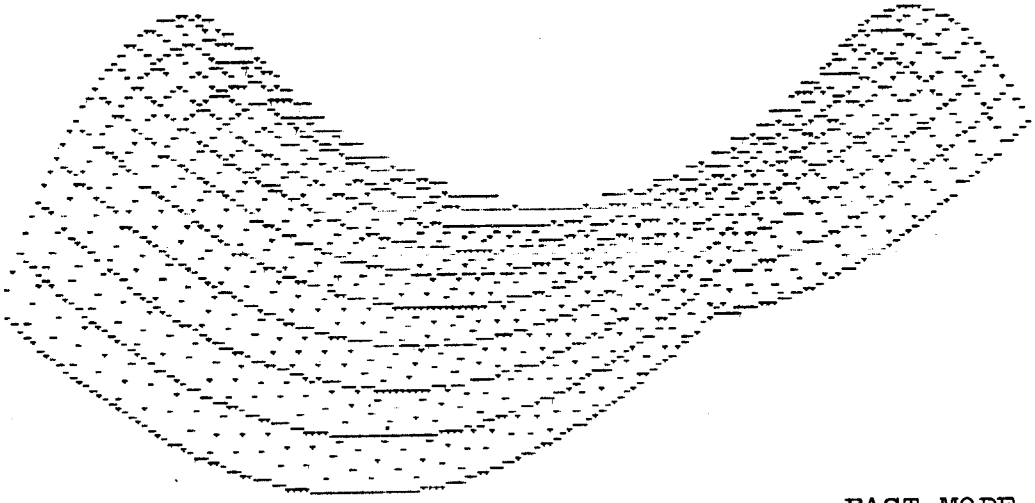
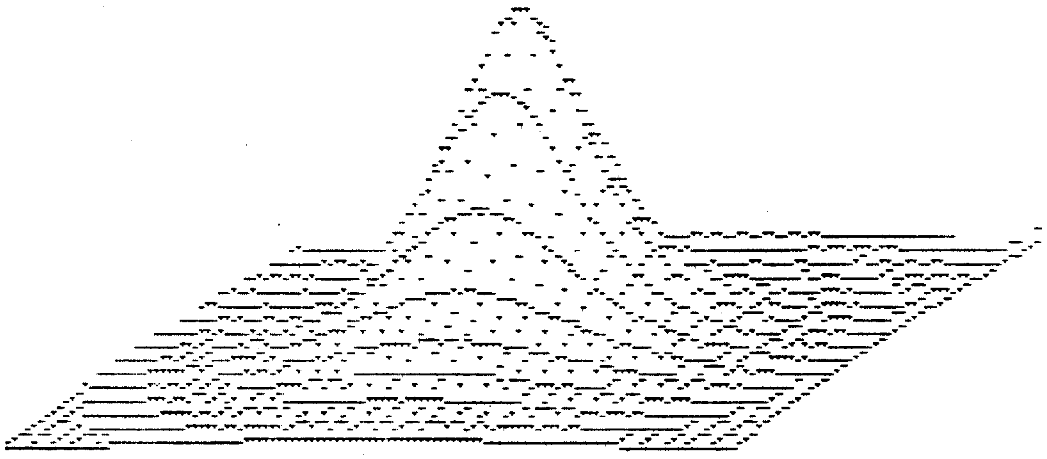
TI PPC NOTES

NEWSLETTER OF THE TI PROGRAMMABLE CALCULATOR CLUB
P.O. Box 1421, Largo, FL 34294

Volume 8, Number 2 March/April 1983



FAST MODE HIGH RESOLUTION
3-D GRAPHICS WITH THE TI-59
(See Page 11)



ERRATA - George Thomson found the following errors in my rendition of his treatise "Thoughts about Curve Fitting" in V8N1. The second line under equation (8) on V8N1P28 should read:

ln Y = A + B ln X

and, of course, on the same page just above the table of Rxx's the text should read Op 12 not Op $_{2}$2.

ERRATUM - In response to a question about the benchmark speed tests at the bottom of V7N7/8P10 Maurice Swinnen writes "I found the PPC newsletter from Richard Nelson in which he says he tested the speeds of the 59, the 41, and the 88. He definitely says '200' but I think it must have been '100'. Otherwise I get double the stated time for any of the three calculators."

Editor's Note: More careful comparisons of execution speed for the counting sequence 1 + 1 + 1 + 1 + ... appear elsewhere in this issue. It will be seen that for that benchmark the TI-59 operating in fast mode matches the speed of the TI-88, and is nearly twice as fast as the HP-41.

ERRATUM - The last paragraph under "Keyboard Interrupt" on V8N1P20 referred the readers to articles by Martin Neef (V5N7P11) and Dejan Ristanovic (V6N9/10P31) for discussions of more complex methods for branching from the keyboard. But Pierre Flener of Luxembourg writes that the Ristanovic article contains a serious error. The ninth line from the top should contain the sequence Pgm 09 SBR 058 ... Pgm 09 BST instead of the Pgm 01 SBR 012 ... Pgm 01 BST sequence. With XY in the display the sequence Pgm 01 SBR 012 clears data registers 1 through XY. See pages 8 and 9 of this issue for a detailed discussion of that technique. Dejan's program at the bottom of V6N9/10P31 uses the Pgm 01 SBR 012 sequence with a 4 in the display to clear registers 1 through 4. See program locations 029 through 036. The branching from the keyboard sequence as listed above appears in Dejan's program at locations 096 through 103. The same sequence appears in his Supertest TI-59 on V7N9P11 at program locations 385 through 392.

ERRATUM - Again, from George Thomson. In the "Profile Plot" article on V8N1P24 in the eighteenth line from the top the reference for a discussion of Flag 9 setting TRACE mode is page IV-65 of Personal Programming, not page V-65. See page 22 for discussion.

ERRATUM - Harold Copping finds that the instructions for the draftsman scaler programs (V5N6P8/9) are reversed. Use Label A when the entry is in fractional form, and Label B when the entry is in decimal form. Harold modified Richard Snow's program to add the capability to convert a fraction to a decimal. The final example for the SR-56 draftsman scaler program on V5N6P9 is incorrect. With a scale factor of 2, and a fractional entry of 1.0732 the result should be 2.0716, not the 1.0716 indicated in the instructions. Harold's revised program appears on page 22.

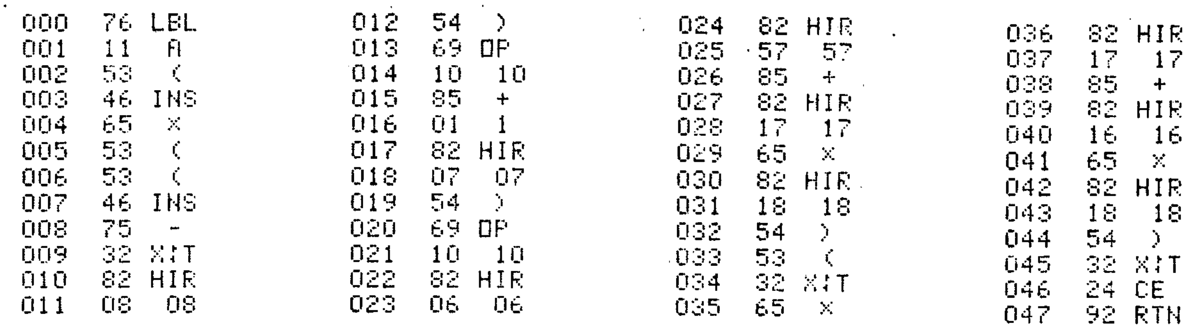
FROM THE EDITOR - The star program of this issue is Peter Poloczek's Fast-Grafik-3-D-Plot routine. I have yet to show the figures on the cover page to anyone who isn't absolutely amazed at this capability in a TI-59/PC-100. Dejan Ristanovic's discussion of hexadecimal codes for the TI-57 is a close second. Stu Smith's sequence for access to the statistics registers of the TI-55II is reminiscent of the late Dallas Egbert's discovery that if you pressed *read on a SR-52 and then simultaneously pressed the keys B, INV, sin, STO, EE, 4 and 0 the drive motor would turn on (V2N8P6 of 52 Notes). Fast mode continues to be popular. It is used in Peter's graphics program and in two new speedy factor finder programs. The newer techniques provide easier fast mode control and more versatility than the old Pgm 02 SBR 239/240 method. One important aspect of all the newer techniques is that they do not clear memory at entry to fast mode. With all this emphasis on speed I am reminded of a comment by a friend who simply isn't into computer technology. He defines a high speed digital computer as a device which can make more mistakes in a microsecond than he can make in a lifetime. The couplet from Bobbie Burns' "To a Louse" seems appropriate: "Oh wad some power the giftie gie us, to see oursels as others see us!"

In response to numerous requests I have added an index--on the last page of this issue. I also plan to consistently list errata on page 2 so that readers don't have to hunt through the entire issue for such material. Finally, I wish to thank the many readers who sent there best wishes for a speedy recovery.

ON NUMERICAL PRECISION - In subsequent issues I plan to examine some aspects of numerical precision of hand-held calculators. To get ready for that I suggest that you perform the following exercise. Enter a 2 in the display of your TI-58 or TI-59. Press the $\sqrt{x}$ key five times. You should have 1.021897149 in the display. Now press the x^2 key five times. You will see a 2. in the display. Press - 2 = and you will see -.0000000001 in the display, indicating that the sequence of five square roots followed by five squares does not quite yield the starting vale. The actual value in the TI-58 or TI-59 thirteen digit display register at the end of the sequence will be 1.999999999917.

Now, if you are bilingual (for newcomers, that means that you use both AOS and RPN calculators) perform the same test sequence. When you finish the display will show 2.000000022, indicating that the error due to truncation, roundoff, or whatever with those ten digit machines has penetrated two digits into the displayed value. The displayed value is also the value in the display register since those machines do not retain guard digits. Readers who don't have ten digit RPN machines can obtain the same 10 digit result on their TI-58 or TI-59 by pressing EE-INV-EE after each $\sqrt{x}$ or x^2 operation in the test sequence. This rounds from the guard digits to the ten digit display, and discards the guard digits. I will pick up at this point in subsequent issues.

MIN-MAX SORTER - Robert Prins. Charlie Williamson's solution which appeared in V7N3P12 cannot sort data pairs which are separated by a factor of 10^{13} or more. For example, with that solution place 100000000 in the t register, and 0.00000001 in the display register. The 100000000 will be returned to the display, but a zero will be in the t register. Henrik Klein's solution in V8N1P5 does not have that problem. Robert proposes the following solution



```
000  76 LBL     012  54  )      024  82 HIR     036  82 HIR
001  11  A      013  69 OP      025  57  57     037  17  17
002  53  (      014  10  10     026  85  +      038  85  +
003  46 INS     015  85  +      027  82 HIR     039  82 HIR
004  65  ×      016  01  1      028  17  17     040  16  16
005  53  (      017  82 HIR     029  65  ×      041  65  ×
006  53  (      018  07  07     030  82 HIR     042  82 HIR
007  46 INS     019  54  )      031  18  18     043  18  18
008  75  -      020  69 OP      032  54  )      044  54  )
009  32 X:T     021  10  10     033  53  (      045  32 X:T
010  82 HIR     022  82 HIR     034  32 X:T     046  24 CE
011  08  08     023  06  06     035  65  ×      047  92 RTN
```

and offers the following question: What is the use for the CE at step 046? Give up? When working with very large numbers such as -2 E +99 and 8 E +99 his program will sort the two values properly, but will end with an error condition. Charlie's solution gives an incorrect answer. Henrik's solution will flash, but can be cured with the same fix as in Robert's solution, namely a CE before the RTN.

1287 DIGITS OF PI - This program was called to my attention by Pierre Flener of Luxembourg. The program was published in the French scientific journal Science et Vie. I have not yet received permission to reprint the program in TI PPC Notes. The program delivers 1287 digits; thirteen digits per register times 99 registers, with register 00 reserved for Dsz control. The program is S - L - O - W ; it requires twenty-four and one-half days. Yes, I really did run my TI-59 that long over the holidays to be sure the program will deliver as promised. Until that program, my record for a continuous run was five days for a thirteen digit factor finder to declare that 9,999,999,999,971 was prime. If nothing else, those exercises attest to the reliability of the TI-59. Does anyone have a longer run to report?

One might conclude that this program would, at least temporarily, put the TI-59 in the digits of pi competition; but Pierre writes that there was also a HP-41C program which will calculate 3200 digits of pi in a few months.

Bob Fruit, who wrote the first TI-59 "digits of pi" program for TI PPC Notes (see V7N4/5P27), is trying to digest all of this for a coming issue. For those who want more and more pi, page 83 of the February 1983 issue of Scientific American describes recent Japanese efforts which are reported to have calculated pi to over eight million digits (2^{23}) in 6.8 hours.

