A ROUTINE WHICH WILL PLACE max(a,b) IN THE DISPLAY REGISTER AND min(a,b) IN THE T-REGISTER.

The underlined words somehow were left out making the puzzle substantially easier.

Charlie's routine satisfied the problem defined above. None of the other responses in v7n3p12 do so since the problem was defined as only placing max(a,b) in the t-register. perhaps you will want to rerun the problem.

Curiously, none of the responders saw fit to compare the title of the puzzle with the statement of the problem. "

So, here we go again: I am rerunning that problem with the following statement:

Place a in the display (or x-) register and b in the t-register. Devise a routine which will place max(a,b) in the display (or x-) register and which will place min(a,b) in the t-register.

As this problem was not stated correctly to begin with,I hate to pronounce a verdict as to which routine, in my humble opinion, is the best. I will wait a couple of more issues to see how you can handle the above one, sorting both min and max.

As to the Powers of Minus 1 puzzle, I definitely have opinions.(and Palmer Hanson *seems to agree largely with me.) Many of the solutions have inadequacies, largely for not taking in account the precision of calculations. Palmer Hanson explains on the next two pages the various ways this problem could be solved and, to his surprise, some of the members still did it a different way. (see Bill Buechner's solution, for example)*

Clyde Durbin's complaint seems to be unfounded. For most of the participants the problem exceeded their capabilities, sorry to say.

Myer Boland's solution seems to be the best one. However, Myer's count of steps ingnores the LBL A entry used in the other solutions, and also ignores the need to have a zero in the t-register. Also, the other solutions include setting the RAD mode in their routine. If you add these requirements to Myer's routine and ignore the INV step, then he has 14 steps in total, still impressive, because his program has the widest range of input values while still giving the proper output.

Before you attack your typewriter and send out that "flaming" letter to me explaining in 36 pages why your entry should have won, please read Palmer Hanson's arguments on the next two pages. Then write me if you still feel like it.

ERRATUM: Data Fit to Eight Curves, Newcomer's Corner, v7n1:

Robert Caldwell tells me that the following changes will take care of most, if not all, the coding problems (and errors) in the above program, the longer of the two:

STEP	CODE	MNEMONIC
242	46	46
614	06	06
615	25	25

The first line you will find on v7n1p18 and the second and third one on v7n1p19.
This way, no more blinking R^2 for curve 7.

POWERS OF MINUS ONE - Palmer O. Hanson, Jr.

One of Charlie Williamson's programming puzzles in V7N1/2P9 of TI PPC Notes was, given an integer in the display then devise a routine which will return (-1) to the nth power to the display. The rules for counting program steps required were to include a label address for entry to the routine and a RTN at the end. Defining the problem more stringently, the output should be exactly plus one for even integer inputs, exactly minus one for odd integer inputs, and exactly one for a zero input. Furthermore, the routine should work with the widest possible range of input integers.

Proposed solutions to the problem appeared in V7N3P12/13 of TI PPC Notes, including a solution of my own. One of the more innovative solutions was Bill Buechner's use of Dsz counting:

```
LBL A STO 00 1 +/- DSZ 0 005 RTN
```

Unfortunately, this routine yields a minus one for an input of zero. Furthermore, the run time is excessive if the input integer is large. Buechner proposed another routine to minimize running time:

```
LBL A +/- DIV 2 = INV INT + .1 = OP 10 RTN
```

This routine yields the correct result for inputs of zero and the positive integers, but fails for negative integers.

Three of the solutions use a special case of DeMoivre's theorem to yield the answer, that is:

$$(\cos\theta)^n = \cos(n\theta)$$

All three solutions, including my own place unnecessary limitations on the range of the input integers if the solutions are to be exactly plus one or minus one. For example, for the version with the calculator in the degree mode:

```
LBL A DEG x 180 ) COS RTN
```

yields a display of -0.999999998 for an odd input integer as small as 20,000,001. For an input as small as 2,000,005 the display will be a minus one, but if one is added to the solution the display will read 1.9-11, not the zero desired, indicating that the value in the display register is not exactly minus one. Similarly the solutions which use the calculator in the radians mode

```
LBL A RAD x PI ) COS RTN
```

yield the same display of -0.999999998 for an odd input integer as small as 20,000,001. For an input as small as 2,000,001 the display will show a minus one, but if one is added to the solution the display will read 1.9-11, again indicating that the display register is not exactly minus one.

The range of allowable input integers for both routines can be extended by adding an OP 10 after the coside function, that is:

```
     LBL A DEG x 180 ) COS OP 10 RTN   when using degree mode

and  LBL A RAD x PI ) COS OP 10 RTN  when using radian mode.
```

Both functions will now yield exactly plus one or minus one for integers over the full ten digit range of the display. However, both functions fail for the thirteen digit odd value of 2,000,000,000,001 which can be synthesized in the display register, say by the sequence

```
     2000000000 x 1000 + 1 =
```

yielding a plus one instead of a minus one. Both routines seem to fail for all odd input values of 2,000,000,000,001 or greater.

