TI PPC NOTES

NEWSLETTER OF THE TI PERSONAL PROGRAMMABLE CALCULATOR CLUB

P. O. Box 1421, Largo, FL 34649

Volume 13, Number 2 Second Quarter 1988/1989

The correction of program errors seems to be the dominant theme for this issue. First, there was a translation error in an earlier TI-95 program for extended precision square root calculations. Second, there were several errors in the "smallest circle" TI-74 program published in the previous issue. Third, there was a long-standing error in a program for curve fitting with the TI-59. The common thread in the errors in all three programs was the absence of flashing displays, overflows, and the like. The answers just turned out to be unreasonable for some cases. This only serves to emphasize the need to test as many cases as possible.

Many members had asked for information on the Journal of Recreational Mathematics. The advertisement on page 11, where the publication of an advertisement is admittedly a departure from tradition, should help. The editors of the journal indicate that they well provide a free issue on request. Please remember to mention our club.

The "big news" in the Hanson family is my retirement which was effective December 31. Some members have asked if that means that we can publish on a more regular schedule. My present plans are to publish issues of about 30 pages at three to four month intervals. That would close out the current "year" in October, and allow us to get the subscription year and the calendar year to correspond in the future.

TABLE OF CONTENTS

We continue to provide magnetic card service for the TI-59.

We also will provide magnetic tape service for TI-74 and TI-95 programs. Write for details.

ERRATA

The best excuse I have found to date for the errors in our newsletter can be paraphrased from the note:

> "Any inaccuracies in this index may be explained by the fact that it has been prepared with the help of a computer."

which is from *The Art of Computer Programming, Volume 1 Fundamental Algorithms*", by D. E. Knuth, Addison-Wesley, 2nd Edition, 1973, page 634.

Smallest Circle - V13N1P28/29. There were several errors buried in this program which made the solution sensitive to such things as the sequence in which the data points were entered, the multiple entry of the same point, etc. The details of the errors and an improved program appear in pages 12-15.

Truncation Effects in Extended Precision - V13N1P1. Robert Prins was puzzled by the report that there was a truncation problem in the TI-95 extended precision square root program on V12N2P24 since the problem did not appear with the TI-59 program from which a translation hsd been made. He found that the problem was with the translation which used data register 010 as a substitute for the t-register, when that register was also used in the extended precision calculations. The error can be eliminated by replacing register 010 with register 099 as shown in the following listing. A solution which uses the unprotected mode of the TI-95 to access the t-register is discussed in more detail in pages 8 to 10.

```
0000 CE `SQUARE ROOT N`
0014 ADV PRT CMS CE `1<`
0020 `N<100 ?` BRK ADV
0029 PRT STO 049 CE `# `
0036 `BLOCKS? <41:` BRK
0049 ADV STO 002 10
0055 STO 000 INV LOG
0060 STO 001 1 STO 008
0067 STO 089
0070 LBL 01 90 STO 003
0078 STO 005 50 STO 004
0086 STO 006 RCL 008
0092 STO 007 +/- ST+ 003
0099 ST+ 004
0102 LBL 02 RCL IND 003
0109 INC 003 STO 099
0115 RCL IND 004 INC 004
0122 IF= 099 GTL 02
0128 IF< 099 GTL 05 0
0135 STO 099
0138 LBL 03 INV INC 005
0145 INV INC 006
0149 ST- IND 006
0153 RCL IND 005
0157 ST- IND 006
0161 RCL IND 006 INV
0166 IF< 099 GTO 0180
0172 RCL 001 ST+ IND 006
0179 1. DSZ 007 GTL 03
0187 INC 009 2
0191 LBL 04 ST+ 089
0197 GTL 01
0200 LBL 05 DSZ 000
0206 GTL 06 10 EXC 000
0214 EXC 009 PRT INC 008
0221 DSZ 002 GTL 06 CE
0228 `MORE BLOCKS?` Y/N
0241 GTL 11 GTO 0000
0247 LBL 06 RCL 007
0253 STO 003 10 ST* 009
0261 LBL 07 INV INC 005
0268 ST* IND 005 DSZ 003
0275 GTL 07 x^2
0279 LBL 08 INV INC 006
0286 ST* IND 006 DSZ 007
0293 GTL 08 2 STO 007 50
0302 LBL 09 STO 006
0308 RCL 008 EXC 003 x~t
0315 0 STO 099 x~t
0320 LBL 10 ST- IND 006
0327 INV INC 006 x~t
0332 ST+ IND 006 (
0337 RCL IND 006 /
0342 RCL 001 )( INT *
0349 x~t RCL 001 )
0354 DSZ 003 GTL 10 90
0362 DSZ 007 GTL 09 9
0369 +/- GTL 04
0373 LBL 11 CE `# BLOCK`
0384 `S?` BRK STO 002
0390 GTL 06
```

HARDWARE FOR SALE - TI-58C for $30.00 and PC-100C for $40.00. The PC-100 has the paper slippage problem which can be solved with the "eraser" fix. Send an additional ten dollars to cover shipping of the TI-58C and an additional twenty dollars to cover shipping of the PC-100C. I will return any portion of the shipping which is not used. Make the check to PPC Publications.