THE GROSH OF FINN - W. J. Widmer. Consider any positive integer N(0). Raise each digit in this to the same power p, and sum these to form a new integer N(1); do the same with N(1) to form N(2), and then with N(2) to form N(3); etc. If this process is continued, the sequence of successive N's thus formed eventually becomes cyclic in a finite number of steps, i, which is unique for the given N(0); and, further, only a finite number of cyclic patterns are possible for a given power, p. The origin of this general problem is obscure. David Kullman discusses it in V14N1P4-10 of the Journal of Recreational Mathematics and credits one Michael Finn (citing Science News, March 1, 1980; page 134) with coining the name "GROSH" for these power sums--hence the name THE GROSH OF FINN.

The sums of the squares (i.e., p = 2) of the digits in the sequence integers N(0), N(1),..., N(i) were shown by A. Porges in 1945 (Amer. Math. Monthly V52P379-382) to converge either to a repeating sequence of 1's or to the single cyclic pattern 145, 42, 20, 4, 16, 37, 58, 89, 145, ... Any term of the latter might be the first to appear in a given case, but the cyclic sequence will otherwise be the same; 145 is singled out as the i'th N simply because it is the largest in the set. These are the only two possible cyclic patterns for p = 2. A simple proof for this "Square" Grosh of Finn is given (without reference) by Hugo Steinhaus in his book (translated from the Polish) One Hundred Problems in Elementary Mathematics (1979 Dover Publications reprint of Basic Books' 1964 translation; pages 55-58). The following TI-59 program will calculate and flash or print successive "Square" Grosh N(0), N(1), ..., N(i) with a final stop display of the number of steps, i, required to first reach 1 or 145, as the case may be. Integers N(0) for which N(i) equals 1 were called "Happy Numbers" by Porges--happy or not, I hope the program gives some amusement. You might even like to expand it to give cubic, quartic, or quintic grosh (Kullman's paper includes a listing through p = 6).

In the program here given, N is of the form $a_n10^{n-1} +...+ a_210 + a_1$ and contains n digits: a_1, a_2, ..., a_n.

To use the program, input N and press A. The program stops with N in the display. Then input the number of digits in N and press R/S. Thus for N = 2583, input 2583 and press A, input 4 and press R/S and see, successively, 102, 5, 25, 29, 85, 89, 145 and on stop, 7. Also, N = 4445556667 will output successively 280, 68, 100, 1 (flashed and/or printed) and a 4 on stop (this N is a "Happy Number").

Two features of the program may be of interest to beginners. (1): in steps 26-29 CP + x⇄t) places displayed digit "a" into the t register while preserving it in the x register. And (2): EE INV EE in steps 40-42 cannot be replaced by a simple INT because, in steps 33-39, RCL 02 INV log X x⇄t) yields hidden guard digits which would cause INT (were it next used, instead, for step 40) to display $[a_k(10^{k-1}) - 1]$ instead of $[a_k(10^{k-1})]$ whenever (k-1) is greater than or equal to 6 (or when k is greater than or equal to 7). This results from the fact that for k greater than or equal to 7, INV log produces an actual value (with hidden guard digits) slightly less than the rounded-up display value (0.00000410 less, for k = 7; the same is true for the y^x function); so that taking INT would then drop the display value by 1 to the calculator's actual integer part. In steps 20-24 and 71-75 this is not critical (though even there, EE INV EE might be preferred as a program nicety); also see Personal Programming, page V20.

Program Listing for "Grosh of Finn"

```
000  76 LBL    025  59 INT    050  43 RCL    075  54  )     100  43 RCL
001  11  A     026  29 CP     051  01  01    076  59 INT    101  00  00
002  99 PRT    027  85  +     052  33 X²     077  29 CP     102  66 PAU
003  98 ADV    028  32 X:T    053  44 SUM    078  22 INV    103  99 PRT
004  47 CMS    029  54  )     054  00  00    079  67  EQ    104  69 OP
005  42 STO    030  33 X²     055  43 RCL    080  00  00    105  24  24
006  01  01    031  44 SUM    056  00  00    081  89  89    106  42 STO
007  91 R/S    032  00  00    057  32 X:T    082  01  1     107  01  01
008  75  -     033  43 RCL    058  01  1     083  22 INV    108  00  0
009  01  1     034  02  02    059  67  EQ    084  44 SUM    109  42 STO
010  42 STO    035  22 INV    060  01  01    085  03  03    110  00  00
011  04  04    036  28 LOG    061  18  18    086  61 GTO    111  43 RCL
012  54  )     037  65  ×     062  01  1     087  00  00    112  03  03
013  42 STO    038  32 X:T    063  04  4     088  68  68    113  42 STO
014  02  02    039  54  )     064  05  5     089  32 X:T    114  02  02
015  42 STO    040  52 EE     065  67  EQ    090  01  1     115  61 GTO
016  03  03    041  22 INV    066  01  01    091  00  0     116  00  00
017  43 RCL    042  52 EE     067  18  18    092  32 X:T    117  17  17
018  01  01    043  22 INV    068  43 RCL    093  22 INV    118  66 PAU
019  55  ÷     044  44 SUM    069  00  00    094  77  GE    119  99 PRT
020  43 RCL    045  01  01    070  55  ÷     095  01  01    120  43 RCL
021  02  02    046  97 DSZ    071  43 RCL    096  00  00    121  04  04
022  22 INV    047  02  02    072  03  03    097  01  1     122  98 ADV
023  28 LOG    048  00  00    073  22 INV    098  44 SUM    123  99 PRT
024  54  )     049  17  17    074  28 LOG    099  03  03    124  91 R/S
```

Editor's Note: Who will write a generalized program which handles p at least through 6 on operator input, and which automatically finds the number of digits in the input integer?

POLYGON - Henrik Klein of Denmark. Translated by Maurice Swinnen.

Given the coordinates of the vertices of a polygon, this program will calculate the area. The equation used is:

$$A = \frac{1}{2}\left\{\begin{vmatrix} a_1 & a_2 \\ b_1 & b_2 \end{vmatrix} + \begin{vmatrix} a_2 & a_3 \\ b_2 & b_3 \end{vmatrix} + \ldots + \begin{vmatrix} a_n & a_1 \\ b_n & b_1 \end{vmatrix}\right\}$$

where each determinant is evaluated in the usual sense.

User Instructions:

1. Initialize by pressing 2nd A'.
2. For i = 1 to n: Enter x_i and press x=t; enter y_i and press A or press R/S.
3. After having entered all the vertices (do not enter the last point as the same one as the starting point) press E and obtain the area of the polygon.
4. Example: See the figure 1 on the next page. In sequence, press 2nd A' 4 x=t 0 A 2 x=t 3 A 5 x=t 6 A 3 x=t 5 A 0 x=t 3 A E and see the area 6.5 in the display.

A similar SR-52 program by Dean Athans appeared in V5N4/5P21.

POLYGON (cont)

```
000  76 LBL     024  00  00     048  32 X:T
001  11  A      025  25 CLR     049  92 RTN
002  53  (      026  92 RTN     050  61 GTO
003  98 ADV     027  61 GTO     051  11  A
004  65  ×      028  11  A      052  76 LBL
005  32 X:T     029  76 LBL     053  15  E
006  99 PRT     030  16 A'      054  43 RCL
007  48 EXC     031  25 CLR     055  02  02
008  00  00     032  42 STO     056  32 X:T
009  75  -      033  00  00     057  43 RCL
010  43 RCL     034  42 STO     058  03  03
011  00  00     035  01  01     059  11  A
012  65  ×      036  42 STO     060  02  2
013  32 X:T     037  04  04     061  22 INV
014  99 PRT     038  22 INV     062  49 PRD
015  48 EXC     039  76 LBL     063  04  04
016  01  01     040  25 CLR     064  43 RCL
017  54  )      041  86 STF     065  04  04
018  44 SUM     042  00  00     066  98 ADV
019  04  04     043  42 STO     067  99 PRT
020  43 RCL     044  03  03     068  98 ADV
021  01  01     045  32 X:T     069  92 RTN
022  22 INV     046  42 STO
023  87 IFF     047  02  02
```

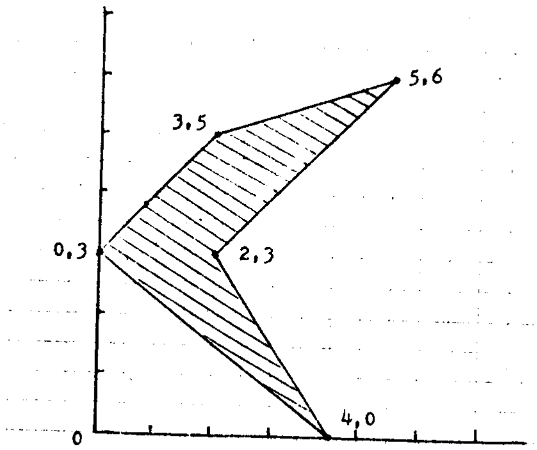


Figure 1

<u>Editor's Note</u>: If one goes around the polygon in the counter-clockwise sense as indicated in the example the area will be a positive number. If one goes around the polygon in a clockwise sense the area will be a negative number. Old timers, in particular those who worked in civil engineering, will recognize those characteristics as being the same as for the mechanical drafting aid known as a planimeter--a device for finding the area of irregularly shaped plane figures.

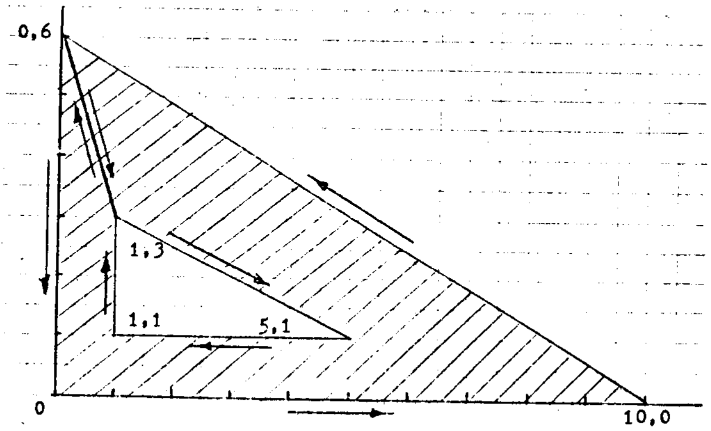


Figure 2

Recognizing the parallel to the planimeter, then one can perform other operations which are routinely accomplished with a planimeter. See the figure 2 with one polygon inside another. If one wishes to find the area between the two polygons, one may start at any vertex of the outer polygon and proceed in a counter-clockwise sense. But at some vertex, go from the outer polygon to the inner polygon and proceed in a clockwise sense around the inner polygon, including a return to the first vertex used from the inner polygon. Then go directly back to the last vertex used from the outer polygon (proceeding back along the same path, if you will), enter it again, and proceed until all the vertices of the outer polygon have been entered. Press E and find the area between the two polygons, in this case, 26. Old planimeter users would probably modify Henrik's program to include return to the starting vertex to maintain a parallel to planimeter useage.

NEWCOMER'S CORNER - Palmer Hanson. The early Solid State Software™ modules numbered 1 through 7 had common capabilities in the -01 diagnostic and module check programs, namely

- * A diagnostic routine initiated by the sequence 2nd Pgm 0 1 SBR =
- * A module check routine initiated by the sequence 2nd Pgm 0 1 SBR 2nd Write
- * A statistics initialization routine initiated by the sequence 2nd Pgm 0 1 SBR CLR

Modules 1 through 7 were also consistent in the location of the statistics initialization routine inside the -01 program such that the program sequence X Y Pgm 01 SBR 012 could be used to clear data registers 01 through XY. This capability was illustrated on page III-10 of the TI Programmable 59 Workbook and has been discussed in 52 Notes, PPX Exchange, and in TI PPC Notes. The only other substantial difference in the early -01 programs was the provision of a print capability for input and output values during the use of other programs in the Master Library, that is, the techniques defined by steps C1 and C2 of the User Instructions on page 7 of the TI Programmable 58/59 Master Library manual.

There is an unpublished idiosyncrasy of the diagnostic routine of the Master Library module, and of the Agriculture module as well: with those modules installed the diagnostic will fail if the calculator is in the EE or Eng mode. The addition of a CLR at the beginning of the sequence calling the diagnostic routine, as called out on page A-4 of Personal Programming will eliminate the problem with EE mode, but will do nothing for Eng mode. For the modules which have a diagnostic routine other than the Master Library and the Agriculture module, the potential difficulty is circumvented by including an INV Eng sequence shortly after entry to the diagnostic routine. The problem when in Eng mode arises because of the method used to check that the proper final value has resulted from the diagnostic calculations. An EE-INV-EE sequence is included immediately after the completion of the diagnostic exercise, for example in the Master Library at locations 055 through 057. The resulting value is placed in the t register, and a ten digit value is then generated for a comparison test. But the EE INV EE sequence when in EE or Eng mode rounds the display to an eight digit mantissa, not to a ten digit mantissa. Of course, when the diagnostic result rounded to eight digits is compared with the check value of ten digits, an error indication results. An alternative solution to the insertion of the INV Eng sequence in the diagnostic routines in the other modules would have been to use a thirteen digit check value.

The library modules numbered 8 through 11, that is SA, BD, M/U and EE do not follow the nicely defined rules for modules 1 through 7 and the Agriculture module. The BD, M/U and EE modules don't even have a diagnostic routine. A count of the individual program steps for the routines in those modules will show that the programmers simply ran out of room.