Myer Boland's routine uses a different algorithm, first dividing by two to determine whether the input integer is even or odd:

```
     DIV 2 ) INV INT X=T PI PI LBL PI COS RTN
```

Curiously enough, that routine fails at exactly the same thirteen digit value of 2,000,000,000,001 as with the DeMoivre's theorem routines. Furthermore, Myer's routine is longer if one includes the necessary controls to ensure that the routine will work, namely that the calculator is in radians mode, the t register is zero, and the input label address is included:

```
     LBL A RAD CP DIV 2) INV INT X=T PI PI LBL PI COS RTN
```

Myer's routine fails at 2,000,000,000,001 because of an unpublished characteristic of the display register, that is, that if a thirteen digit integer is divided by a single digit integer such that the quotient would still have a thirteen digit integer part, then the calculator <u>truncates</u> rather than rounds the result. An effective integer function takes place. I used this characteristic to provide a thirteen digit speedy factor finder capability for the TI-59.

The same characteristic can be used to extend the range of a powers of minus one routine to the full range of 9,999,999,999,999 but at a penalty in keystrokes:

```
     LBL A IxI - (CE DIV 2) INT x 2 = x 2 - 1 = +/- RTN
```

where this routine is preferred over a similar routine which uses the same number of keystrokes

```
     LBL A IxI - (CP DIV 2) INT x 2 = +/- INV X=T 018 1 RTN
```

since the first routine leaves the display in a dead state. The INT function is redundant if the quotient has a thirteen digit integer part.

DATE OF EASTER.- In PGM 09, Mar-Apr 1982 the latest issue of the newsletter edited by the new Danish Programming Club, I saw an article by P.Korsgaard Johansen, telling the members about a new algorithm to compute the date of Easter that appeared in the February 1981 issue of Scientific American and written by Thomas H.O'Beirne. It is useful to compute the date of Easter between the years 1900 and 2099:

1. Call the year Y. Subtract 1900 from Y. Call the difference N.
2. Divide N by 19. Call the remainder A.
3. Divide (7A + 1) by 19. Call the quotient B. Discard the remainder.
4. Divide (11A + 4 - B) by 29. Call the remainder M.
5. Divide N by 4. Call the quotient Q. Discard the remainder.
6. Divide (N + Q + 31 - M) by 7. Call the reaminder W.
7. Compute 25 - M - W = D_4. If $D_4 \geq 1$, Then it is the date in April.
8. If $D_4 < 1$, then compute 31 - $|D_4|$= D_3 which is the date in March.

It would not be too difficult to fit all that into a program for a 58 or a 59. But what about a TI-57 program?

A few test dates:

2008 on March 23	1978 on March 26	1986 on March 30
1991 on March 31	2018 on April 1	1961 on April 2
1981 on April 19	1943 on April 25	

ERRATUM- On v6n8p7 I had several programs in answer to a challenge on DIMENSION OPERA-PERATIONS I had published in an earlier issue. One of the programs was by Ralph Snyder of Indianapolis, IN. The program, to tell the truth, was not entirely his. I had made a few enhancements on it, which Ralph didn't go along with at all. Ever since he has been out for blood, mine to be exact. I tried to bury the matter by telling him I had mislaid the file. No dice; Ralph sent me a new one. Then, he sent me this text as a "suggested" one, to be "run at your convenience in one of the future issues." Moral of the story: Never underestimate the power of persuasion of somebody who has been around for the last 82 years. They get as tenacious as bull dogs.(Just kidding, Ralph. You see, I am running it, just as you told me to, OK?)

If I had the space, I'd write "I GOOFED" fifty times. Ralph W. Snyder sent in a little gem for my DIMENSIONS OPERATIONS contest last summer. But in a fit of misguided zeal, before printing it I added some warts, like reconversion of products and quotients; and my programming changes were kindergarten level, like GTO A' instead of A' and GTO 0 48 instead of B', and an OP 05 between his PRT and R/S. Mea culpa, mea culpa! Ralph was steamed when he saw my mayhem on his brainchild. I told him I'd run a correction, then mislaid the file. Mea culpa, mea culpa!