HARDWARE FOR SALE - In past issues members have offered used hardware for sale; for example, the V12N2 issue included four such offers. Several of the members have reported that they have sold the hardware in question. The following listings summarize the current availability of the used hardware. Since the material may be sold at any time it will be best to write first to check on terms and availability and to send money later.

TI-59
PC-100C with paper
Library Modules with Manuals:
- Statistics
- Math Utilities
- Real Estate/Investment
- Leisure Library
- Electrical Engineering

Over 100 Magnetic Cards
12V Cigarette Lighter Adapter
Specialty Packs:
- 59 Fun
- Printer Utility
- Electronics

Service Manual

In addition, several PPX programs will be included in a package deal for $150.00. Write to Robert J. Stucker, 7606 E. 90th Terrace, Kansas City, MO 64138.

TI-59 Modules - Leisure Library, Math/Utilities, and Real Estate/ Investment modules together with the manuals are available. Send ten dollars per module to PPC Publications, P.O. Box 1421, Largo, Florida 34649. First come, first served.

Memo Processor Card for the CC-40	$ 50.00
Hex-bus RS-232 Interface for the CC-40	125.00
Hex-bus Video Interface for the CC-40	125.00
Hex-bus Wafer Tape Drive for the CC-40	125.00

All hardware sold as is. He will sell the whole package for $300.00 . Write to Dan Eicher, P.O. Box 17401, Indianapolis, IN 46217. You can call at nights at 317-241-9942.

NEW TI-66'S AND PC-200'S - Former editor Maurice Swinnen sent in a catalog from Damark International, 6707 Shingle Creek Parkway, Minneapolis, MN 55430. Page 18 includes an offer to sell a TI-66 with a PC-200 for a combined price of $49.00 plus $4.50 for insurance, shipping and handling. The order number is B-908-106898. You can call toll-free 1-800-950-9090 and can use VISA, Mastercard or Discover credit cards.

TI-95 SALE - Page 2 of the Educalc Catalog #43 offers the TI-95 for $89.95 plus $1.00 shipping. With the purchase you can order a Math or Statistics cartridge for only $5.00. Write to 27953 Cabot Road, Laguna Niguel, CA 92677, or telephone 1-800-633-2252, extension 352. Since the offer is limited to stock on hand I suggest that you call. Volume 17 of the Elek-Tek catalog lists the TI-95 at $75.00 plus $4.00 shipping. Write to 6557 N. Lincoln Ave., Chicago IL 60645-3986 or call 1-800-621-1269.

A LINEAR/HYPERBOLIC PROGRAM INADEQUACY

In V8N2P20 I reviewed the second edition of *Curve Fitting for Programmable Calculators* by William Kolb. The book gives formulas, graphs, and sample problems for forty different curve fitting solutions. One of the appendixes included a set of programs by Maurice Swinnen for fitting nine different curves with the TI-59. I noted that the set of TI-59 programs was an extension of the Maurice's curve fitting programs from V7N1/2, 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 from seven of the functions was also included. At that time I did not test the programs thoroughly.

The book included Maurice's offer to provide magnetic cards for the programs. Early this year Maurice called to ask if I could take over that task. I agreed and he sent a set of magnetic cards. Before delivering the first set of cards to a user I decided run a sample problem as a way determining that the cards had been copied properly. I used the set of ten points from V7N1/2P15-17, and followed the instructions on pages F-4 and F-5 of the book. The results were as expected until I tried the last program, the linear/hyperbolic function. The coefficient of determination (R^2, or RR in the book) was nearly six, where by definition the coefficient of determination is between 0 and 1. After considerable experimentation I found that a correct linear/hyperbolic solution could be obtained if that program was run immediately after completion of data entry, or immediately after a linear solution had been completed. If the linear/hyperbolic program was run immediately after a solution for any of the other seven functions then an incorrect linear/ hyperbolic solution would result. The printout at the right illustrates the problem, where the first linear/ hyperbolic solution was obtained immediately after data entry, and the remaining solutions are shown in the

```
      Y = A   +BX  +C/X
 .9865292832       R²
-240.7269436        A
 72.56438251        B
 197.0244632        C


      Y = AX²  + BX  +C
 .9999996076       R²
 5.001704545        A
  2.95549243        B
 7.264166655        C


      Y = A   +BX  +C/X
 5.938133597       R²   ←
 11.51295924        A
-.0138535647        B
 188.6230062        C


            1.
      Y = A   +BX
 .9545285755       R²
-102.7733333        A
 57.97424242        B

      Y = A   +BX  +C/X
 .9865292832       R²
-240.7269436        A
 72.56438251        B
 197.0244632        C


            7.
      Y = AX↑B
 .9902703064       R²
  12.2614812        A
 1.590468948        B

      Y = A   +BX  +C/X
 -1291.46662       R²   ←
-73.91481165        A
 62.41518559        B
 -52.7519787        C
```

The Linear/Hyperbolic Program - (cont)

order completed. The last solution shows that the coefficient of determination for a linear/hyperbolic solution performed after a power solution (y = ax^b) was -1291.4662! Again, the value must be between zero and one. By contrast, the polynomial curve fit solution, which was also a separate program, was insensitive to what solution had been completed before.