<u>NEWCOMER'S</u> <u>CORNER</u> (cont) Those three modules share another special feature. In each one the -01 program sequence was rearranged after deletion of the diagnostic capability such that the sequence X Y Pgm 01 SBR 004 must be used to access the appropriate part of the statistics initialization routine to clear registers 01 through XY.

The Securities Analysis module (number 8) is unique. It has a diagnostic routine, but it will not pass the diagnostic when in Eng mode. Its program sequence in the -01 module has been altered slightly from all the other modules such that the sequence X Y Pgm 01 SBR 007 must be used to clear registers 01 through XY.

Finally, the RPN module is entirely different. It includes no diagnostic and no statistics initialization. The instructions on page 1 of the <u>RPN</u> <u>Simulator</u> manual indicate that the sequence 2nd Pgm 0 1 SBR 2nd Write will cause the calculator to run about fifteen seconds prior to stopping with a 13 in the display, suggesting that there might be some sort of diagnostic, since that time is very close to the run time for the diagnostic routines in other modules. Actually, the calculator will run about four seconds, the time required to search for the Write label, which for some reason was moved out to location 494. The following table summarizes what has been said. Thank you's are due to Charlie Williamson and George Thomson who provided information on modules that I do not own.

Module Name	Module Number	Includes Diagnostic?	Diagnostic runs in Eng?	Includes Statistics Initialize	nnn to use for XY Pgm 01 SBR nnn to clear 01 through XY
ML	1	Yes	No	Yes	012
ST	2	Yes	Yes	Yes	012
RE	3	Yes	Yes	Yes	012
SY	4	Yes	Yes	Yes	012
NG	5	Yes	Yes	Yes	012
AV	6	Yes	Yes	Yes	012
LE	7	Yes	Yes	Yes	012
SA	8	Yes	No	Yes	007
BD	9	No	---	Yes	004
MU	10	No	---	Yes	004
EE	11	No	---	Yes	004
AG	12	Yes	No	Yes	012
RPN	13	No	---	No	---

PALINDROMIC NUMBERS - Myer Boland. Palindromic numbers read the same forwards as backwards, for example, 66, 121, 79497 and 62488426. Martin Gardiner in "The Ambidextrous Universe" posed the question:

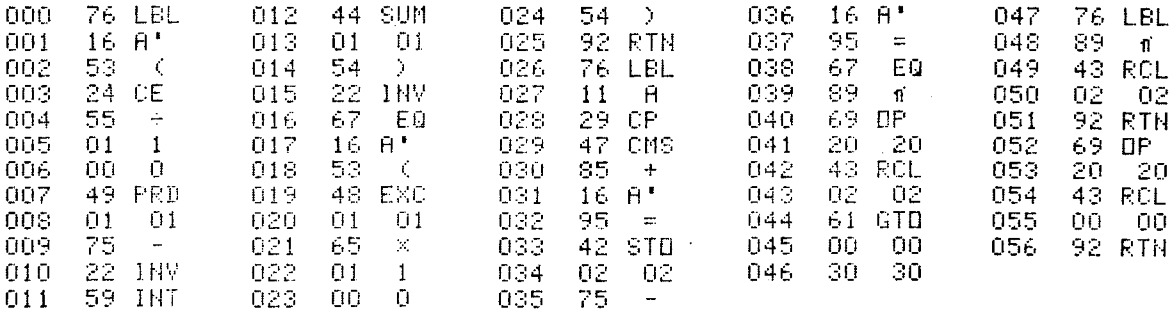
> If you add any positive integer to the integer formed by reversing the digits, then do the same thing with the sum, and keep repeating the process, will you eventually reach a sum that is palindromic?

Examples: 51 + 15 = 66 in one step.

59 + 95 = 154; 154 + 451 = 605; 605 + 506 = 1111 in three steps.

Other Examples: 195 converts to 9339 in 4 steps.
99 converts to 79497 in 6 steps.
89 converts to 8813200023188 in 24 steps.
196 doesn't reach a palindromic number in over 100 steps.

Program:



```
000  76 LBL    012  44 SUM    024  54  )     036  16 A'     047  76 LBL
001  16 A'     013  01  01    025  92 RTN    037  95  =     048  89  π
002  53  (     014  54  )     026  76 LBL    038  67  EQ    049  43 RCL
003  24 CE     015  22 INV    027  11  A     039  89  π     050  02  02
004  55  ÷     016  67  EQ    028  29 CP     040  69 OP     051  92 RTN
005  01  1     017  16 A'     029  47 CMS    041  20  20    052  69 OP
006  00  0     018  53  (     030  85  +     042  43 RCL    053  20  20
007  49 PRD    019  48 EXC    031  16 A'     043  02  02    054  43 RCL
008  01  01    020  01  01    032  95  =     044  61 GTO    055  00  00
009  75  -     021  65  ×     033  42 STO    045  00  00    056  92 RTN
010  22 INV    022  01  1     034  02  02    046  30  30
011  59 INT    023  00  0     035  75  -
```

Procedure: Enter any number and press A. A palindromic number is returned. Press R/S to find the number of steps required to reach it.

MORE PALINDROMES - My dictionary defines a palindrome as a word, a phrase, or a sentence which reads the same backward as forward. A commonly cited example sentence is "Able was I ere I saw Elba" which is attributed to Napoleon. I wonder if he really did generate that sentence in English. Sample words which are palindromes are I, aa (a form of solidified lava beloved of crossword puzzle fanciers), did, noon, madam, redder, and reviver. I don't know any palindromic words of more than seven letters. Now here is the challenge. Write down all the palindromic words you can. Give yourself one point for a one letter word, four points for a two letter word, nine points for a three letter word, etc., where the value of each word is the square of the number of its letters. No proper names please. How large a score can you amass? The seven words listed above give you a starting total of 140. My score is over 600.

<u>FAST-GRAFIK-3-D-PLOT</u> - by Peter Poloczek. This superb program was first mentioned in V7N6P6. Then, V7N7/8P7 announced the availability of a 25 page English language version for four dollars in the United States. Now, the Volume 5 Number 1 issue of the TISOFT Newsletter (October,November, December 1982) has reduced the program description and operating instructions to seven pages, a length suitable for inclusion in a newsletter. I tested the program in several ways with outstanding results as illustrated by the figures on page 1 of this issue. I showed the resulting figures to others--every one expressed amazement at that kind of capability from a TI-59/PC-100. Peter has kindly agreed to let us publish the program in TI PPC Notes.

The two part program provides for user selection of the minimum and maximum values for the x, y and z axes, of the number of tapes and print lines, and of the distance between the grid lines. The x axis is in the direction of paper transport. The user can also select the angle between the x and y axes, and whether points that would be hidden by surfaces of the three dimensional figure should be plotted or not. Much of the user control techniques will be recognized as similar to those used with the Superplotter program in the Math/Utilities module. But this program also provides a user-friendly set of prompting commands.

The first part of the program asks for definition of the plotting parameters:

*	X-MAX	Final value for X	
*	X-MIN	Starting value for X	
*	Y-MAX	Final value for Y	
*	Y-MIN	Starting value for Y	
*	Z-MAX	The scale is not correct	
*	Z-MIN		
*	DIF-X	Distance between X grid lines	
*	DIF-Y	Distance between Y grid lines	
*	ZEILEN	The number of lines	
*	STREIFEN	The number of tapes	
*	X-Y-WINKEL	The perspective angle	
*	VERDECKT?	Whether to plot hidden points	1 = no 0 = yes

Each of these inputs is prompted with the print notation shown. The user enters the desired values and presses R/S. The input value is printed, followed by the next prompt. After the last plot control input is entered in response to VERDECKT? the number of function values to be calculated is calculated and printed following the notation F-WERTE or function values. This number provides an estimate of how long the plotting routine will run. It takes about an hour to plot 870 function values.

FAST-GRAFIK-3-D-PLOT (cont)

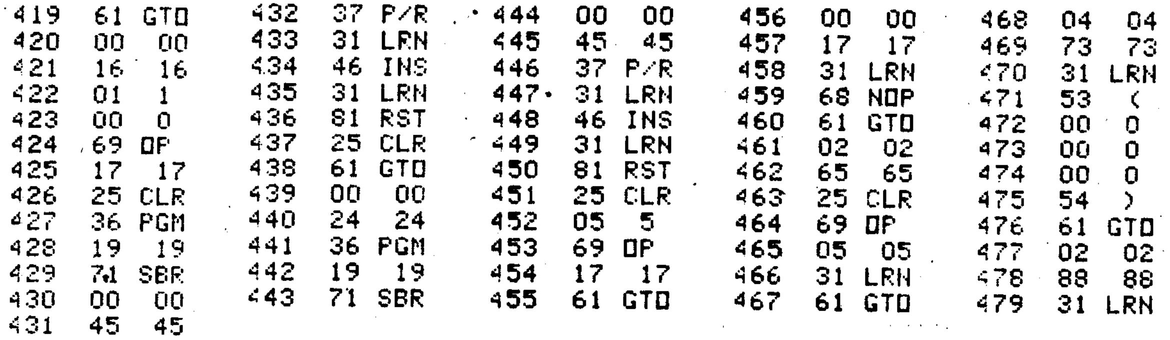
After the F-WERTE is calculated and printed, the program goes on to list the keystrokes needed to initialize the second part of the program for fast mode and high resolution graphics, and prompting sequences for the entry of the function to be plotted. This will be recognized as similar to the prompting Richard Snow provided with his Stars and Stripes program in V6N4-5P8. The three zeroes at steps 472 through 474 of the prompting list are symbolic of the function to be entered, which could include as many as 87 steps, but which must finish with the program sequence GTO 288. The function calls X and Y values from data registers 40 and 41 respectively. The function routine should end with the value for Z in the display. Plotting begins by pressing A.

User Instructions:

1. Enter and record the first part of the program in 6 Op 17. Banks 1 and 2 are required.

2. Enter the second part of the program into memory in 4 Op 17, but record it in 6 Op 17. Banks 1, 2 and 3 are required. If you are like Maurice Swinnen and can't stand the sight of Nop's, then you can use zeroes at locations 158 and 159. Whatever you put in those locations will be pushed out of memory by the sequence which inserts the hexadecimal codes.

3. Enter the first part of the program and press A. The program will respond with the prompt "X-MAX". Enter the desired value and press R/S. The program will respond by printing the entered value followed by the prompt "X-MIN". Continue through the prompting sequence until you enter the response to the prompt "VERDECKT ?". After calculating and printing the number of function values (F-WERTE) the program will print out the following listing and stop with the "F-WERTE" value in the display.



```
419  61 GTO    432  37 P/R    444  00  00    456  00  00    468  04  04
420  00  00    433  31 LRN    445  45  45    457  17  17    469  73  73
421  16  16    434  46 INS    446  37 P/R    458  31 LRN    470  31 LRN
422  01   1    435  31 LRN    447  31 LRN    459  68 NOP    471  53   (
423  00   0    436  81 RST    448  46 INS    460  61 GTO    472  00   0
424  69 OP     437  25 CLR    449  31 LRN    461  02  02    473  00   0
425  17  17    438  61 GTO    450  81 RST    462  65  65    474  00   0
426  25 CLR    439  00  00    451  25 CLR    463  25 CLR    475  54   )
427  36 PGM    440  24  24    452  05   5    464  69 OP     476  61 GTO
428  19  19    441  36 PGM    453  69 OP     465  05  05    477  02  02
429  71 SBR    442  19  19    454  17  17    466  31 LRN    478  88  88
430  00  00    443  71 SBR    455  61 GTO    467  61 GTO    479  31 LRN
431  45  45
```

This printout provides the prompting for the hexadecimal initialization. Enter the second part of the program. Then follow the prompting sequence of keystrokes from the keyboard. That is, once the three card sides of the second part of the program have been entered you begin by pressing GTO 0 1 6 1 0 2nd Op 1 7 CLR 2nd Pgm 1 9 SBR 0 4 5 At this point you will see a flashing 0. Do not clear it but proceed

FAST-GRAFIK-3-D-PLOT (cont)

with the initialization sequence by pressing 2nd P/R. The flashing display will change from a 0. to a 0 (no decimal point). Press LRN and see 016 55 in the display. Press 2nd INS and continue to see 016 55 in the display. Press LRN and see the zero without the decimal point in the display. Press RST CLR GTO 0 2 4 2nd Pgm 1 9 SBR 0 4 5 and again see the flashing 0. (with a decimal point) in the display. Do not clear it and press 2nd P/R to see the flashing zero without the decimal point. Press LRN and see 024 55 in the display. Press 2nd INS and continue to see 024 55 in the display. Press LRN RST CLR and see the zero without the decimal point in the display. Press 5 2nd Op 1 7 GTO 0 1 7 LRN and see 017 01 in the display. Now press 2nd Nop (sorry about that Maurice--you really have to use this Nop) GTO 2 6 5 CLR 2nd Op 0 5 and see 024 25 in the display. This is a critical checkpoint; if you don't have that display you have made a mistake. Press LRN and see 559.49 in the display. Press GTO 4 7 3 LRN and see 473 00 in this display.