Here now is his program exactly as he submitted it. Associate it with instructions and examples on v6n8p7 and cross out any mention of a division example, and any mention of reconversion in the multiply examples. (As Ralph wrote me: 4½'x4½' = 20.25 sq.ft.; "reconversion" gives 20.03, and what shall the 3 be called? And he questions the rationale of dividing one length by another, which I had asked for, and suggests that his LBL D segment may as well be deleted.)

```
000  76 LBL|021  04   4|042  43 RCL|063  14   D|084  02   2
001  16 A' |022  54   )|043  01  01|064  53   (|085  75   -
002  99 PRT|023  55   ÷|044  54   )|065  16  A'|086  22 INV
003  75   -|024  01   1|045  99 PRT|066  35 1/X|087  59 INT
004  53   (|025  02   2|046  42 STO|067  65   ×|088  65   ×
005  22 INV|026  54   )|047  01  01|068  43 RCL|089  93   .
006  59 INT|027  54   )|048  91 R/S|069  01  01|090  08   8
007  75   -|028  99 PRT|049  17  B'|070  54   )|091  04   4
008  53   (|029  92 RTN|050  76 LBL|071  99 PRT|092  54   )
009  24  CE|030  76 LBL|051  13   C|072  91 R/S|093  65   ×
010  65   ×|031  11   A|052  53   (|073  76 LBL|094  93   .
011  01   1|032  53   (|053  16  A'|074  17  B'|095  00   0
012  00   0|033  16  A'|054  65   ×|075  75   -|096  01   1
013  00   0|034  42 STO|055  43 RCL|076  53   (|097  54   )
014  85   +|035  01  01|056  01  01|077  22 INV|098  54   )
015  22 INV|036  91 R/S|057  54   )|078  59 INT|099  99 PRT
016  59 INT|037  76 LBL|058  99 PRT|079  75   -|100  91 R/S
017  65   ×|038  12   B|059  42 STO|080  53   (|
018  02   2|039  53   (|060  01  01|081  24  CE|
019  01   1|040  16  A'|061  91 R/S|082  65   ×|
020  55   ÷|041  85   +|062  76 LBL|083  01   1|
```

LOTS OF PIE, by Bob Fruit.(no pun intended) Bob has risen to the challenge and attacked the problem of PI. I hope that this first "sheep" being over the bridge will be followed by the whole herd. Let's show the "friends" from the HP PPC Club that we are not beaten yet, even we lost (temporarily) the battle of the calendar, we don't intend to loose the whole "war."

On the next page you will find a table of PI computed to the first 10,000 places. It was obtained in July 1961 by Shanks and Wrench on an IBM 7090. At that time they computed PI to 100,265 places. This table was reprinted with permission of the publisher, The American Mathematical Society, from Mathematics of Computation, 1962,v14n77pp76-99.

LOTS OF π

by Bob Fruit

The "Calcu-Letter" column of Popular Science (July, 1981) reported that the Hewlett-Packard calculator club had calculated the value of π to 1,000 decimal places in less then 11½ hours and had challenged the Texas Instruments Personal Calculator Club to try and beat their time. This got my interest on the subject of calculating the value of π on my TI-59. I had never considered trying to calculate the value of π, particularly to 1,000 places.

I started by writing to both the TI club (to join it) and the H-P club (to get a copy of the article on their calculating the value of π). Not waiting for the replies I went about figuring out how I would tackle this problem. That effort led to program I. It is shown here because it makes it easier to follow the programming techniques and algorithms developed here, but also used in my later programs. (There are many improvements that are obviously needed in program I, but I do not want to dwell on them here. Remember it is just to help you see what's going on in the later program.)

When considering doing a problem like π you must determine how to do two things: First, what algorithm to use for calculating the value of π? I looked in Petr Beckman's book A History of π (this book is much more entertaining than the title might imply) to see what algorithms had been used in the past. The one I liked was;

$$\pi = 16 * \arctan(1/5) - 4 * \arctan(1/239).$$

This turned out to be the same algorithm the H-P people used. The arctan function has a nice numerical method;

$$\arctan(X) = X - X^3/3 + X^5/5 - X^7/7 + ...$$

The second thing needed is, how to do the multi-precision arithmatic. The multi-precision addition and subtraction is not to hard to figure out, but the multi-precision division looked like a formidable challenge. It turns out that a multi-precision numerator divided by a single precision denominator (the situation faced in this problem) is easy. Look at each block of digits of the multi-precision numerator as a stand alone single precision number. Carry out the normal division between two numbers, save the dividend (this is part of the multi-precision answer), and add the remainder onto the front of the next block of digits. Repeat the division process untill the entire multi-precision division is complete. This is just the way long division is taught in grade school, it just looks a little different.

Since the largest remainder could be 238 not more then 10 digits can be stored in each register or you would exceed the 13 digit accuracy of the TI-59 (and there are some

LOTS OF π -2-

obvious printing advantages too). This means that I will only get 460 decimal places in my calculation of π. I would need 207 registers to be able to get 1,000 decimal places. As best I can determine (I can not really read H-P program code) the H-P people used 205 registers in their effort on π. Since the H-P calculator has more user memory they will be able to calculate more decimal places using similar algorithms.

The procedure for running programs II are: Load all the programs on to magnetic cards (the calculator configurations are II-A 479.59, II-B 159.99, II-C 159.99). Make sure that registers 90-99 are all zero when saving program II-B. Read in program II-A and press A. Read in program II-A. Put program II-B in the card reader (it will be read when the calculator is ready for it). After program II-B has been read put program II-C into the card reader. To print additional copies of the answer use INV LIST. The last digit is too large by 4 (the last 3 digits should be 799).

What happens while the program is running? Program II-A sets the calculator in fast mode and calculates the value of 16 * arctan(1/5). Program II-B calculates the value of 4 * arctan(1/239) and subtracts it from the value found in the first program. Program II-C takes the value of π and performs all the carry/borrow 1 operations that were ignored during the addition/subtractions in the first two programs and prints the results.