Some additional study revealed the source of the problem with the linear/hyperbolic solution. The program as listed on page F-10 of the book requires that the sums of y, y^2, x, x^2, and xy be in TI-59 data registers R01, R02, R04, R05, and R06 as indicated by the data register list on page F-3 of the book. The five sums will be in those locations at the end of the data entry program since that program uses the Σ+ function of the calculator to generate those sums, and that function accumulates those sums in those locations. All of those values will typically not be in the corresponding locations after one of the solutions has been completed. This is because linearized equations are used to solve for the last six functions in the seven curve fit program. The values in the five statistics data registers will have been replaced by other appropriate sums so that the built-in linear regression solutions (Op12 through Op15) can be used to obtain solutions to the linearized functions. Of course, this is exactly the kind of advanced use of the statistics functions which was described at the bottom of page V-38 of *Personal Programming*. Some of the five sums calculated with the Σ+ function are used in the linearized functions; therefore, the five sums are stored elsewhere to be available for recall as needed. This explains why the linear/hyperbolic program performs properly when run immediately after the linear solution has been run.

Modifying the linear/hyperbolic program so that it will run correctly after any other solution has been run is reduced to simply replacing the RCL's from the five statistics registers listed above to RCL's from the alternate storage registers according to the following table:

Sum	Statistics Register	Alternate Storage
y	R01	R08
y^2	R02	R09
x	R04	R15
x^2	R05	R16
xy	R06	R23

I modified the linear/hyperbolic program to recall the sums from the proper locations and demonstrated that the program then yields the correct solution independent of the solution sequence. A listing of the modified program appears on the next page.

I also modified the table of data register useage from page F-3 of the Kolb book to reflect the alternate storage for the five sums described above, and to more properly describe data registers R01, R02, R04, R05 and R06 as "scratch pad" memories. If you would like a copy of the revised table to paste in your copy of the Kolb book send a SASE and an extra stamp.

The Linear/Hyperbolic Program - (cont)

Program Listing:

000	76	LBL	046	26	26	092	19	19	138	95	=	184	54	)	230	69	OP
001	69	OP	047	54	)	093	33	X²	139	42	STO	185	85	+	231	91	R/S
002	69	OP	048	75	-	094	95	=	140	58	58	186	43	RCL	232	76	LBL
003	00	00	049	43	RCL	095	42	STO	141	43	RCL	187	57	57	233	13	C
004	69	OP	050	08	08	096	56	56	142	08	08	188	65	×	234	43	RCL
005	04	04	051	65	×	097	43	RCL	143	75	-	189	43	RCL	235	57	57
006	32	X:T	052	43	RCL	098	52	52	144	43	RCL	190	26	26	236	32	X:T
007	69	OP	053	19	19	099	65	×	145	58	58	191	54	)	237	01	1
008	06	06	054	95	=	100	43	RCL	146	65	×	192	75	-	238	05	5
009	92	RTN	055	42	STO	101	53	53	147	43	RCL	193	43	RCL	239	71	SBR
010	76	LBL	056	53	53	102	75	-	148	15	15	194	14	14	240	69	OP
011	15	E	057	43	RCL	103	43	RCL	149	95	=	195	95	=	241	98	ADV
012	98	ADV	058	03	03	104	54	54	150	75	-	196	55	÷	242	91	R/S
013	69	OP	059	33	X²	105	65	×	151	43	RCL	197	53	(	243	76	LBL
014	00	00	060	75	-	106	43	RCL	152	57	57	198	43	RCL	244	14	D
015	43	RCL	061	43	RCL	107	55	55	153	65	×	199	09	09	245	42	STO
016	45	45	062	15	15	108	95	=	154	43	RCL	200	75	-	246	07	07
017	69	OP	063	65	×	109	55	÷	155	19	19	201	43	RCL	247	32	X:T
018	02	02	064	43	RCL	110	53	(	156	95	=	202	14	14	248	04	4
019	43	RCL	065	19	19	111	53	(	157	55	÷	203	95	=	249	04	4
020	38	38	066	95	=	112	43	RCL	158	43	RCL	204	32	X:T	250	71	SBR
021	69	OP	067	42	STO	113	52	52	159	03	03	205	03	3	251	69	OP
022	03	03	068	54	54	114	65	×	160	95	=	206	05	5	252	65	×
023	43	RCL	069	43	RCL	115	43	RCL	161	42	STO	207	07	7	253	43	RCL
024	50	50	070	23	23	116	56	56	162	59	59	208	08	8	254	58	58
025	69	OP	071	65	×	117	54	)	163	43	RCL	209	71	SBR	255	54	)
026	04	04	072	43	RCL	118	75	-	164	08	08	210	69	OP	256	85	+
027	69	OP	073	03	03	119	43	RCL	165	33	X²	211	91	R/S	257	43	RCL
028	05	05	074	54	)	120	54	54	166	55	÷	212	76	LBL	258	59	59
029	43	RCL	075	75	-	121	33	X²	167	43	RCL	213	11	A	259	54	)
030	16	16	076	43	RCL	122	95	=	168	03	03	214	43	RCL	260	85	+
031	65	×	077	15	15	123	42	STO	169	95	=	215	59	59	261	43	RCL
032	43	RCL	078	65	×	124	57	57	170	42	STO	216	32	X:T	262	57	57
033	03	03	079	43	RCL	125	43	RCL	171	14	14	217	01	1	263	55	÷
034	54	)	080	08	08	126	55	55	172	43	RCL	218	03	3	264	43	RCL
035	75	-	081	95	=	127	75	-	173	59	59	219	71	SBR	265	07	07
036	43	RCL	082	42	STO	128	53	(	174	65	×	220	69	OP	266	95	=
037	15	15	083	55	55	129	43	RCL	175	43	RCL	221	91	R/S	267	32	X:T
038	33	X²	084	43	RCL	130	54	54	176	08	08	222	76	LBL	268	04	4
039	95	=	085	03	03	131	65	×	177	54	)	223	12	B	269	05	5
040	42	STO	086	65	×	132	43	RCL	178	85	+	224	43	RCL	270	06	6
041	52	52	087	43	RCL	133	57	57	179	43	RCL	225	58	58	271	05	5
042	43	RCL	088	20	20	134	95	=	180	58	58	226	32	X:T	272	71	SBR
043	03	03	089	54	)	135	55	÷	181	65	×	227	01	1	273	69	OP
044	65	×	090	75	-	136	43	RCL	182	43	RCL	228	04	4	274	98	ADV
045	43	RCL	091	43	RCL	137	52	52	183	23	23	229	71	SBR	275	91	R/S