At this point you are ready to enter the function you wish to plot. Locations 471 through 478 are symbolic prompting to indicate that you may begin your function with a parenthesis and close your function with a parenthesis, and follow the end of your function with a GTO 288 to return to the main program. The size of the function to be plotted is limited to 87 steps including the GTO 288 by the continuation of the main program at step 560. When you have entered your function followed by GTO 288 press LRN and you are ready to plot. Press A to start plotting.

4. Some rules for setting up the function:

a. The variables used by the function (x,y) are in memory registers 40 (x) and 41 (y).

b. The function routine should end with the calculated value for z in the display.

c. The calculator is in DEG mode when it calls up the function.

d. You can use = in your function, Parentheses not required.

5. Troubleshooting: If you are having problems you can make two quick checks on your hexadecimal initialization. First, as noted in paragraph 3 you must see 024 25 in the display after the GTO 2 6 5 CLR 2nd Op 0 5 part of the initialization sequence. Second, when the initialization is complete you may list the modified program starting at location 000 and see the code listed below. Note there is no location 016 but two locations 017, and that a blank space like an advance replaces location 024. What had been the code at locations 032 and following has been moved down two steps by the two INS commands in the initialization sequence, and the code from what had been locations 016 through 033 is entirely different.

```
000  83 GO*    009  42 STO    018  61 GTO    027  17  17    036  22  22
001  33  33    010  09  09    019  02  02    028  68 NOP    037  42 STO
002  76 LBL    011  61 GTO    020  65  65    029  69 OP     038  36  36
003  11  A     012  00  00    021  25 CLR    030  68  68    039  00  0
004  43 RCL    013  35  35    022  69 OP     031  97 DSZ    040  42 STO
005  00  00    014  86 STF    023  05  05    032  68  68    041  07  07
006  42 STO    015  40 IND            0    033  00  00    042  43 RCL
007  37  37    017  02  02    025  92 RTN    034  92  92    043  27  27
008  25 CLR    017  68 NOP    026  42 STO    035  43 RCL    044  44 SUM
```

FAST-GRAFIX-3-D-PLOT (cont)

6. If you prefer not to have the prompting printed out for the hexadecimal initialization simply replace the GTO at location 396 of the first part with a R/S.

7. A sample problem to plot the function

$$z = 12/(1 + x^2 + y^2)$$

appears at the right.

The plots on the cover sheet were obtained using two strips. The upper plot used the same function listed above, but with DIF-X = 0.25, DIF-Y = 0.5, ZEILEN = 160, and STREIFEN = 2.

The lower plot on the cover sheet used the function

$$z = (x^2 - y^2)/4$$

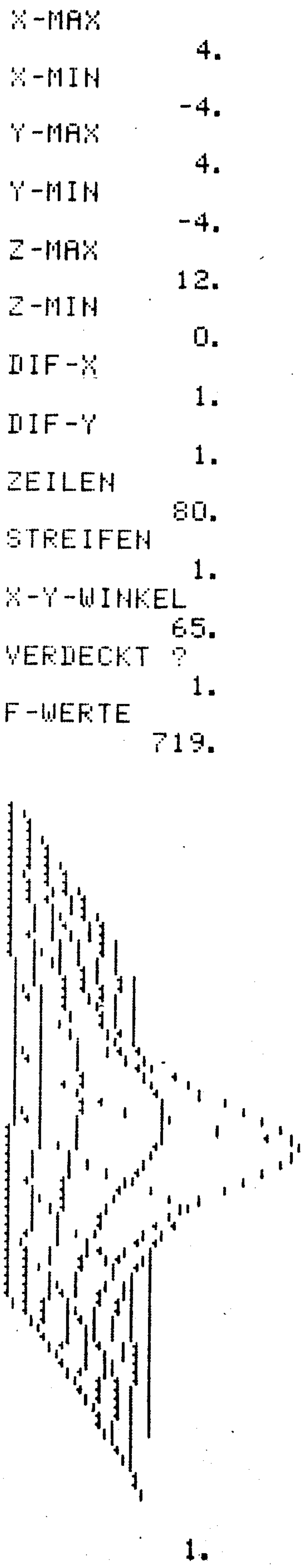
with X-MAX = 8, X-MIN = -8, Y-MAX = 8, Y-MIN = -8, Z-MAX = 16, Z-MIN = -16, DIF-X = 0.5, DIF-Y = 1, ZEILEN = 160, STREIFEN = 2, and X-Y-WINKEL = 65.

The program presented here is for single valued functions. Peter has also provided a multi-function version which will be presented in a future issue. The program is a revision of an original program by Josef Schnieder to include the fast mode capability.

Happy 3-D plotting. Peter requests that you send printouts and data for any particularly nice plots to him. His address is

Peter G. Poloczek
Kalbacher Hauptstrasse 71
D-6000 Frankfurt/M.-56
West Germany

The first part of the program appears on page 15. The second part of the program, before hexadecimal initialization, appears on page 16.



```
X-MAX
            4.
X-MIN
           -4.
Y-MAX
            4.
Y-MIN
           -4.
Z-MAX
           12.
Z-MIN
            0.
DIF-X
            1.
DIF-Y
            1.
ZEILEN
           80.
STREIFEN
            1.
X-Y-WINKEL
           65.
VERDECKT ?
            1.
F-WERTE
          719.
```

```
            1.
```

FAST-GRAPHICS-3-D-PLOT (cont) First Part Program Listing

```
000  76  LBL
001  16  A'
002  69  OP
003  01   01
004  76  LBL
005  17  B'
006  69  OP
007  05   05
008  25  CLR
009  91  R/S
010  99  PRT
011  92  RTN
012  76  LBL
013  11   A
014  69  OP
015  00   00
016  04   4
017  04   4
018  02   2
019  00   0
020  03   3
021  00   0
022  01   1
023  03   3
024  04   4
025  04   4
026  16  A'
027  42  STO
028  12   12
029  04   4
030  04   4
031  02   2
032  00   0
033  03   3
034  00   0
035  02   2
036  04   4
037  03   3
038  01   1
039  16  A'
040  42  STO
041  13   13
042  04   4
043  05   5
044  02   2
045  00   0
046  03   3
047  00   0
048  01   1
049  03   3
050  04   4
051  04   4
052  16  A'
053  42  STO
054  10   10
055  04   4
056  05   5
057  02   2
058  00   0
059  03   3
060  00   0
061  02   2
062  04   4
063  03   3
064  01   1
065  16  A'
066  42  STO
067  11   11
068  04   4
069  06   6
070  02   2
071  00   0
072  03   3
073  00   0
074  01   1
075  03   3
076  04   4
077  04   4
078  16  A'
079  42  STO
080  23   23
081  04   4
082  06   6
083  02   2
084  00   0
085  03   3
086  00   0
087  02   2
088  04   4
089  03   3
090  01   1
091  16  A'
092  42  STO
093  24   24
094  01   1
095  06   6
096  02   2
097  04   4
098  02   2
099  01   1
100  02   2
101  00   0
102  04   4
103  04   4
104  16  A'
105  42  STO
106  18   18
107  01   1
108  06   6
109  02   2
110  04   4
111  02   2
112  01   1
113  02   2
114  00   0
115  04   4
116  05   5
117  16  A'
118  42  STO
119  19   19
120  04   4
121  06   6
122  01   1
123  07   7
124  02   2
125  04   4
126  02   2
127  07   7
128  01   1
129  07   7
130  69  OP
131  01   01
132  03   3
133  01   1
134  52  EE
135  08   8
136  22  INV
137  52  EE
138  69  OP
139  02   02
140  17  B'
141  42  STO
142  22   22
143  03   3
144  06   6
145  03   3
146  07   7
147  03   3
148  05   5
149  01   1
150  07   7
151  02   2
152  04   4
153  69  OP
154  01   01
155  02   2
156  01   1
157  01   1
158  07   7
159  03   3
160  01   1
161  00   0
162  00   0
163  00   0
164  00   0
165  69  OP
166  02   02
167  17  B'
168  42  STO
169  26   26
170  04   4
171  04   4
172  02   2
173  00   0
174  04   4
175  05   5
176  02   2
177  00   0
178  04   4
179  03   3
180  69  OP
181  01   01
182  02   2
183  04   4
184  03   3
185  01   1
186  02   2
187  06   6
188  01   1
189  07   7
190  02   2
191  07   7
192  69  OP
193  02   02
194  17  B'
195  42  STO
196  17   17
197  43  RCL
198  12   12
199  75   -
200  43  RCL
201  13   13
202  95   =
203  42  STO
204  14   14
205  43  RCL
206  10   10
207  75   -
208  43  RCL
209  11   11
210  95   =
211  42  STO
212  15   15
213  65   ×
214  43  RCL
215  17   17
216  39  COS
217  42  STO
218  06   06
219  95   =
220  42  STO
221  16   16
222  43  RCL
223  18   18
224  55   ÷
225  43  RCL
226  06   06
227  95   =
228  42  STO
229  20   20
230  43  RCL
231  19   19
232  65   ×
233  43  RCL
234  06   06
235  95   =
236  42  STO
237  21   21
238  43  RCL
239  14   14
240  85   +
241  43  RCL
242  16   16
243  54   )
244  55   ÷
245  43  RCL
246  22   22
247  95   =
248  42  STO
249  27   27
250  43  RCL
251  23   23
252  75   -
253  43  RCL
254  24   24
255  85   +
256  43  RCL
257  15   15
258  65   ×
259  43  RCL
260  17   17
261  38  SIN
262  42  STO
263  17   17
264  95   =
265  55   ÷
266  05   5
267  09   9
268  93   .
269  09   9
270  09   9
271  55   ÷
272  43  RCL
273  26   26
274  95   =
275  42  STO
276  25   25
277  04   4
278  02   2
279  01   1
280  07   7
281  03   3
282  05   5
283  01   1
284  06   6
285  01   1
286  07   7
287  69  OP
288  01   01
289  01   1
290  05   5
291  02   2
292  06   6
293  03   3
294  07   7
295  00   0
296  00   0
297  07   7
298  01   1
299  69  OP
300  02   02
301  17  B'
302  42  STO
303  00   00
304  43  RCL
305  24   24
306  85   +
307  43  RCL
308  11   11
309  65   ×
310  43  RCL
311  17   17
312  95   =
313  55   ÷
314  43  RCL
315  25   25
316  95   =
317  42  STO
318  23   23
319  43  RCL
320  14   14
321  55   ÷
322  43  RCL
323  18   18
324  85   +
325  01   1
326  95   =
327  65   ×
328  53   (
329  43  RCL
330  16   16
331  55   ÷
332  53   (
333  43  RCL
334  14   14
335  85   +
336  43  RCL
337  16   16
338  95   =
339  85   +
340  53   (
341  43  RCL
342  15   15
343  55   ÷
344  43  RCL
345  19   19
346  85   +
347  01   1
348  54   )
349  65   ×
350  53   (
351  43  RCL
352  14   14
353  55   ÷
354  53   (
355  24  CE
356  85   +
357  43  RCL
358  16   16
359  95   =
360  65   ×
361  43  RCL
362  22   22
363  65   ×
364  69  OP
365  00   00
366  02   2
367  01   1
368  02   2
369  00   0
370  04   4
371  03   3
372  01   1
373  07   7
374  03   3
375  05   5
376  69  OP
377  01   01
378  03   3
379  07   7
380  01   1
381  07   7
382  52  EE
383  06   6
384  22  INV
385  52  EE
386  69  OP
387  02   02
388  69  OP
389  05   05
390  43  RCL
391  26   26
392  95   =
393  59  INT
394  99  PRT
395  98  ADV
396  61  GTO
397  04   04
398  18   18
399  00   0
400  00   0
401  00   0
402  00   0
403  00   0
404  00   0
405  00   0
406  00   0
407  00   0
408  00   0
409  00   0
410  00   0
411  00   0
412  00   0
413  00   0
414  00   0
415  00   0
416  00   0
417  00   0
418  90  LST
419  61  GTO
420  00   00
421  16   16
422  01   1
423  00   0
424  69  OP
425  17   17
426  25  CLR
427  36  PGM
428  19   19
429  71  SBR
430  00   00
431  45   45
432  37  P/R
433  31  LRN
434  46  INS
435  31  LRN
436  81  RST
437  25  CLR
438  61  GTO
439  00   00
440  24   24
441  36  PGM
442  19   19
443  71  SBR
444  00   00
445  45   45
446  37  P/R
447  31  LRN
448  46  INS
449  31  LRN
450  81  RST
451  25  CLR
452  05   5
453  69  OP
454  17   17
455  61  GTO
456  00   00
457  17   17
458  31  LRN
459  68  NOP
460  61  GTO
461  02   02
462  65   65
463  25  CLR
464  69  OP
465  05   05
466  31  LRN
467  61  GTO
468  04   04
469  73   73
470  31  LRN
471  53   (
472  00   0
473  00   0
474  00   0
475  54   )
476  61  GTO
477  02   02
478  88   88
479  31  LRN
```