Programs II will calculate the value of π to 460 decimal places (with the error noted before) in 6 hours 18 minutes 15 seconds. For program II-A the running time is 4:14:51. Program II-B's running time is 2:03:24. I do not count the couple of minutes of program II-C running time since that is not part of the calculation of π.

Now for the diappointing part. It is easy to project the running time if 1,000 decimal places could be calculated on a TI-59. There is a linear relationship between the number of decimal places and the running time of the programs, so 1000 / 460 * (6:18:15) = 13:46. This puts my approach 2¼ hours longer than the H-P effort.

Time will not permit me to get back into this problem now. If I do come back to this problem I will look at incororating the algabric contraction the H-P people developed for the formulas. I don't know if that would significantly improve the run times I have already gotten. I would like to thank Rich Nelson, editor for the H-P newsletter, for sending me a copy of their article π. I had sent him a SASE with my request.

```
1415926535 8979323846 2643383279 5028841971 6939937510 5820974944 5923078164 0628620899 8628034825 3421170679
8214808651 3282306647 0938446095 5058223172 5359408128 4811174502 8410270193 8521105559 6446229489 5493038196
4428810975 6659334461 2847564823 3786783165 2712019091 4564856692 3460348610 4543266482 1339360726 0249141273
7245870066 0631558817 4881520920 9628292540 9171536436 7892590360 0113305305 4882046652 1384146951 9415116094
3305727036 5759591953 0921861173 8193261179 3105118548 0744623799 6274956735 1885752724 8912279381 8301194912
9833673362 4406566430 8602139494 6395224737 1907021798 6094370277 0539217176 2931767523 8467481846 7669405132
0005681271 4526356082 7785771342 7577896091 7363717872 1468440901 2249534301 4654958537 1050792279 6892589235
4201995611 2129021960 8640344181 5981362977 4771309960 5187072113 4999999837 2978049951 0597317328 1609631859
5024459455 3469083026 4252230825 3344685035 2619311881 7101000313 7838752886 5875332083 8142061717 7669147303
5982534904 2875546873 1159562863 8823537875 9375195778 1857780532 1712268066 1300192787 6611195909 2164201989

3809525720 1065485863 2788659361 5338182796 8230301952 0353018529 6899577362 2599413891 2497217752 8347913151
5574857242 4541506959 5082953311 6861727855 8890750983 8175463746 4939319255 0604009277 0167113900 9848824012
8583616035 6370766010 4710181942 9555961989 4676783744 9448255379 7747268471 0404753464 6208046684 2590694912
9331367702 8989152104 7521620569 6602405803 8150193511 2533824300 3558764024 7496473263 9141992726 0426992279
6782354781 6360093417 2164121992 4586315030 2861829745 5570674983 8505494588 5869269956 9092721079 7509302955
3211653449 8720275596 0236480665 4991198818 3479775356 6369807426 5425278625 5181841757 4672890977 7727938000
8164706001 6145249192 1732172147 7235014144 1973568548 1613611573 5255213347 5741849468 4385233239 0739414333
4547762416 8625189835 6948556209 9219222184 2725502542 5688767179 0494601653 4668049886 2723279178 6085784383
8279679766 8145410095 3883786360 9506800642 2512520511 7392984896 0841284886 2694560424 1965285022 2106611863
0674427862 2039194945 0471237137 8696095636 4371917287 4677646575 7396241389 0865832645 9958133904 7802759009

9465764078 9512694683 9835259570 9825822620 5224894077 2671947826 8482601476 9909026401 3639443745 5305068203
4962524517 4939965143 1429809190 6592509372 2169646151 5709858387 4105978859 5977297549 8930161753 9284681382
6868386894 2774155991 8559252459 5395943104 9972524680 8459872736 4469584865 3836736222 6260991246 0805124388
4390451244 1365497627 8079771569 1435997700 1296160894 4169486855 5848406353 4220722258 2848864815 8456028506
0168427394 5226746767 8895252138 5225499546 6672782398 6456596116 3548862305 7745649803 5593634568 1743241125
1507606947 9451096596 0940252288 7971089314 5669136867 2287489405 6010150330 8617928680 9208747609 1782493858
9009714909 6759852613 6554978189 3129784821 6829989487 2265880485 7564014270 4775551323 7964145152 3746234364
5428584447 9526586782 1051141354 7357395231 1342716610 2135969536 2314429524 8493718711 0145765403 5902799344
0374200731 0578539062 1983874478 0847848968 3321445713 8687519435 0643021845 3191048481 0053706146 8067491927
8191197939 9520614196 6342875444 0643745123 7181921799 9839101591 9561814675 1426912397 4894090718 6494231961

5679452080 9514655022 5231603881 9301420937 6213785595 6638937787 0830390697 9207734672 2182562599 6615014215
0306803844 7734549202 6054146659 2520149744 2850732518 6660021324 3408819071 0486331734 6496514539 0579626856
1005508106 6587969981 6357473638 4052571459 1028970641 4011097120 6280439039 7595156771 5770042033 7869936007
2305587631 7635942187 3125147120 5329281918 2618612586 7321579198 4148488291 6447060957 5270695722 0917567116
7229109816 9091528017 3506712748 5832228718 3520935396 5725121083 5791513698 8209144421 0067510334 6711031412
6711136990 8658516398 3150197016 5151168517 1437657618 3515565088 4909989859 9823873455 2833163550 7647918535
8932261854 8963213293 3089857064 2046752590 7091548141 6549859461 6371802709 8199430992 4488957571 2828905923
2332609729 9712084433 5732654893 8239119325 9746366730 5836041428 1388303203 8249037589 8524374417 0291327656
1809377344 4030707469 2112019130 2033038019 7621101100 4492932151 6084244485 9637669838 9522868478 3123552658
2131449576 8572624334 4189303968 6426243410 7732269780 2807318915 4411010446 8232527162 0105265227 2111660396

6655730925 4711055785 3763466820 6531098965 2691862056 4769312570 5863566201 8558100729 3606598764 8611791045
3348850346 1136576867 5324944166 8039626579 7877185560 8455296541 2665408530 6143444318 5867697514 5661406800
7002378776 5913440171 2749470420 5622305389 9456131407 1127000407 8547332699 3908145466 4645880797 2708266830
6343285878 5698305235 8089330657 5740679545 7163775254 2021149557 6158140025 0126228594 1302164715 5097925923
0990796547 3761255176 5675135751 7829666454 7791745011 2996148903 0463994713 2962107340 4375189573 5961458901