Page 30 of issue #42 of the EduCALC Catalog lists the 3rd edition of the Kolb book for $13.95. The stock number is M-135. I do not know if the problem with the linear/hyperbolic program has been corrected in that edition.

YOU CAN ONLY PLEASE SOME OF THEM SOME OF THE TIME - The first issue for this year of our newsletter devoted eleven pages to compiling and updating the index of PPX programs which are available from other members. I anticipated that there would be a mixed response to that effort:

"Yes, I suppose I wasn't very happy with 11 pages of PPX programs. Fortunately you also included 23 pages of normal material, thereby creating one of the biggest issues of the Notes." R. P.

"How convenient that more programs have been added to the exchange promoted by TI PPC Notes. ... " J.V.

THE USE OF SYSTEM FLAGS IN A PROGRAM WHILE IN THE PROTECTED MODE

Pages A-2 through A-7 of the *TI-95 Programming Guide* describe the features of the system-protected and the system-unprotected modes. Page A-2 states in part:

"Besides the functions available in the system-protected mode, the system unprotected mode lets you access the system flags (16-99) using the SF, RF, and TF instructions. ..."

It turns out that the system flags can be accessed from a program in the protected mode. For example, in previous issues:

* the Linear Equations program on V11N4P8 demonstrated the use of the sequence TF 74 to skip over a print command as a means to speed up program execution when a printer is not connected, and

* V11N1P19 demonstrated the use the system flags 33 and 34 in a program to control the degrees/radians/grad modes.

I have also been able to demonstrate the use of system flags 41 and 42 in a program to control the octal and hexadecimal modes. More importantly, I have been able to demonstrate that the SF 49 and RF 49 commands can be used in a program to change from the protected mode to the unprotected mode, and vice versa. This is in direct contradiction of the statement on page A-4 of the *TI-95 Programming Guide* that

"You cannot remove system protection from within a program".

The ability to control entry into and exit out of the unprotected mode from inside a program should relieve some of the concerns over operator errors causing calculator crashes while in the unprotected mode. The revised program for extended precision calculation of square roots on page 10 of this issue demonstrates this capability.

I expect that the remaining system flags which are listed on pages C-7 through C-9 of the *TI-95 Programming Guide* can also be controlled from a program with the calculator operating in the protected mode.

Finally, my tests show that the system flags cannot be accessed from the keyboard when in the protected mode, but can be accessed from the keyboard in the unprotected mode. If you try to access a system flag from the keyboard in the protected mode you will receive the error message "INVALID SEQUENCE".

<u>A QUOTE FROM THE PAST</u> - William Vogel stopped using his TI-59 and sent some of his residual material to me. One of the items which I had not seen previously was Richard Vanderburgh's January 1980 notice of closeout options for the subscribers to *52 Notes*. In discussing the *TI PPC Notes* option Richard stated:

"... Although there may not be much more strictly new material to cover (until such time as TI markets a new PPC) ... "

That was about six months before the discovery of fast mode and about a year before the discovery of high resolution graphics.

<u>TI-95 T REGISTER TESTS USING SYSTEM REGISTER 2079</u> - The discussion of translation of programs from the TI-59 to the TI-95 in V12N4P16-18 stated that "Comparison tests are available in the TI-95 between the display register and any data register, but NOT with the t register." Robert Prins and Darrin Chambers have commented that the statement is incorrect since t register comparisons can be made by accessing system register 2079 with the TI-95 operating in the unprotected mode. The equivalent instructions are then:

TI-59	TI-95
x=t	x=t
CP (clear the t register)	x=t 0 x=t
x=t N	IF= 2079 GTL NN
x≥t N	INV IF< 2079 GTL NN
INV x=t N	INV IF= 2079 GTL NN
INV x≥t N	IF< 2079 GTL NN

where N is any valid TI-59 label, and NN is any valid two-character TI-95 label. The labels can be replaced by appropriate absolute addressing.