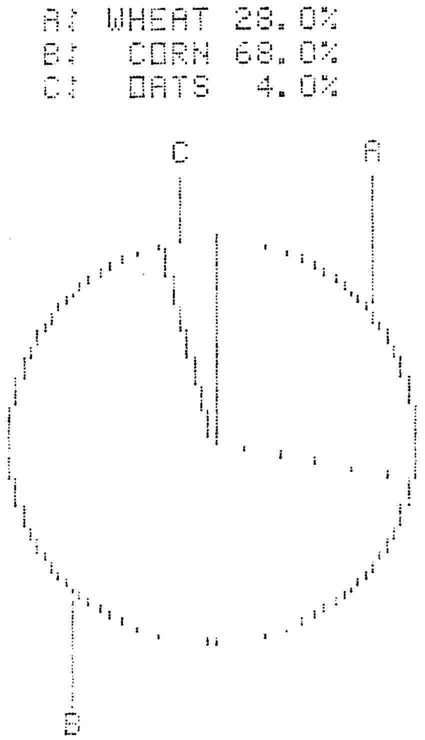
FAST-GRAPHICS-3-D-PLOT (cont) Second Part Program Listing.

```
000  63 GO*
001  33  33
002  76 LBL
003  11  A
004  43 RCL
005  00  00
006  42 STO
007  37  37
008  25 CLR
009  42 STO
010  09  09
011  61 GTO
012  00  00
013  35  35
014  86 STF
015  40 IND
016  51  51
017  00  0
018  00  0
019  00  0
020  00  0
021  00  0
022  00  0
023  74 SM*
024  80  80
025  02  2
026  68 NOP
027  68 NOP
028  68 NOP
029  68 NOP
030  68 NOP
031  00  0
032  92 RTN
033  43 RCL
034  22  22
035  42 STO
036  36  36
037  00  0
038  42 STO
039  07  07
040  43 RCL
041  27  27
042  44 SUM
043  07  07
044  43 RCL
045  11  11
046  42 STO
047  29  29
048  42 STO
049  31  31
050  43 RCL
051  13  13
052  42 STO
053  28  28
054  43 RCL
055  07  07
056  32 X:T
057  43 RCL
058  14  14
059  77  GE
060  00  00
061  70  70
062  04  4
063  69 OP
064  17  17
065  61 GTO
066  05  05
067  60  60
068  43 RCL
069  07  07
070  44 SUM
071  28  28
072  55  ÷
073  43 RCL
074  18  18
075  54  )
076  59 INT
077  65  ×
078  32 X:T
079  43 RCL
080  18  18
081  85  +
082  43 RCL
083  13  13
084  95  =
085  42 STO
086  30  30
087  43 RCL
088  07  07
089  55  ÷

090  43 RCL
091  06  06
092  75  -
093  32 X:T
094  65  ×
095  43 RCL
096  20  20
097  95  =
098  44 SUM
099  31  31
100  05  5
101  69 OP
102  17  17
103  86 STF
104  01  01
105  43 RCL
106  07  07
107  32 X:T
108  43 RCL
109  16  16
110  77  GE
111  01  01
112  21  21
113  43 RCL
114  07  07
115  75  -
116  43 RCL
117  16  16
118  65  ×
119  43 RCL
120  13  13
121  95  =
122  42 STO
123  32  32
124  43 RCL
125  28  28
126  75  -
127  43 RCL
128  32  32
129  54  )
130  55  ÷
131  43 RCL
132  21  21
133  54  )
134  52 EE
135  22 INV
136  52 EE
137  59 INT
138  85  +
139  01  1
140  54  )
141  42 STO
142  05  05
143  43 RCL
144  30  30
145  75  -
146  43 RCL
147  32  32
148  54  )
149  55  ÷
150  43 RCL
151  18  18
152  54  )
153  52 EE
154  22 INV
155  52 EE
156  59 INT
157  85  +
158  68 NOP
159  68 NOP
160  01  1
161  54  )
162  44 SUM
163  05  05
164  43 RCL
165  31  31
166  32 X:T
167  43 RCL
168  29  29
169  77  GE
170  02  02
171  00  00
172  43 RCL
173  28  28
174  42 STO
175  40  40
176  43 RCL
177  29  29
178  42 STO
179  41  41

180  01  1
181  03  8
182  08  8
183  42 STO
184  33  33
185  61 GTO
186  04  04
187  73  73
188  43 RCL
189  21  21
190  22 INV
191  44 SUM
192  28  28
193  43 RCL
194  19  19
195  44 SUM
196  29  29
197  61 GTO
198  02  02
199  23  23
200  43 RCL
201  30  30
202  42 STO
203  40  40
204  43 RCL
205  31  31
206  42 STO
207  41  41
208  02  2
209  01  1
210  04  4
211  61 GTO
212  01  01
213  83  83
214  43 RCL
215  18  18
216  22 INV
217  44 SUM
218  30  30
219  43 RCL
220  20  20
221  44 SUM
222  31  31
223  97 DSZ
224  05  05
225  01  01
226  64  64
227  04  4
228  42 STO
229  38  38
230  00  0
231  63 EX*
232  38  38
233  84 OP*
234  38  38
235  97 DSZ
236  38  38
237  02  02
238  30  30
239  02  2
240  04  4
241  06  6
242  42 STO
243  33  33
244  25 CLR
245  81 RST
246  03  3
247  04  4
248  42 STO
249  33  33
250  71 SBR
251  00  00
252  21  21
253  60 DEG
254  02  2
255  52 EE
256  01  1
257  02  2
258  94 +/-
259  85  +
260  08  8
261  95  =
262  61 GTO
263  00  00
264  14  14
265  22 INV
266  58 FIX
267  25 CLR
268  42 STO
269  38  38

270  42 STO
271  39  39
272  97 DSZ
273  36  36
274  00  00
275  42  42
276  98 ADV
277  69 OP
278  29  29
279  43 RCL
280  09  09
281  99 PRT
282  97 DSZ
283  26  26
284  00  00
285  35  35
286  25 CLR
287  91 R/S
288  29 CP
289  69 OP
290  19  19
291  87 IFF
292  07  07
293  04  04
294  67  67
295  85  +
296  43 RCL
297  41  41
298  65  ×
299  43 RCL
300  17  17
301  95  =
302  55  -
303  43 RCL
304  25  25
305  75  -
306  43 RCL
307  23  23
308  95  =
309  59 INT
310  82 HIR
311  06  06
312  43 RCL
313  37  37
314  67  EQ
315  03  03
316  46  46
317  43 RCL
318  38  38
319  32 X:T
320  82 HIR
321  16  16
322  77  GE
323  03  03
324  38  38
325  32 X:T
326  43 RCL
327  39  39
328  22 INV
329  77  GE
330  04  04
331  64  64
332  32 X:T
333  42 STO
334  39  39
335  61 GTO
336  03  03
337  46  46
338  82 HIR
339  16  16
340  42 STO
341  38  38
342  87 IFF
343  01  01
344  03  03
345  33  33
346  82 HIR
347  16  16
348  55  ÷
349  06  6
350  00  0
351  95  =
352  59 INT
353  32 X:T
354  43 RCL
355  09  09
356  22 INV
357  67  EQ
358  04  04
359  64  64

360  82 HIR
361  16  16
362  75  -
363  43 RCL
364  09  09
365  65  ×
366  06  6
367  00  0
368  54  )
369  42 STO
370  00  00
371  55  ÷
372  03  3
373  54  )
374  59 INT
375  75  -
376  32 X:T
377  53  (
378  43 RCL
379  00  00
380  55  ÷
381  01  1
382  05  5
383  54  )
384  59 INT
385  42 STO
386  08  08
387  69 OP
388  28  28
389  65  ×
390  05  5
391  95  =
392  42 STO
393  34  34
394  43 RCL
395  00  00
396  75  -
397  03  3
398  65  ×
399  32 X:T
400  75  -
401  01  1
402  95  =
403  42 STO
404  00  00
405  50 I×I
406  65  ×
407  02  2
408  05  5
409  75  -
410  69 OP
411  30  30
412  43 RCL
413  00  00
414  69 OP
415  10  10
416  65  ×
417  02  2
418  95  =
419  42 STO
420  35  35
421  29 CP
422  73 RC*
423  08  08
424  55  ÷
425  53  (
426  05  5
427  75  -
428  43 RCL
429  34  34
430  54  )
431  22 INV
432  28 LOG
433  33 X²
434  52 EE
435  22 INV
436  52 EE
437  82 HIR
438  07  07
439  95  =
440  22 INV
441  59 INT
442  65  ×
443  01  1
444  00  0
445  00  0
446  95  =
447  59 INT
448  22 INV
449  67  EQ

450  04  04
451  70  70
452  82 HIR
453  17  17
454  55  ÷
455  01  1
456  00  0
457  00  0
458  65  ×
459  43 RCL
460  35  35
461  95  =
462  74 SM*
463  08  08
464  22 INV
465  86 STF
466  01  01
467  22 INV
468  86 STF
469  07  07
470  25 CLR
471  83 GO*
472  33  33
473  00  0
474  00  0
475  00  0
476  00  0
477  00  0
478  00  0
479  00  0
480  00  0
481  00  0
482  00  0
483  00  0
484  00  0
485  00  0
486  00  0
487  00  0
488  00  0
489  00  0
490  00  0
491  00  0
492  00  0
493  00  0
494  00  0
495  00  0
496  00  0
497  00  0
498  00  0
499  00  0
500  00  0
501  00  0
502  00  0
503  00  0
504  00  0
505  00  0
506  00  0
507  00  0
508  00  0
509  00  0
510  00  0
511  00  0
512  00  0
513  00  0
514  00  0
515  00  0
516  00  0
517  00  0
518  00  0
519  00  0
520  00  0
521  00  0
522  00  0
523  00  0
524  00  0
525  00  0
526  00  0
527  00  0
528  00  0
529  00  0
530  00  0
531  00  0
532  00  0
533  00  0
534  00  0
535  00  0
536  00  0
537  00  0
538  00  0
539  00  0

540  00  0
541  00  0
542  00  0
543  00  0
544  00  0
545  00  0
546  00  0
547  00  0
548  00  0
549  00  0
550  00  0
551  00  0
552  00  0
553  00  0
554  00  0
555  00  0
556  00  0
557  00  0
558  00  0
559  00  0
560  43 RCL
561  12  12
562  42 STO
563  30  30
564  43 RCL
565  07  07
566  75  -
567  43 RCL
568  14  14
569  54  )
570  55  -
571  32 X:T
572  43 RCL
573  06  06
574  95  =
575  44 SUM
576  31  21
577  32 X:T
578  55  ÷
579  43 RCL
580  21  21
581  54  )
582  52 EE
583  22 INV
584  52 EE
585  59 INT
586  85  +
587  01  1
588  54  )
589  65  ×
590  32 X:T
591  43 RCL
592  21  21
593  94 +/-
594  85  +
595  43 RCL
596  07  07
597  95  =
598  44 SUM
599  28  28
600  32 X:T
601  65  ×
602  43 RCL
603  19  19
604  95  =
605  44 SUM
606  29  29
607  61 GTO
608  01  01
609  02  02
610  00  0
611  00  0
```

The zeroes at locations 473 through 559 provide space for the function to be plotted.

<u>CLARIFICATION</u> - Albert Smith of Brooksville, Florida wrote that he could not obtain the pie chart printout on V7N10P7. In the heading for the chart he did not get the "NO 01" notation. I had experienced the same problem but had not investigated since the pie chart itself was good. I assumed that the published program might be somewhat different from the program used to make the illustration. But then the March-April issue of <u>PGM-15</u> arrived from Denmark. They had the same pie chart program with an example which made everything clear. Both Albert and I had misread the instructions for item 6 on V7N10P6 to mean that we were to enter the letter code by which each segment is identified, not a five letter message such as the "NO 01" mentioned above. This suggests that the instructions for item 6 on V7N10P6 might be more clearly written as:

> 6. Enter the print code for the sector descriptor for the first sector (it will be indicated on the pie chart as sector A). Up to five characters may be entered using The TI-59 print code.



```
A: WHEAT 28.0%
B:  CORN 68.0%
C:  OATS  4.0%
```

A copy of a more limited pie chart than that on V7N10P7 is illustrated above. The descriptors used, which I added to Albert's selection of the percentages, substantiate the old saying that you can take the boy out of the country, but you can't take the country out of the boy. There is one other minor error in the pie chart documentation. The definitions of tracks 2.1 and 2.2 are reversed in the text on V7N10P6, but correct on the listing on V7N10P7. Of course Robert Prins' fail-safe programming techniques prevent any adverse effect from that error.

<u>A FIX IND QUIRK</u> - Patrick Acosta. In the article "Flag Sorting" in the January 1979 issue of <u>PPX Exchange</u> William Bowman stated "When performing the IF FLAG INDIRECT XX instruction the TI-59 only recognizes the first digit to the left of the decimal point of the number stored in the indirect register XX. For example, if 96124.03129 is stored in indirect register XX, *IFF *IND XX tests for flag 4 ..." But in the article "There's Gold in Those Guard Digits" in the May/June 1982 issue of <u>PPX Exchange</u> Palmer Hanson reported that the STF IND XX sequence sets a flag according to the ones digit if the value in register XX is an integer of twelve digits or less. If the value in register XX is an integer of thirteen digits then the STF IND XX function sets a flag on the basis of the tens digit.