9389713111 7904297828 5647503203 1986915140 2870808599 0480109412 1472213179 4764777262 2414254854 5403321571
8530614228 8137585043 0633217518 2979866223 7172159160 7716692547 4873898665 4949450114 6540628433 6639379003
9769265672 1463853067 3609657120 9180763832 7166416274 8888007869 2560290228 4721040317 2118608204 1900042296
6171196377 9213375751 1495950156 6049631862 9472654736 4252308177 0367515906 7350235072 8354056704 0386743513
6222247715 8915049530 9844489333 0963408780 7693259939 7805419341 4473774418 4263129860 8099888687 4132604721

5695162396 5864573021 6315981931 9516735381 2974167729 4786724229 2465436680 0980676928 2382806899 6400482435
4037014163 1496589794 0924323789 6907069779 4223625082 2168895738 3798623001 5937764716 5122893578 6015881617
5578297352 3344604281 5126272037 3431465319 7777416031 9906655418 7639792933 4419521541 3418994854 4473456738
3162499341 9131814809 2777710386 3877343177 2075456545 3220777092 1201905166 0962804909 2636019759 8828161332
3166636528 6193266863 3606273567 6303544776 2803504507 7723554710 5859548702 7908143562 4014517180 6246436267
9456127531 8134078330 3362542327 8394497538 2437205835 3114771199 2606381334 6776879695 9703098339 1307710987
0408591337 4641442822 7726346594 7047458784 7787201927 7152807317 6790770715 7213444730 6057007334 9243693113
8350493163 1284042512 1925651798 0694113528 0131470130 4781643788 5185290928 5452011658 3934196562 1349143415
9562586586 5570552690 4965209858 0338507224 2648293972 8584783163 0577775606 8887644624 8246857926 0395352773
4803048029 0058760758 2510474709 1643961362 6760449256 2742042083 2085661190 6254543372 1315359584 5068772460

2901618766 7952406163 4252257719 5429162991 9306455377 9914037340 4328752628 8896399587 9475729174 6426357455
2540790914 5135711136 9410911939 3251910760 2082520261 8798531887 7058429725 9167781314 9699009019 2116971737
2784768472 6860849003 3770242429 1651300500 5168323364 3503895170 2989392233 4517220138 1280696501 1784408745
1960121228 5993716231 3017114448 4640903890 6449544400 6198690754 8516026327 5052983491 8740786680 8818338510
2283345085 0486082503 9302133219 7155184306 3545500766 8282949304 1377655279 3975175461 3953984683 3936383047
4611996653 8581538420 5685338621 8672523340 2830871123 2827892125 0771262946 3229563989 8989358211 6745627010
2183564622 0134967151 8819097303 8119800497 3407239610 3685406643 1939509790 1906996395 5245300545 0580685501
9567302292 1913933918 5680344903 9820595510 0226353536 1920419947 4553859381 0234395544 9597783779 0237421617
2711172364 3435439478 2218185286 2408514006 6604433258 8856986705 4315470696 5747458550 3323233421 0730154594
0516553790 6866273337 9958511562 5784322988 2737231989 8757141595 7811196358 3300594087 3068121602 8764962867

4460477464 9159950549 7374256269 0104903778 1986835938 1465741268 0492564879 8556145372 3478673303 9046883834
3634655379 4986419270 5638729317 4872332083 7601123029 9113679386 2708943879 9362016295 1541337142 4892830722
0126901475 4668476535 7616477379 4675200490 7571555278 1965362132 3926406160 1363581559 0742202020 3187277605
2772190055 6148425551 8792530343 5139844253 2234157623 3610642506 3904975008 6562710953 5919465897 5141310348
2276930624 7435363256 9160781547 8181152843 6679570611 0861533150 4452127473 9245449454 2368288606 1340841486
3776700961 2071512491 4043027253 8607648236 3414334623 5189757664 5216413767 9690314950 1910857598 4423919862
9164219399 4907236234 6468441173 9403265918 4044378051 3338945257 4239950829 6591228508 5558215725 0310712570
1266830240 2929525220 1187267675 6220415420 5161841634 8475651699 9811614101 0029960783 8690929160 3028840026
9104140792 8862150784 2451670908 7000699282 1206604183 7180653556 7252532567 5328612910 4248776182 5829765157
9598470356 2226293486 0034158722 9805349896 5022629174 8788202734 2092222453 3985626476 6914905562 8425039127

5771028402 7998066365 8254889264 8802545661 0172967026 6407655904 2909945681 5065265305 3718294127 0336931378
5178609040 7086671149 6558343434 7693385781 7113864558 7367812301 4587687126 6034891390 9562009939 3610310291
6161528813 8437909904 2317473363 9480457593 1493140529 7634757481 1935670911 0137751721 0080315590 2485309066
9203767192 2033229094 3346768514 2214477379 3937517034 4366199104 0337511173 5471918550 4644902636 5512816228
8244625759 1633303910 7225383742 1821408835 0865739177 1509682887 4782656995 9957449066 1758344137 5223970968
3408005355 9849175417 3818839994 4697486762 6551658276 5848358845 3142775687 9002909517 0283529716 3445621296
4043523117 6006651012 4120065975 5851276178 5838292041 9748442360 8007193045 7618932349 2292796501 9875187212
7267507981 2554709589 0455635792 1221033346 6974992356 3025494780 2490114195 2123828153 0911407907 3860251522
7429958180 7247162591 6685451333 1239480494 7079119153 2673430282 4418604142 6363954800 0448002670 4962482017
9289647669 7583183271 3142517029 6923488962 7668440323 2609275249 6035799646 9256504936 8183609003 2380929345

9588970695 3653494060 3402166544 3755890045 6328822505 4525564056 4482465151 8754711962 1844396582 5337543885
6909411303 1509526179 3780029741 2076651479 3942590298 9695946995 5657612186 5619673378 6236256125 2163208628
6922210327 4889218654 3648022967 8070576561 5144632046 9279068212 0738837781 4233562823 6089632080 6822246801
2248261177 1858963814 0918390367 3672220888 3215137556 0037279839 4004152970 0287830766 7094447456 0134556417
2543709069 7939612257 1429894671 5435784687 8661444581 2314593571 9849225284 7160504922 1242470141 2147805734
5510500801 9086996033 0276347870 8108175450 1193071412 2339086639 3833952942 5786905076 4310063835 1983438934
1596131854 3475464955 6978103829 3097164651 4384070070 7360411237 3599843452 2516105070 2705623526 6012764848
3084076118 3013052793 2054274628 6540360367 4532865105 7065874882 2569815793 6789766974 2205750596 8344086973
5020141020 6723585020 0724522563 2651341055 9240190274 2162484391 4035998953 5394590944 0704691209 1409387001
2645600162 3742880210 9276457931 0657922955 2499672758 4610126483 6999892256 9596881592 0560010165 5256375678
```