To enter a test of system register 2079 you must have first placed the calculator in the unprotected mode. Press FUNC, press F3 (SYS), and press F1 (YES) to set the unprotected mode. You can then use the system registers while entering a program in LEARN mode. The only difference you will see will be that an entry of any data register, user or system, requires the entry of four digits. However, if you enter fewer than four digits, say for a user data register, the LEARN mode will automatically supply the necessary leading zeroes when you press the next entry key.

In the past I had reservations about routinely operating in the unprotected mode for the convenience of using the t-register rather than substituting another unused register. Perhaps I was unduly influenced by the caution on page A-3 of the *TI-95 Programming Guide*:

> "Unless you have a specific reason for accessing a system register, always leave the calculator in the protected mode. This prevents you from inadvertently changing any system registers that may affect the operation of the calculator.
>
> Each system register has a fixed use that is determined by the design of the calculator. If you accidentally or indiscriminately change the contents of a system register, you could alter option settings or destroy important data.
>
> In extreme cases, you could temporarily disable the keyboard or the display. In such a case, you must press the RESET button to restore normal operation."

TI-95 T Register Tests Using System Register 2079 - (cont)

My "foot-dragging" on the use of the unprotected mode caused some frustration to some of our members. For example, recently Robert Prins wrote:

> "Why is one of the gurus of 'synthetic programming' on the TI-59 so very afraid of using the TI-95 in unprotectd mode? If you don't use any unlisted SBA routines it is certainly not much more dangerous than using Fast or Graphics Mode on the TI-59."

Of course, not using any unlisted SBA routines is not necessarily the same as not *intending* to use any unlisted SBA routines. I have typically kept the file space of my TI-95 full of programs, and an inadvertent crash could cause real problems with recovery. By contrast the potential loss due to a crash with the TI-59 was much more limited. However, my attitude was changed by two recent events:

1. The inadvertent designation of a data register (R10) which was used elsewhere in the translation of the program for extended precision square root calculations (V12N2P24). Of course, as illustrated on page 2 of this issue, that kind of problem can be circumvented by selecting a data register for comparison tests which isn't available to the TI-59 program. In the case of the correction for the extended precision square root program I used data register 099 since that was clearly beyond the range of data registers used in the TI-59 program which ran with partitioning 9 Op 17. Even safer would be the use of a data register which is entirely beyond the capability of the TI-59, say any TI-95 data register of 100 or higher.

2. As noted on page 7 of this issue I found that it is possible to set the unprotected mode from inside a program and to return to the protected mode at the end of program execution. This method of operation should substantially reduce the chance of inadvertent change of a system register with a possible system crash.

A modification of the extended precision square root program from V12N2P24 was used to illustrate

(1) the use of t registers tests using system register 2079, and

(2) the use of SF 49 and RF 49 to control the use of the unprotected mode from the program.

The upper listing on page 10 was made with the TI-95 in the unprotected mode. Note that all data register numbers are shown to four digits. The lower listing on the next page was made with the TI-95 in the protected mode. Note that the only data register numbers which are shown to four digits are those which define system registers. In the upper listing on page 10 the t register tests using system register 2079 appear at lines 0119, 0125, and 0165. Entry to the unprotected mode with SF 49 appears at line 0049. Two exits from unprotected mode with RF 49 appear at lines 0223 and 0382.

TI-95 T Register Tests Using System Register 2079 - (cont)

Square Root of N Program Listing in Unprotected Mode

```
0000 CE `SQUARE ROOT N`
0014 ADV PRT CMS CE `1<`
0020 `N<100 ?` BRK ADV
0029 PRT STO 0049 CE `#`
0035 ` BLOCKS? <41:` BRK
0049 ADV SF 49 STO 0002
0055 10 STO 0000 INV LOG
0062 STO 0001 1 STO 0008
0069 STO 0089
0072 LBL 01 90 STO 0003
0080 STO 0005 50
0085 STO 0004 STO 0006
0091 RCL 0008 STO 0007
0097 +/- ST+ 0003
0101 ST+ 0004
0104 LBL 02 RCL IND 0003
0111 INC 0003 x~t
0115 RCL IND 0004
0119 INC 0004 IF= 2079
0125 GTL 02 IF< 2079
0131 GTL 05 0 x~t 0
0137 LBL 03 INV INC 0005
0144 INV INC 0006
0148 ST- IND 0006
0152 RCL IND 0005
0156 ST- IND 0006
0160 RCL IND 0006 INV
0165 IF< 2079 GTO 0179
0171 RCL 0001
0174 ST+ IND 0006 1.
0180 DSZ 0007 GTL 03
0186 INC 0009 2
0190 LBL 04 ST+ 0089
0196 GTL 01
0199 LBL 05 DSZ 0000
0205 GTL 06 10 EXC 0000
0213 EXC 0009 PRT
0217 INC 0008 DSZ 0002
0223 GTL 06 RF 49 CE `M`
0230 `ORE BLOCKS?` Y/N
0242 GTL 11 GTO 0000
0248 LBL 06 RCL 0007
0254 STO 0003 10
0259 ST* 0009
0262 LBL 07 INV INC 0005
0269 ST* IND 0005
0273 DSZ 0003 GTL 07 x^2
0280 LBL 08 INV INC 0006
0287 ST* IND 0006
0291 DSZ 0007 GTL 08 2
0298 STO 0007 50
0303 LBL 09 STO 0006
0309 RCL 0008 EXC 0003
0315 x~t 0 x~t
0318 LBL 10 ST- IND 0006
0325 INV INC 0006 x~t
0330 ST+ IND 0006 (
0335 RCL IND 0006 /
0340 RCL 0001 )( INT *
0347 x~t RCL 0001 )
0352 DSZ 0003 GTL 10 90
0360 DSZ 0007 GTL 09 9
0367 +/- GTL 04
0371 LBL 11 CE `# BLOCK`
0382 `S?` BRK SF 49
0387 STO 0002 GTL 06
```