A similar quirk affects the FIX IND XX command sequence. Thus with 96124.03129 in memory register 01 the sequence FIX IND 01 will set the Fix 4 display mode. But with the thirteen digit number 1234567890124 in register 01 (make it by the sequence 1234567890 x 1000 + 124 =) the sequence FIX IND 01 will set the Fix 2 display mode.

A SHORT PAUSE ON THE 58C? - Pierre Flener writes: I noticed that some of my older programs became impossible to use as soon as there were Pause instructions in them. Knowing the "official" length of a Pause command is ½ second, I took my stop watch, put 10 Pause commands in memory and started this small program. The result was a surprising 2.1 seconds to complete the program, that is 0.21 seconds per Pause command. Previously, I had read a letter in the French magazine "L'ordinateur de Poche" where a Frenchman complained about the same defect on his TI-58C. Up to now I have tested four other TI-58C's of ages from six months to three years and find the execution times for Pause commands to be from 0.18 to 0.25 seconds. This short Pause can cost steps to get a usable Pause function, but can be useful in writing reaction time programs.

Editor's Note: I tested for the duration of Pause commands on my TI-58C and on my TI-59, using a sequence of 100 Pause commands. The TI-59 will complete the sequence in 45 seconds, while the TI-58C will complete the sequence in only 16 seconds, or 0.16 seconds per Pause command. My TI-58C is serial number 8172257. I purchased it in late September 1982, less than six months ago. Some careful reading of Personal Programming shows that the 1977 issue for the TI-58 (not C) and TI-59 states on page V-44 "PAUSE - When encountered during program execution, causes the current value of the display register to be displayed for approximately ½ second ...". The underlining is mine. But the same page number of the 1979 version of Personal Programming for the 58C/59 states "PAUSE - When encountered during program execution, causes the current value of the display register to be displayed for a portion of a second ...". Again, the underlining is mine. What does it all mean? Pierre's comment implies that at one time his TI-58C yielded longer Pause commands. Does anyone else have similar experience?

MORE ON EXECUTION TIME - V7N7/8P10 presented a table of execution times for the TI-59 and a benchmark test of calculators devised by Richard Nelson. As noted on page 2 of this issue it seems that there was a misprint in the description of the benchmark program where Maurice Swinnen thinks that Richard's test must have been to let the calculators count to 100, not to 200 as indicated in V7N7/8P10. I suspect that what Richard did is to program 200 steps, which would count to 100, since there is one step for each + sign and one step for each "1". If those comments are correct, and Maurice's rerun on the HP-41, TI-59 and TI-88 are correct, then it seems that the TI-88 was going to provide no substantial advantage over the TI-59 operating in fast mode. But remember that the discussion in V7N7/8P10 cautioned against putting too much importance on this single test.

Patrick Acosta, Pierre Flener and others have questioned the accuracy of the table of TI-59 execution times on V7N7/8P10. They report that execution time is a function of the value in the display, and that the execution of the Op commands can hardly be described by a single number. On the next page I have listed the results of some simple tests of execution time for a variety of programmable TI calculators. Execution time is affected both by the values used and by the display mode.

MORE ON EXECUTION TIME - (cont)

To test execution time I programmed a series of repetitious commands starting at location 000, started execution with a RST R/S and found the number of cycles which could be completed in one minute. For the TI-58C and TI-59 I used 100 cycles before the RST. For other units which did not have sufficient memory to support 100 cycles I used as many cycles as possible before ending with a RST. For the TI-55II it was necessary to manually press the R/S after each RST had occurred. The table to follow will show variations in execution speed of up to eight per cent depending on changes in the display mode or in the number used in the calculations. I tested only two kinds of calculations; the first case was the 1 + 1 + ... + 1 + RST sequence similar to the test used by Richard Nelson; for the second case I used the sequence SUM 01 SUM 01 .... SUM 01 RST, but with different values in the display. For case 2A I used a one in the display. For case 2B I used a 1001001 in the display, and for case 2C I used the value 1.001001 in the display. The results for various calculators in various configurations were in terms of the number of repetitions in one minute:

Calculator/configuration		Case 1	Case 2A	Case 2B	Case 2C
TI-MBA	Turnon	348	386	382	475
	Fix 0	365	425	410	425
TI-55II	Turnon	290	173		
TI-57	Turnon	410	715	715	835
	Scientific Mode	423	725	859	850
TI-57LCD	Turnon	373	384	400	395
	Scientific Mode	373	384	400	395
TI-58C with PC-100C					
	Turnon	646			
	Scientific Mode	670			
TI-58C without printer					
	Turnon	668	444	358	438
	Scientific Mode	695	426	416	417
TI-59 with PC-100C					
	Turnon	790			
	Scientific Mode	819			
TI-59 without printer					
	Turnon	812	625	491	603
	Scientific Mode	850	599	589	579
	Engineering Mode	800	582	562	566
	Fix 0	775	576	481	573
SR-56	Turnon	770	710		
	Scientific Mode	848			

These comparisons clearly illustrate the execution speed advantage of the TI-59 over the TI-58C, typically a factor of 1.2 to 1.4. In a future issue we will explore the relative speeds of the Op 01 through Op 04 commands and the HIR 05 through HIR 08 commands in loading the print registers.

BOOK REVIEWS:

STATISTICS BY CALCULATOR. Solving statistics problems with the programmable calculator. Peter W. Zehna & Donald R. Barr. Prentice-Hall, Englewood Cliffs, NJ 07632, 1982. 308 pages. Paper back, $13.95.

This book contains a wealth of good TI-59 programs to solve all sorts of statistics problems. Although the book aims mainly at classroom use, anybody who has had at least first year calculus should be able to derive enormous benefits from it. The chapters in the book are:

1. About the book.
2. Sampling
3. Data processing.
4. Estimation.
5. Hypothesis testing.
6. Bivariate populations.
7. Proportions.
8. Analysis of variance.
9. Simple linear regression.
10. Multiple regression.

All the problems in the book are nicely worked out by means of the many programs. Answers are given to the practice problems at the end of each chapter. Peter Zehna teaches mathematics at the Naval Postgraduate School in Monterey, California. Another great book in the growing number on the practical use of the TI-59. It sure is worth its modest price of $13.95. (Maurice E. T. Swinnen.)

CURVE FITTING FOR PROGRAMMABLE CALCULATORS - Second Edition. William M. Kolb. Imtec, P.O. Box 1402, Bowie, MD 20716, 1983. 143 pages, spiral bound. $14.95 shipping included.

This book gives limited attention to theory and concentrates on formulas, graphs and sample problems for forty different curves. The majority of the curves are for one independent variable. Appendixes provide curve fitting programs for specific calculators including nineteen curve programs for the HP-41C/V and HP-75, an eight curve program for the Sharp PC-1211/TRS-80 PC-1, and a nine curve program for the TI-59 by Maurice Swinnen, erstwhile editor of TI PPC Notes. Bar codes are provided for the HP-41CV. There are instructions in the book for ordering cards or cassettes for the HP devices and magnetic cards for the TI-59.

The TI-59 program is an extension of Maurice's program for fitting data to curves which appeared in V7N1/2 of TI PPC Notes, modified to be more friendly to the user, and to add a capability to fit a linear-hyperbolic function ($Y = a + bX + c/X$). An automatic selection of the "best fitting" curve based on the coefficient of determination is also provided.

The book provides a handy reference for the curve fitting formulas to be used for a wide variety of functions. The major deficiency is a carryover from an earlier edition--the formulas are written in terms of register assignments for the nineteen curve HP-41 program--an inconvenience for TI-59 users. Users of the TI-59, the Sharp PC-1211/TRS-80 PC-1 and the HP-75 should specify the second edition when ordering this book. (Palmer O. Hanson, Jr.)

REHASHED QUIRKS - Dr. D. M. Graham of Vancouver, B.C. In V5N8P11 Patrick Acosta reported an unusual fix mode could be set as a result of the keyboard sequence STF IND 3 LIST where the 3 can be any digit, provided that the contents of the corresponding memory register is equal to or less than zero. Dr. Graham reports that the resulting display mode is really Fix 10, not Fix 9 provided there is room for it. To see this effect put 5.E-8 in R10, 5.E-9 in R11, and so on up to 5.E-13 in R15. Then depending on whether you are in the normal Fix 9 or INV Fix mode, which is really a floating point mode or in the special mode set by the sequence defined above you will recall the following values to the display:

	Register Contents	Normal Fix 9 Mode	"Fix 10" Mode
R10	5.E-8	0.00000005	.0000000500
R11	5.E-9	0.000000005	.0000000050
R12	5.E-10	.0000000005	.0000000005
R13	5.E-11	.0000000001	.0000000001
R14	5.E-12	5.-12	0.000000000
R15	5.E-13	5.-13	0.000000000

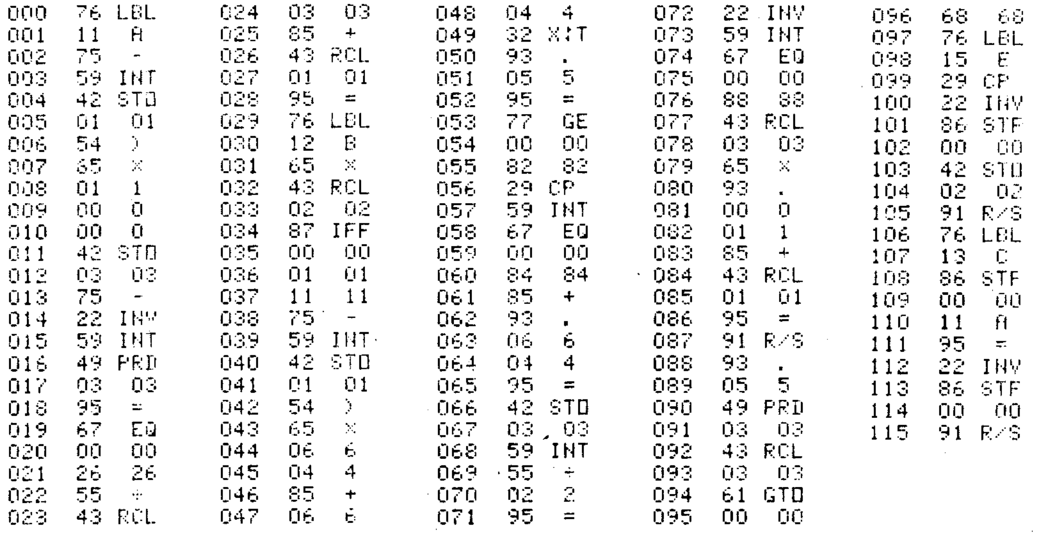
The failure to switch automatically to the scientific format when the register value becomes less than one-half of the smallest value which can be displayed is consistent with all of the normal fix modes except the "floating point" turn-on mode for the TI-59, otherwise known as Fix 9 and INV FIX.

A TI-55II TRICK - Stu Smith. One of the limitations of the TI-55II is that the memory registers used for the statistics routines cannot be accessed during the statistics mode. Even a return to normal mode does not permit examination of the contents of the memory registers used in statistics operations, since the 2nd CSR used to leave statistics mode clears memory registers 3 through 7. During discussions of the statistics quirk of the TI-55II (V8N1P26) with Stu Smith of TI Software Communications he told me of a trick which can be used to avoid the use of 2nd CSR to leave statistics mode and permit recall of the numbers accumulated in memory registers 3 through 7. The sequence is:

1. Press and hold R/S, $\sqrt{x}$, and OFF simultaneously. You will see a blank display.
2. While holding those three keys down, press and hold ON/C. See -8.⊦].].8.8.8 -88 in the display.
3. While ON/C down, let up on R/S, $\sqrt{x}$, and OFF. The display does not change.
4. Release ON/C. See -00000000 00 in the display.
5. Press ON/C once and see 1.1111111 11 in the display.
6. Press ON/C a second time and see Error in the display.
7. Press ON/C a third time and see a 0 in the display. The calculator is out of statistics mode partitioned at 0.8. You can now operate with the memory registers used during statistics mode, but cannot return to statistics mode for further statistics calculations. You will not find that the values in registers 3 through 7 make sense in terms of what you are used to with the TI-58/59. Remember that V8N1P26 had explained that a different statistics algorithm is used in the TI-55II.

<u>TRACING UNDER PROGRAM CONTROL</u> - George Wm. Thomson. Debugging TI-59 and indeed all calculator and computer programs is a difficult and time consuming job. There are many devices which help. For very short programs we can depress the TRACE button on the printer and follow the arithmetic step by step. Or we can insert stops using R/S and maybe a label and examine the contents of registers using INV LIST, being careful at our restarts. Or, good with larger programs, we can run at normal speed down to a stop and TRACE from the printer until the next stop, and so on, with endless variations. But how often I have cried for tracing under program control. How to do it was in the book all right, though not in the index: Page IV-65 of <u>Personal Programming</u>, but for some reason I missed it. And I found it in the notes to David Lobbestael's Profile Plot program (V8N1P24), a tip worth a year's subscription. To trace, set Flag 9. To return to normal printing, reset Flag 9.