LOTS OF PI, Bob Fruit.

PROGRAM II A

```
000   76  LBL
001   11   A
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   25  CLR
013   01   1
014   00   0
015   69  OP
016   17   17
017   47  CMS
018   22  INV
019   58  FIX
020   03   3
021   03   3
022   01   1
023   42  STO
024   00   00
025   01   1
026   42  STO
027   03   03
028   52  EE
029   01   1
030   00   0
031   22  INV
032   52  EE
033   42  STO
034   04   04
035   65   ×
036   08   8
037   00   0
038   95   =
039   42  STO
040   54   54
041   05   5
042   04   4
043   42  STO
044   01   01
045   08   8
046   42  STO
047   02   02
048   04   4
049   06   6
050   42  STO
051   05   05
052   25  CLR
053   42  STO
054   06   06
055   82  HIR
056   08   08
057   29  CP
058   73  RC*
059   01   01
060   22  INV
061   67   EQ
062   00   00
063   73   73
064   69  OP
065   21   21
066   69  OP
067   22   22
068   69  OP
069   35   35
070   61  GTO
071   00   00
072   58   58
073   82  HIR
074   18   18
075   65   ×
076   43  RCL
077   04   04
078   85   +
079   73  RC*
080   01   01
081   75   -
082   53   (
083   24  CE
084   55   ÷
085   02   2
086   05   5
087   54   )
088   59  INT
089   72  ST*
090   01   01
091   65   ×
092   02   2
093   05   5
094   95   =
095   82  HIR
096   08   08
097   43  RCL
098   06   06
099   65   ×
100   43  RCL
101   04   04
102   85   +
103   73  RC*
104   01   01
105   75   -
106   53   (
107   24  CE
108   55   ÷
109   43  RCL
110   03   03
111   54   )
112   59  INT
113   42  STO
114   06   06
115   65   ×
116   43  RCL
117   03   03
118   95   =
119   48  EXC
120   06   06
121   22  INV
122   87  IFF
123   01   01
124   01   01
125   27   27
126   22  INV
127   74  SM*
128   02   02
129   69  OP
130   21   21
131   69  OP
132   22   22
133   97  DSZ
134   05   05
135   00   00
136   73   73
137   22  INV
138   87  IFF
139   01   01
140   01   01
141   43   43
142   22  INV
143   86  STF
144   01   01
145   69  OP
146   23   23
147   69  OP
148   23   23
149   97  DSZ
150   00   00
151   00   00
152   41   41
153   25  CLR
154   22  INV
155   96  WRT
156   61  GTO
157   00   00
158   00   00
159   00   0
```