Square Root of N Program Listing in Protected Mode

```
0000 CE `SQUARE ROOT N`
0014 ADV PRT CMS CE `1<`
0020 `N<100 ?` BRK ADV
0029 PRT STO 049 CE `# `
0036 `BLOCKS? <41:` BRK
0049 ADV SF 49 STO 002 1
0056 0 STO 000 INV LOG
0062 STO 001 1 STO 008
0069 STO 089
0072 LBL 01 90 STO 003
0080 STO 005 50 STO 004
0088 STO 006 RCL 008
0094 STO 007 +/- ST+ 003
0101 ST+ 004
0104 LBL 02 RCL IND 003
0111 INC 003 x~t
0115 RCL IND 004 INC 004
0122 IF= 2079 GTL 02
0128 IF< 2079 GTL 05 0
0135 x~t 0
0137 LBL 03 INV INC 005
0144 INV INC 006
0148 ST- IND 006
0152 RCL IND 005
0156 ST- IND 006
0160 RCL IND 006 INV
0165 IF< 2079 GTO 0179
0171 RCL 001 ST+ IND 006
0178 1. DSZ 007 GTL 03
0186 INC 009 2
0190 LBL 04 ST+ 089
0196 GTL 01
0199 LBL 05 DSZ 000
0205 GTL 06 10 EXC 000
0213 EXC 009 PRT INC 008
0220 DSZ 002 GTL 06
0226 RF 49 CE `MORE BLO`
0237 `CKS?` Y/N GTL 11
0245 GTO 0000
0248 LBL 06 RCL 007
0254 STO 003 10 ST* 009
0262 LBL 07 INV INC 005
0269 ST* IND 005 DSZ 003
0276 GTL 07 x^2
0280 LBL 08 INV INC 006
0287 ST* IND 006 DSZ 007
0294 GTL 08 2 STO 007 50
0303 LBL 09 STO 006
0309 RCL 008 EXC 003 x~t
0316 0 x~t
0318 LBL 10 ST- IND 006
0325 INV INC 006 x~t
0330 ST+ IND 006 (
0335 RCL IND 006 /
0340 RCL 001 )( INT *
0347 x~t RCL 001 )
0352 DSZ 003 GTL 10 90
0360 DSZ 007 GTL 09 9
0367 +/- GTL 04
0371 LBL 11 CE `# BLOCK`
0382 `S?` BRK SF 49
0387 STO 002 GTL 06
```

ENTRY OF NULL STRINGS - V12N3P4-6 reviewed three books co-authored by Maurice Swinnen which provide a wide range of BASIC programs which are easily converted for the TI-74. The programs which require multiple input accept the input as string values and convert the string values to numerics internally using VAL commands. This permits termination of the input by entering the letter "E". In V13N1P28 the editor reported that the input of a null string to signify the end of input data was more convenient than the input of "E" as in the Swinnen programs. Maurice writes that he considered that method for his books, but the version of BASIC used in the Sharp machines will not accept a null string input.

<u>COPROCESSING TRIG FUNCTIONS</u> - George Wm. Thomson. I have been evaluating various methods for the precise calculation of trigonometric functions on digital computers and the design of suitable benchmarks. The core of most modern methods appears to be the computation of the tangent after scaling down the angle to the first octant (0 to 45 degrees) and the computation of the arctangent. All direct and inverse functions can be easily derived from tan and arctan by simple relations. Both of these functions can be calculated by a "digit-by-digit" method which goes back to the English mathematician, Henry Briggs, who published the first table of logarithms in 1624. It has also been called the CORDIC method. Professor W. Kahan of the University of California at Berkeley, in his paper "Mathematics Written in Sand", *Proceedings of the Statistical Computing Section*, American Statistical Association, 1983, pp 12-26, stated that the HP pocket calculators and the Intel 8087 chip use these methods to generate tan and arctan to 64 significant bits (19.3 decimal SF) and that he and Steve Baumel had written the programs, apparently for both HP and Intel.