<u>REVISED DRAFTSMAN SCALER</u> - Harold Copping. This revision to Richard Snow's program from V5N6P9 adds a Label C mode which will convert a fractional input to a decimal input. The program is



```
000  76 LBL    024  03  03    048  04  4     072  22 INV    096  68  68
001  11  A     025  85  +     049  32 X:T    073  59 INT    097  76 LBL
002  75  -     026  43 RCL    050  93  .     074  67  EQ    098  15  E
003  59 INT    027  01  01    051  05  5     075  00  00    099  29 CP
004  42 STO    028  95  =     052  95  =     076  88  88    100  22 INV
005  01  01    029  76 LBL    053  77  GE    077  43 RCL    101  86 STF
006  54  )     030  12  B     054  00  00    078  03  03    102  00  00
007  65  ×     031  65  ×     055  82  82    079  65  ×     103  42 STO
008  01  1     032  43 RCL    056  29 CP     080  93  .     104  02  02
009  00  0     033  02  02    057  59 INT    081  00  0     105  91 R/S
010  00  0     034  87 IFF    058  67  EQ    082  01  1     106  76 LBL
011  42 STO    035  00  00    059  00  00    083  85  +     107  13  C
012  03  03    036  01  01    060  84  84    084  43 RCL    108  86 STF
013  75  -     037  11  11    061  85  +     085  01  01    109  00  00
014  22 INV    038  75  -     062  93  .     086  95  =     110  11  A
015  59 INT    039  59 INT    063  06  6     087  91 R/S    111  95  =
016  49 PRD    040  42 STO    064  04  4     088  93  .     112  22 INV
017  03  03    041  01  01    065  95  =     089  05  5     113  86 STF
018  95  =     042  54  )     066  42 STO    090  49 PRD    114  00  00
019  67  EQ    043  65  ×     067  03  03    091  03  03    115  91 R/S
020  00  00    044  06  6     068  59 INT    092  43 RCL
021  26  26    045  04  4     069  55  ÷     093  03  03
022  55  ÷     046  85  +     070  02  2     094  61 GTO
023  43 RCL    047  06  6     071  95  =     095  00  00
```

User Instructions:

1. Enter any scale in decimal form and press E.

2. Enter an actual dimension in fractional form I.NNDD where I is the integer part, NN is a two digit numerator, and DD is a two digit denominator. One digit numerators or denominators must be entered with a leading zero. Press A to obtain the draftsman format, rounded to the nearest 64th of an inch and reduced to lowest terms. Again, the output format is I.NNDD.

3. Enter an actual dimension in decimal form I.dddddd... where I is the integer part and ddddd... is the decimal part. Press B to obtain the draftsman format.

4. Enter a fractional format I.NNDD. With the scale set at 1, press C and obtain a decimal format I.dddd... .

SYNTHETIC CODES ON A TI-57 - Dejan Ristanovic. The little TI-57 still has a lot of secrets. It seems obvious that it uses some byte arrangement for instruction coding that leaves a lot of "spare" bytes. Every computer, however, has to react in some way to any instruction, so let's examine the reaction to some synthetic codes.

To synthesize some of the new codes for the TI-57 at program location KK + 4, you first enter the following program:

```
KK+0   02     2
KK+1   38     EXC SST
KK+2   86 0   LBL 0
KK+3   00     0
KK+4   00     0
```

Step KK+1 is most interesting. Press EXC SST and the TI-57 is tricked such that a 38 code will remain, but without an address part. With the short program in place you set up to synthesize code at location KK+4 with the keyboard sequence GTO 2nd KK SST SST SST. The program pointer will be at KK+4. At this point there are three ways to continue:

(1) You can press LRN and, after that, any digit between 2 and 7 included. This sequence seems to synthesize a hexadecimal code On where n can be one of the hexadecimal digits (A, B, C, D, E, or F). If you press 2, for example, you will synthesize the OA code. This code seems to be the ordinary digit-entering line that enters digit A. The OA code will be displayed as the letter A. The same correspondence holds for the other digits B through F. This can be very useful since you can write text on the display, providing the text consists of the letters A through F and some of the numbers which look like letters (zero for O, for example). One might generate the word FIASCO as a comment for some games. This "text writing" was left intact intentionally by the TI-57 designer. The proof for this is the fact that the letters A, C, E, and F are block capitals while the letters B and D are not, because it would be impossible to differentiate them from the numbers 8 and zero. Unfortunately, you can not do arithmetic with hex-numbers. Number 1B, for example, acts as 21--the same situation holds with the TI-59 by the way.

(2) You can press EE LRN and enter any digit between 0 and 7. With this option you will synthesize codes m2 where m = n + 1. These are codes of the instructions in the second column of the keyboard (INV, STO, EE ...). Number m represents the row in which the digit is. For example, if you press digit 0 (n) then m = 1 and code 12 results.

(3) You can press . LRN n where n is any digit from 0 through 7. With this option you synthesize codes m1 where again m = n + 1. With this option you can synthesize some hex-codes of the instructions which normally cannot remain in a program (2nd, LRN, etc.).

SYNTHETIC CODES ON A TI-57 (cont)

Let's take a look at some of the useful hex-codes:

(1) The sequence obtained by pressing GTO SST or SBR SST will first search for the first instruction 00 in the program. Program execution will start from the instruction placed immediately following this one.

(2) Sequence 51 On will do the same, but program execution will be transferred to the first On instruction in the program.

(3) FIX SST On (n being A, B, C, D, E or F) has some useful meanings. For example, having n = F will put the TI-57 into a "FIX -1" state useful for saving program and date while the display is off to save energy.

(4) Sequence 11 = will return the last number that was in the display before CLR was pressed. This is very interesting but not too useful.

(5) Sequence = 12 will, providing that the calculator is in EE mode and in Fix n, divide the number in the display register by 10**(9-n).

(6) Sequence 12 11 EE INT will return the last three digits of the number in the display register providing the calculator is in EE and Fix mode.

(7) Sequence CLR 12 has some interesting effects, useful mostly for having fun with the display. For example, if the TI-57 is in Fix -1, you can press +/- +/- . and only a point will remain in the display.

To illustrate some of these effects here is a short program which generates random numbers between 0 and 999 (noone could do it in less steps without the new codes):

```
LBL 0 RCL 0 Lnx 12 11 EE Int STO 0 RTN
```

The seed in R0 should be positive. I hope these hex-codes will inspire a lot of searching. Take those TI-57's out of your dark desk drawers. I thank Sinisa Djurekovic from Zagreb, Yugoslavia for bringing my attention to these new codes.

Editor's Notes: I have not had time to digest all of this, or even to fully verify the effects listed above. My experience to date shows that:

(1) These effects occur with the TI-57 but not with the TI-57LCD.

(2) You can move these codes around with Ins and Del commands and the hexadecimal integrity is maintained. This is not the case with hex-codes on the TI-58C and TI-59.

(3) For option 2 and n = 0 through 3 I get the hex-codes -12, -22, -32, and -42 where in TI-57 language the minus sign usually indicates the inverse function. For n of 4 through 7 I get 51 7, 51 4, 51 1, and 51 0. The addresses of these GTO commands are recognized as the numerical keys of the second column of the keyboard in descending order.

SYNTHETIC CODES ON A TI-57 (cont)

(4) The initialization program on page 24 will work equally well with either a PRD SST or a FIX SST in place of the EXC SST.

(5) If you replace the 2 at KK+0 of the initialization program with a zero then the EE LRN and . LRN n options will work to make hex-codes. The minus signs mentioned in paragraph 3 above do not appear, but the various effects discussed by Dejan still occur. It seems that the program does not care if the hex-codes are positive or negative (inverse). With a zero at KK+0 the first option which generates alphanumeric display for the hexadecimal digits does not work. I recommend that you do not use a zero at KK+0 to eliminate the minus signs and intermix the codes so generated with alphanumerics generated with a 2 at KK+0. The reason is simple--you could then generate pairs of hexadecimal code which appear the same in LRN mode but which act entirely different in the run mode. An example is (a) with a 2 at KK+0 do the initialization sequence and press LRN 4 and generate a code 12 which will make a C in the display, and (b) with a zero at KK+0 do the initialization sequence and press EE LRN 0 and generate a code 12 which will not make a C in the display, but will provide the arithmetic functions defined for code 12 in Dejan's discussion.

Entry of a sample program: The random number generator program can be entered in the following manner: We will start with KK+0=8. We can go to LRN and SST to program step 08, or we can use GTO-2nd-0-8-LRN. Starting at step 08 we key in the initialization sequence and press LRN to return to keyboard control. Press GTO-2nd-0-8-SST-SST-SST-.-LRN-0-BST and see 12 11 in the display indicating that we have generated a hex-code 11 at step 12. Press 2nd-Ins and see 12 00 to make room for the code 12. Press LRN to return to keyboard control. Now press GTO-2nd-0-8-SST-SST-SST-EE-LRN-0-BST and see 12 - 12 in the display indicating we have generated a hex-code 12 at step 12. At this point we have hex-code 12 at step 12 and hex-code 11 at step 13. To build the rest of the program we BST to step 11, 2nd-Ins to make room for one more line of code, and enter RCL-0-Lnx and we are at step 13 with the - 12 hex-code in the display. Two SST's take us to step 15 to enter the rest of the program. Press EE-2nd-Int-STO-0-INV-EE-2nd-Pause-GTO-0-LRN. To run the program place a seed in memory register 0. If you use 21 as the seed you will see the three digit numbers 452, 368, 808, 456, 249 .... flashed in the display. You will note that I added the INV-EE-GTO-0 to remove the scientific display for the output number, a pause for readout, and a GTO-0 for iterating the solution. Happy hexadecimal programming on your TI-57.

ZERO TO ZERO POINT ONE - This brainteaser from V8N1P13 asked the reader to start with a zero in the display and obtain 0.1 in the display in a minimum number of steps. Use of the number keys is not allowed. Laurance Leeds submits

Cos DIV INV log =

Myer Boland submits Cos INV log 1/X which accomplishes the task without disturbing any pending operations.

<u>SPEEDY</u> <u>FACTOR</u> <u>FINDERS</u> <u>REVISITED</u> - Patrick Acosta's Modulo 210 Speedy Factor Finder (V6N4/5P13) is the fastest TI-59 factor finder published to date. However, it requires four card sides, and unusual keystrokes are required to extend its range to eleven or twelve digits as described in V8N1P10. Two new modulo 30 programs are available which use later fast mode entry techniques, permit straightforward entry of 11 or 12 digit integers, and which fit on two card sides. The execution speeds are significantly faster that previously published programs such as those in V6N1P5, but not as fast as the modulo 210 SFF.

The first program, which is listed on page 28, is by Patrick Acosta. It is primarily designed for use with a printer. To operate without a printer one must either try to catch the factors as they are flashed with Pause commands, or replace the PRT-PAU sequences at locations 202/203 and 232/233 with PAU-R/S sequences. The listing on page 28 is <u>before</u> fast mode initialization. Note that the return addresses in memory registers 00 through 15 must be entered as well. The fast mode initialization sequence is:

9 Op 17 CLR GTO 016 Pgm 19 SBR 045 P/R LRN Ins LRN RST CLR.

A listing of the first 30 program steps after initialization is shown in the right hand column on page 28. Once the initialization is complete, simply enter the integer to be tested in the display register and press A. The integer and its factors will be printed. The program stops with a flashing 1 in the display. This program can also be run on a TI-58.

The second program, which is listed on page 27, comes from some new talent applied to the SFF problem. Laurance Leeds program fits on two card sides, stores the factors as they are found for later recall but does not print them, provides a mode for recall of the input integer including viewing all digits of an eleven or twelve digit integer, and provides a mode for recalling the factors. To use his program:

(1) Enter the integer to be tested. Press A to initialize fast mode. Ignore the flashing 1. Press 7 EE to run the program. A flashing zero shows that the solution is complete.

(2) To see the first factor press B. Press R/S to see additional factors. A flashing zero indicates that all the factors have been displayed.

(3) To verify the input integer press C. The integer will be displayed. To view all the digits of 11 or 12 digit integers, continue by pressig R/S and see the digits down to the 10^6 digit. Press R/S again and see a 0. followed by the six lower order digits.

<u>Execution</u> <u>speeds</u>:	<u>111111111111</u>	<u>103569859</u>	<u>987654321</u>	<u>9999999967</u>
Leeds	17 sec	46 sec	61 sec	2 hr 31 min
Acosta TI-59	21 sec	50 sec	65 sec	2 hr 33 min
Acosta TI-58C	27 sec	61 sec	79 sec	3 hr 6 min

Laurance reports that the use of HIR arithmetic is not important when the integers tested are small, but yields up to a one percent speed increase when the numbers are large.