PROGRAM I

```
160   97  DSZ
161   00   00
162   01   01
163   38   38
164   92  RTN
165   68  NOP
166   71  SBR
167   00   00
168   95   95
169   25  CLR
170   73  RC*
171   01   01
172   87  IFF
173   01   01
174   01   01
175   94   94
176   74  SM*
177   02   02
178   85   +
179   73  RC*
180   02   02
181   32  X:T
182   43  RCL
183   04   04
184   77   GE
185   02   02
186   16   16
187   22  INV
188   74  SM*
189   02   02
190   01   1
191   61  GTO
192   02   02
193   10   10
194   22  INV
195   74  SM*
196   02   02
197   85   +
198   73  RC*
199   02   02
200   29  CP
201   77   GE
202   02   02
203   16   16
204   43  RCL
205   04   04
206   74  SM*
207   02   02
208   01   1
209   94  +/-
210   69  OP
211   32   32
212   74  SM*
213   02   02
214   69  OP
215   22   22
216   69  OP
217   21   21
218   69  OP
219   22   22
220   97  DSZ
221   00   00
222   01   01
223   70   70
224   00   0
225   95   =
226   69  OP
227   23   23
228   69  OP
229   23   23
230   22  INV
231   87  IFF
232   01   01
233   02   02
234   36   36
235   22  INV
236   86  STF
237   01   01
238   92  RTN
239   00   0
```

PROGRAM II C

```
000   05   5
001   03   3
002   42  STO
003   02   02
004   04   4
005   06   6
006   42  STO
007   01   01
008   73  RC*
009   02   02
010   32  X:T
011   43  RCL
012   04   04
013   77   GE
014   00   00
015   29   29
016   22  INV
017   74  SM*
018   02   02
019   69  OP
020   32   32
021   01   1
022   74  SM*
023   02   02
024   69  OP
025   22   22
026   61  GTO
027   00   00
028   08   08
029   73  RC*
030   02   02
031   29  CP
032   77   GE
033   00   00
034   50   50
035   43  RCL
036   04   04
037   74  SM*
038   02   02
039   69  OP
040   32   32
041   01   1
042   22  INV
043   74  SM*
044   02   02
045   69  OP
046   22   22
047   61  GTO
048   00   00
049   29   29
050   69  OP
051   32   32
052   97  DSZ
053   01   01
054   00   00
055   08   08
056   06   6
057   69  OP
058   17   17
059   07   7
060   22  INV
061   90  LST
062   66  PAU
063   91  R/S
064   00   0
```

PROGRAM II B

```
040   67   EQ
041   00   00
042   52   52
043   69  OP
044   21   21
045   69  OP
046   22   22
047   69  OP
048   35   35
049   61  CTO
050   00   00
051   37   37
052   22  INV
053   86  STF
054   02   02
055   82  HIR
056   18   18
057   32  X:T
058   82  HIR
059   17   17
060   82  HIR
061   08   08
062   32  X:T
063   65   ×
064   43  RCL
065   04   04
066   85   +
067   73  RC*
068   01   01
069   75   -
070   53   (
071   24  CE
072   55   ÷
073   02   2
074   03   3
075   09   9
076   54   )
077   59  INT
078   72  ST*
079   01   01
080   65   ×
081   02   2
082   03   3
083   09   9
084   95   =
085   82  HIR
086   07   07
087   22  INV
088   87  IFF
089   02   02
090   00   00
091   53   53
092   43  RCL
093   06   06
094   65   ×
095   43  RCL
096   04   04
097   85   +
098   73  RC*
099   01   01
100   75   -
101   53   (
102   24  CE
103   55   ÷
104   43  RCL
105   03   03
106   54   )
107   59  INT
108   42  STO
109   06   06
110   65   ×
111   43  RCL
112   03   03
113   95   =
114   48  EXC
115   06   06
116   22  INV
117   87  IFF
118   01   01
119   01   01
120   22   22
121   22  INV
122   74  SM*
123   02   02
124   69  OP
125   21   21
126   69  OP
127   22   22
128   97  DSZ
129   05   05
130   00   00
131   52   52
132   22  INV
133   87  IFF
134   01   01
135   01   01
136   38   38
137   22  INV
138   86  STF
139   01   01
140   69  OP
141   23   23
142   69  OP
143   23   23
144   97  DSZ
145   00   00
146   00   00
147   18   18
148   25  CLR
149   22  INV
150   90  LST
151   25  CLR
152   22  INV
153   96  WRT
154   61  GTO
155   00   00
156   00   00
157   00   0
158   00   0
159   00   0
```