I had hoped to include among my many wide range comparisons, the exact algorithms used in the Intel 8087 chip and its successors for both the angle scale down and the generation of the tan and arctan functions in view of their vaunted high accuracy. I have a vague memory that I saw a technical or semi-techical article describing these exact algorithms in detail, not generalities. However, correspondence with a list of authors who should be in the Who's-who of coprocessing drew a complete blank! Inquiries to Intel were not fruitful. A telephone call to Professor Kahan was also unfruitful, since he could not cite either open literature or patents describing the algorithms. The subscribers to *TI PPC Notes* have long memories and are great for sharing their information and know-how. Perhaps someone will let me know if they have knowledge of these algorithms. If so, write to George Wm. Thomson, 5066 Elmhurst Avenue, Royal Oak, MI 48073-1102, telephone 313-435-2070).

THE SMALLEST CIRCLE PROBLEM REVISITED - In V12N4P26 Don Laughery proposed the finding of the smallest circle which would just enclose n random points as a programming challenge. V13N1P28-29 presented a solution which was the result of the combined effort of Larry Leeds, Peter Messer and the editor. The publication of the program elicited interest from other members. The details of the subsequent work on the solution are presented below to show the kind of results that can be achieved when we get several members working on a problem. One result is that I have a file of notes that is nearly an inch thick.

Carl Rabe, George Thomson and Larry Leeds all found that a divide by zero error will be generated when three of the points are on a straight line, or when the same point is entered twice, which is the same thing as saying that three points are in a straight line. The problem is caused by the divides by G at line 445, where the value of G calculated at line 410 will be equal to zero for the conditions mentioned above. The problem was circumvented by testing for G = 0 and replacing the zero by a very small number; that is, by inserting a new line

```
412 IF G=0 THEN G=1.E-06
```

Carl Rabe then found that the smallest circle will not be found for the sequence of points

(5,7) (9,10) (12,13) (16,18) (21,25) and (25,21)

but would be found if the sequence was rearranged. The set of points involves the case where the two most distant points are the first and last points, and the remaining points are within the circle with its diameter equal to the distance between the two points, and its center midway between the two points. There was a fundamental flaw in the program at lines 205 and 210 which were

```
205 FOR I=1 TO N
210 FOR J=2 TO N-1
```

If you think about those two statements you will realize that the distance will never be calculated between the first and last points in the input sequence. There will be other cases which will not be tested as well. The deficiency can be eliminated for programs on the TI-74 or CC-40 by changing line 210 to

```
210 FOR J=(I+1) TO N
```

but that will not work for computers which use Microsoft BASIC such as the Radio Shack Color Computer or Model 100. A routine which seems to work with all the BASIC implementations is

```
205 FOR I=2 TO N
210 FOR J=1 TO I-1
```

The program was modified in the second way to avoid potential problems if the program were to be used on other machines. With the changes described above the program appeared to operate satisfactorily. Carl Rabe found that if he entered the set of points

(8,11) (21,2) (22,18) (34,26) (33,38) and (29,31)

Smallest Circle - (cont)

the program would yield the correct answer with the center at (24.4375, 20.845166...) and a radius of 19.1649... . But if the point (22,18) was duplicated at the end of the set an incorrect answer with the center at (22,18) and a radius of 16.0312... would result. At first we were really puzzled because the routine from lines 705 to 725 was expressly put into the program to ensure that we couldn't have this sort of problem. Finally, we realized that line 710 prevents the testing of the three points which were used to define the circle being tested by the routine. We deleted line 710 and received a correct solution with the duplicate points. (Incidentally, line 710 is not very important--it was added to provide some small saving in execution time by not testing the points used to define the circle). After thinking about the problem some more we found that there was another solution to the difficulty with duplicate points. It is to change the branch at line 712 when G = 0 from

```
IF G=0 THEN G=0.000001     to     IF G=0 THEN 735
```

where we rely on the idea that if G = 0 then there are either all three points in a straight line, or duplicate points and we really don't want to examine the resulting "circle" which has an infinite radius anyway. Again, we received correct results with duplicate points.

The program now appears to work satisfactorily, but not necessarily efficiently. The inefficiency results from having to examine the circles defined by every combination of three points, a result of our inability to find a methodology for finding the correct combination more directly. The inefficiency was emphasized when Don Laughery wrote that the solution could sometimes be required to examine twenty or more points. The following table compares solution times for various numbers of points for the program on the TI-74, for the equivalent program on the Radio Shack Model 100 and for a program by Carl Rabe in True BASIC on the ATARI 1040ST processor.

		Execution Time (seconds)		
n	nC3	TI-74	M 100	ATARI
3	1	1	1	< 1
4	4	3	3	
5	10	8	6	
6	20	17	13	
9	84	92	71	
12	220	301	239	
15	455	752	608	
18	816	1558	1305	
21	1330	2940	2444	78

Smallest Circle - (cont)

Those execution times are for the condition where all the points are not within the circle defined by the two most distant points. If all the points are within the circle defined by the most distant points the execution times would be much shorter. nC3 in the table is the number of combinations of n points taken three at a time. Not surprisingly, the execution time appears to approximately follow nC3 since the major portion of the calculations are involved with testing each combination of three points to determine whether it defines the smallest circle. Actually, we would expect that the execution time would be proportional to the product of n times nC3 since for each combination of three points all of the remaining points are tested to determine whether they are inside the circle defined by the three points.