SPEEDY FACTOR FINDERS REVISITED (cont) Leeds Program Listing

```
000  91 R/S    080  95  =     160  18  18    240  67  EQ    320  61 GTO    400  25 CLR
001  29 CP     081  22 INV    161  55  ÷     241  04  04    321  01  01    401  69 OP
002  61 GTO    082  59 INT    162  82 HIR    242  00  00    322  17  17    402  99  99
003  00  00    083  67  EQ    163  17  17    243  29 CP     323  82 HIR    403  81 RST
004  21  21    084  02  02    164  95  =     244  61 GTO    324  17  17    404  82 HIR
005  76 LBL    085  47  47    165  22 INV    245  00  00    325  72 ST*    405  18  18
006  11  A     086  02  2     166  59 INT    246  54  54    326  07  07    406  72 ST*
007  47 CMS    087  82 HIR    167  67  EQ    247  82 HIR    327  69 OP     407  07  07
008  82 HIR    088  37  37    168  03  03    248  17  17    328  27  27    408  61 GTO
009  08  08    089  82 HIR.   169  61  61    249  72 ST*    329  82 HIR    409  04  04
010  42 STO    090  18  18    170  06  6     250  07  07    330  68  68    410  00  00
011  50  50    091  55  ÷     171  61 GTO    251  69 OP     331  01  1     411  76 LBL
012  25 CLR    092  82 HIR    172  00  00    252  27  27    332  32 X:T    412  13  C
013  82 HIR    093  17  17    173  52  52    253  82 HIR    333  82 HIR    413  25 CLR
014  07  07    094  95  =     174  02  2     254  68  68    334  18  18    414  58 FIX
015  08  8     095  22 INV    175  72 ST*    255  01  1     335  67  EQ    415  00  00
016  42 STO    096  59 INT    176  07  07    256  32 X:T    336  04  04    416  43 RCL
017  07  07    097  67  EQ    177  69 OP     257  82 HIR    337  00  00    417  50  50
018  61 GTO    098  02  02    178  27  27    258  18  18    338  29 CP     418  91 R/S
019  04  04    099  66  66    179  82 HIR    259  67  EQ    339  61 GTO    419  55  ÷
020  65  65    100  04  4     180  68  68    260  04  04    340  01  01    420  01  1
021  82 HIR    101  82 HIR    181  01  1     261  00  00    341  31  31    421  52 EE
022  18  18    102  37  37    182  32 X:T    262  29 CP     342  82 HIR    422  06  6
023  55  ÷     103  82 HIR    183  82 HIR    263  61 GTO    343  17  17    423  22 INV
024  02  2     104  18  18    184  18  18    264  00  00    344  72 ST*    424  52 EE.
025  95  =     105  55  ÷     185  67  EQ    265  75  75    345  07  07    425  75  -
026  22 INV    106  82 HIR    186  04  04    266  82 HIR    346  69 OP     426  59 INT
027  59 INT    107  17  17    187  00  00    267  17  17    347  27  27    427  91 R/S
028  67  EQ    108  95  =     188  29 CP     268  72 ST*    348  82 HIR    428  95  =
029  01  01    109  22 INV    189  61 GTO    269  07  07    349  68  68    429  58 FIX
030  74  74    110  59 INT    190  00  00    270  69 OP     350  01  1     430  06  06
031  82 HIR    111  67  EQ    191  21  21    271  27  27    351  32 X:T    431  91 R/S
032  18  18    112  02  02    192  03  3     272  82 HIR    352  82 HIR    432  61 GTO
033  55  ÷     113  85  85    193  72 ST*    273  68  68    353  18  18    433  04  04
034  03  3     114  02  2     194  07  07    274  01  1     354  67  EQ    434  14  14
035  95  =     115  82 HIR    195  69 OP     275  32 X:T    355  04  04    435  00  0
036  22 INV    116  37  37    196  27  27    276  82 HIR    356  00  00    436  00  0
037  59 INT    117  82 HIR    197  82 HIR    277  18  18    357  29 CP     437  00  0
038  67  EQ    118  18  18    198  68  68    278  67  EQ    358  61 GTO    438  00  0
039  01  01    119  55  ÷     199  01  1     279  04  04    359  01  01    439  00  0
040  92  92    120  82 HIR    200  32 X:T    280  00  00    360  45  45    440  00  0
041  82 HIR    121  17  17    201  82 HIR    281  29 CP     361  82 HIR    441  00  0
042  18  18    122  95  =     202  18  18    282  61 GTO    362  17  17    442  00  0
043  55  ÷     123  22 INV    203  67  EQ    283  00  00    363  72 ST*    443  00  0
044  05  5     124  59 INT    204  04  04    284  89  89    364  07  07    444  00  0
045  95  =     125  67  EQ    205  00  00    285  82 HIR    365  69 OP     445  00  0
046  22 INV    126  03  03    206  29 CP     286  17  17    366  27  27    446  00  0
047  59 INT    127  04  04    207  61 GTO    287  72 ST*    367  82 HIR    447  00  0
048  67  EQ    128  04  4     208  00  00    288  07  07    368  68  68    448  00  0
049  02  02    129  82 HIR    209  31  31    289  69 OP     369  01  1     449  00  0
050  10  10    130  37  37    210  05  5     290  27  27    370  32 X:T    450  00  0
051  07  7     131  82 HIR    211  72 ST*    291  82 HIR    371  82 HIR    451  00  0
052  82 HIR    132  18  18    212  07  07    292  68  68    372  18  18    452  00  0
053  37  37    133  55  ÷     213  69 OP     293  01  1     373  67  EQ    453  00  0
054  82 HIR    134  82 HIR    214  27  27    294  32 X:T    374  04  04    454  00  0
055  18  18    135  17  17    215  82 HIR    295  82 HIR    375  00  00    455  00  0
056  55  ÷     136  95  =     216  68  68    296  18  18    376  29 CP     456  00  0
057  34 ΓX     137  22 INV    217  01  1     297  67  EQ    377  61 GTO    457  00  0
058  32 X:T    138  59 INT    218  32 X:T    298  04  04    378  01  01    458  00  0
059  82 HIR    139  67  EQ    219  82 HIR    299  00  00    379  59  59    459  00  0
060  17  17    140  03  03    220  18  18    300  29 CP     380  76 LBL    460  00  0
061  95  =     141  23  23    221  67  EQ    301  61 GTO    381  12  B     461  00  0
062  22 INV    142  06  6     222  04  04    302  01  01    382  25 CLR    462  00  0
063  77  GE    143  82 HIR    223  00  00    303  03  03    383  08  8     463  00  0
064  04  04    144  37  37    224  29 CP     304  82 HIR    384  42 STO    464  00  0
065  04  04    145  82 HIR    225  61 GTO    305  17  17    385  07  07    465  02  2
066  29 CP     146  18  18    226  00  00    306  72 ST*    386  29 CP     466  52 EE
067  22 INV    147  55  ÷     227  41  41    307  07  07    387  58 FIX    467  01  1
068  59 INT    148  82 HIR    228  82 HIR    308  69 OP     388  00  00    468  02  2
069  67  EQ    149  17  17    229  17  17    309  27  27    389  73 RC*    469  94 +/-
070  02  02    150  95  =     230  72 ST*    310  82 HIR    390  07  07    470  85  +
071  28  28    151  22 INV    231  07  07    311  68  68    391  67  EQ    471  01  1
072  04  4     152  59 INT    232  69 OP     312  01  1     392  04  04    472  95  =
073  82 HIR    153  67  EQ    233  27  27    313  32 X:T    393  00  00    473  22 INV
074  37  37    154  03  03    234  82 HIR    314  82 HIR    394  91 R/S    474  52 EE
075  82 HIR    155  42  42    235  68  68    315  18  18    395  69 OP     475  58 FIX
076  18  18    156  02  2     236  01  1     316  67  EQ    396  27  27    476  00  00
077  55  ÷     157  82 HIR    237  32 X:T    317  04  04    397  61 GTO    477  60 DEG
078  82 HIR    158  37  37    238  82 HIR    318  00  00    398  03  03    478  86 STF
079  17  17    159  82 HIR    239  18  18    319  29 CP     399  89  89    479  40 IND
```

SPEEDY FACTOR FINDERS REVISITED (cont) Acosta Program Listing

```
000  76 LBL    048  18 18     096  37 37     144  95 =      192  05 5        168.   00
001  11 A      049  82 HIR    097  82 HIR    145  22 INV    193  95 =          0.   01
002  98 ADV    050  18 18     098  18 18     146  59 INT    194  42 STO       50.   02
003  42 STO    051  55 ÷      099  55 ÷      147  67 EQ     195  04 04        63.   03
004  01 01     052  05 5      100  82 HIR    148  01 01     196  73 RC*      154.   04
005  82 HIR    053  95 =      101  17 17     149  83 83     197  04 04        84.   05
006  08 08     054  22 INV    102  95 =      150  06 6      198  42 STO       98.   06
007  22 INV    055  59 INT    103  22 INV    151  82 HIR    199  04 04         0.   07
008  58 FIX    056  67 EQ     104  59 INT    152  37 37     200  82 HIR      112.   08
009  60 DEG    057  01 01     105  67 EQ     153  82 HIR    201  17 17       126.   09
010  93 .      058  83 83     106  01 01     154  18 18     202  99 PRT        0.   10
011  01 1      059  02 2      107  83 83     155  55 ÷      203  66 PAU      140.   11
012  34 ΓX     060  82 HIR    108  04 4      156  82 HIR    204  82 HIR        0.   12
013  33 X²     061  37 37     109  82 HIR    157  17 17     205  68 68         0.   13
014  35 1/X    062  82 HIR    110  37 37     158  95 =      206  83 GO*      154.   14
015  86 STF    063  18 18     111  82 HIR    159  22 INV    207  04 04        16.   15
016  61 61     064  55 ÷      112  18 18     160  59 INT    208  68 NOP        0.   16
017  66 PAU    065  34 ΓX     113  55 ÷      161  67 EQ     209  03 3
018  33 X²     066  32 X:T    114  82 HIR    162  01 01     210  00 0      000  76 LBL
019  63 EX*    067  82 HIR    115  17 17     163  83 83     211  42 STO    001  11 A
020  68 68     068  17 17     116  95 =      164  02 2      212  04 04     002  98 ADV
021  30 TAN    069  95 =      117  22 INV    165  82 HIR    213  02 2      003  42 STO
022  61 GTO    070  22 INV    118  59 INT    166  37 37     214  61 GTO    004  01 01
023  54 )      071  77 GE     119  67 EQ     167  82 HIR    215  02 02     005  82 HIR
024  68 NOP    072  02 02     120  01 01     168  18 18     216  03 03     006  08 08
025  29 CP     073  26 26     121  83 83     169  55 ÷      217  04 4      007  22 INV
026  05 5      074  29 CP     122  02 2      170  82 HIR    218  00 0      008  58 FIX
027  82 HIR    075  22 INV    123  82 HIR    171  17 17     219  42 STO    009  60 DEG
028  07 07     076  59 INT    124  37 37     172  95 =      220  04 04     010  93 .
029  82 HIR    077  67 EQ     125  82 HIR    173  22 INV    221  03 3      011  01 1
030  18 18     078  01 01     126  18 18     174  59 INT    222  61 GTO    012  34 ΓX
031  55 ÷      079  83 83     127  55 ÷      175  67 EQ     223  02 02     013  33 X²
032  02 2      080  04 4      128  82 HIR    176  01 01     224  03 03     014  35 1/X
033  95 =      081  82 HIR    129  17 17     177  83 83     225  01 1      015  86 STF
034  22 INV    082  37 37     130  95 =      178  06 6      226  32 X:T    016  12 12
035  59 INT    083  82 HIR    131  22 INV    179  61 GTO    227  82 HIR    017  68 NOP
036  67 EQ     084  18 18     132  59 INT    180  00 00     228  18 18     018  43 RCL
037  02 02     085  55 ÷      133  67 EQ     181  61 61     229  67 EQ     019  01 01
038  10 10     086  82 HIR    134  01 01     182  82 HIR    230  02 02     020  99 PRT
039  82 HIR    087  17 17     135  83 83     183  17 17     231  35 35     021  61 GTO
040  18 18     088  95 =      136  04 4      184  55 ÷      232  99 PRT    022  00 00
041  55 ÷      089  22 INV    137  82 HIR    185  03 3      233  66 PAU    023  26 26
042  03 3      090  59 INT    138  37 37     186  00 0      234  98 ADV    024  54 )
043  95 =      091  67 EQ     139  82 HIR    187  95 =      235  25 CLR    025  68 NOP
044  22 INV    092  01 01     140  18 18     188  22 INV    236  35 1/X    026  29 CP
045  59 INT    093  83 83     141  55 ÷      189  59 INT    237  01 1      027  05 5
046  67 EQ     094  02 2      142  82 HIR    190  65 ×      238  92 RTN    028  82 HIR
047  02 02     095  82 HIR    143  17 17     191  01 1      239  00 0      029  07 07
```

TABLE OF CONTENTS

MAGNETIC CARD SERVICE

See V8N1P32 for details. One dollar per card plus a stamped and self-addressed envelope. For programs in this issue:

Fast-Grafik-3-D-Plot	$3.00
Acosta Factor Finder	$1.00
Leeds Factor Finder	$1.00

No checks please.

The end of V8N2