CUBIC EQUATION. - Björn Gustavsson wrote this 172 step program that solves an equation of the form $x^3 + ax^2 + bx + c = 0$.

User Instructions:

1. Enter a and press A.
2. Enter b and press B.
3. Enter c and press C. The first (real) root will be displayed.
4. Press D.
5. If the display is not flashing, the display shows the second real root. Press E to obtain the third real root.
6. If the display is flashing, the roots are complex. The display shows the real part. Press E to obtain the imaginary part.. No need to press CE.(program does it automatically.)

```
000  76 LBL
001  34 √X
002  75  -
003  43 RCL
004  00  00
005  95  =
006  92 RTN
007  76 LBL
008  16 A'
009  22 INV
010  45 Yˣ
011  32 X:T
012  03  3
013  65  ×
014  32 X:T
015  69 OP
016  10  10
017  95  =
018  92 RTN
019  76 LBL
020  11  A
021  42 STO
022  00  00
023  92 RTN
024  76 LBL
025  12  B
026  42 STO
027  01  01
028  92 RTN

029  76 LBL
030  13  C
031  22 INV
032  86 STF
033  00  00
034  60 DEG
035  32 X:T
036  43 RCL
037  01  01
038  55  ÷
039  03  3
040  22 INV
041  49 PRD
042  00  00
043  75  -
044  43 RCL
045  00  00
046  33 X²
047  95  =
048  94 +/-
049  42 STO
050  04  04
051  32 X:T
052  55  ÷
053  02  2
054  85  +
055  43 RCL
056  00  00
057  65  ×

058  53  (
059  33 X²
060  75  -
061  43 RCL
062  01  01
063  55  ÷
064  02  2
065  95  =
066  94 +/-
067  42 STO
068  02  02
069  42 STO
070  06  06
071  33 X²
072  75  -
073  32 X:T
074  65  ×
075  33 X²
076  95  =
077  29 CP
078  77  GE
079  01  01
080  14  14
081  43 RCL
082  06  06
083  55  ÷
084  43 RCL
085  04  04
086  34 √X

087  45 Yˣ
088  03  3
089  95  =
090  22 INV
091  39 COS
092  55  ÷
093  03  3
094  95  =
095  42 STO
096  05  05
097  86 STF
098  00  00
099  01  1
100  02  2
101  00  0
102  44 SUM
103  05  05
104  02  2
105  65  ×
106  43 RCL
107  04  04
108  34 √X
109  65  ×
110  43 RCL
111  05  05
112  39 COS
113  61 GTO
114  34 √X
115  44 SUM

116  02  02
117  94 +/-
118  85  +
119  43 RCL
120  06  06
121  95  =
122  16 A'
123  48 EXC
124  02  02
125  16 A'
126  24 CE
127  42 STO
128  01  01
129  85  +
130  43 RCL
131  02  02
132  61 GTO
133  34 √X
134  76 LBL
135  14  D
136  87 IFF
137  00  00
138  00  00
139  99  99
140  43 RCL
141  01  01
142  85  +
143  43 RCL
144  02  02

145  85  +
146  95  =
147  55  ÷
148  02  2
149  94 +/-
150  61 GTO
151  34 √X
152  76 LBL
153  15  E
154  25 CLR
155  87 IFF
156  00  00
157  00  00
158  99  99
159  03  3
160  34 √X
161  55  ÷
162  02  2
163  65  ×
164  53  (
165  43 RCL
166  01  01
167  75  -
168  43 RCL
169  02  02
170  95  =
171  92 RTN
```

ASTRONOMY.- Darrell Newell, who works in the Global Positioning Program (GPS) and has used programmable calculators for orbital operations during the last four years, sent me a copy of a letter he sent to Turner F. Morgan of the HP PPC club. (v6n6/7p14) In it he tells about his experiences with HP-41C programs written by Turner and has some comments on how to improve upon them.

If you send me a self-addressed stamped-envelope (SASE) you will get a copy of this copy. Canadian, Mexican and other non-US based members, just a SAE (forget about the stamp) will do.

PC100 ON THE FRITZ.- Our printer is a simple, but surprisingly rugged mechanical device. Mechanical breakdowns are rare.The only thing it cannot stand very well are spikes on the power line. When that happens, the results are unpredictable and most of the time spectacular. Here is another example in our ongoing saga of "PC100 on the fritz."