Now, purists will object that we haven't demonstrated that we *always* find the smallest circle. Larry Leeds has pointed out that, if one of the circles defined by one of the sets of three points encloses all of the remaining points, then the circle is of minimum radius. This follows from the fact that the center of the circle is equidistant from each of the three points, and if the center is displaced in any direction the distance to at least one of the three points will be increased. But, we have never proved that there cannot be sets of points for which none of the circles defined by any three points of the set will enclose all of the remaining points. Lines 705 through 725 of the present program account for that possibility by expanding the circle radius to enclose any point outside the circle defined by a set of three points, but it seems obvious that this will not yield the smallest circle. For example, for a single point outside the circle defined by three points, a smaller enclosing circle could probably be obtained by moving the center toward the outside point, and increasing the radius by a smaller amount. Larry has also found the following example which controverts our early assumptions that the smallest circle must go through at least one of the two points which are furthest apart:

(0,6) (3.5,5) (0,-6) (-7,0) (3.5,-5)

where the samllest circle passes through the second, fourth and fifth points,and encloses the first and third points which are the furthest apart. Meanwhile, on February 23 Don Laughery wrote:

> "I had occasion to used the smallest circle program under fire this week. ... ... I collected the data and enterd it into the program. While it was running (3 1/2 minutes) I plotted the data and checked it with the overlay - BINGO!! I even caught a few reversed-sign errors I made in plotting. Out of 25 runs no anomalies showed up, so it looks like maybe it's the answer. My thanks to you and all the others who worked on this. ..."

A printout of the current version of the program for the TI-74 appears on page 15. Members are invited to examine the remaining issues of improved efficiency and proof of minimum radius for all cases. Any input will be reported in subsequent issues.

Smallest Circle - (cont)

TI-74 Program Listing

```
10 A$="Smallest Circle":
PRINT A$:PAUSE 1
20 DIM X(30),Y(30)
25 INPUT "Use Printer? Y
/N ";Z$
30 IF Z$="Y"OR Z$="y"THE
N PN=1 ELSE 100
35 PRINT "Device Numbers
:":PAUSE 1
40 PRINT "For the HX-100
0 enter 10":PAUSE 1
45 PRINT "For the PC-324
 enter 12":PAUSE 1
50 INPUT "Enter device n
umber ";D$
55 OPEN #1,D$,OUTPUT
60 IF D$="10"THEN PRINT
#1,CHR$(18)
65 PRINT #1:PRINT #1,A$
70 PRINT #1
100 PRINT "End Input by
Entering "&CHR$(255):PAU
SE 2
110 N=1
120 X$="X = "
125 Y$="Y = "
130 INPUT X$;XX$:IF XX$=
""THEN 190
135 INPUT Y$;YY$:IF YY$=
""THEN 190
140 X(N)=VAL(XX$)
145 Y(N)=VAL(YY$)
150 IF PN=0 THEN 180
155 PRINT #PN,X$;X(N)
160 PRINT #PN,Y$;Y(N)
170 PRINT #PN
180 N=N+1:GOTO 130
190 N=N-1
195 PRINT "Solving"
200 M1=0
205 FOR I=2 TO N
210 FOR J=1 TO I-1
215 D2=(X(I)-X(J))^2+(Y(
I)-Y(J))^2
220 IF D2>M1 THEN M1=D2:
S=I:T=J
225 NEXT J
230 NEXT I
235 HH=(X(S)+X(T))/2
240 KK=(Y(S)+Y(T))/2
245 M2=M1/4
300 Z=0
305 FOR I=1 TO N
310 IF I=S OR I=T THEN 3
25
315 D2=(X(I)-HH)^2+(Y(I)
-KK)^2
320 IF D2>M2 THEN M2=D2:
Z=1
325 NEXT I
335 IF Z=0 THEN 800
345 FOR I=N TO 3 STEP -1
350 FOR J=(I-1)TO 2 STEP
 -1
355 FOR L=(J-1)TO 1 STEP
 -1
365 A=X(I):B=Y(I)
370 C=X(J):D=Y(J)
375 E=X(L):F=Y(L)
400 A=A-E:C=C-E
405 B=B-F:D=D-F
410 G=C*B-A*D:G=G+G
412 IF G=0 THEN 735
415 H1=C*C+D*D
420 X1=B:B=H1
425 B=X1*B:H1=A*H1
430 X1=X1*X1+A*A
435 D=X1*D:C=X1*C
440 B=B-D:C=C-H1
445 B=B/G:C=C/G
450 E=E+B:F=F+C
455 R2=B*B+C*C
460 H=E:K=F
705 FOR M=1 TO N
710 IF M=L OR M=J OR M=I
 THEN 725
715 D2=(X(M)-H)^2+(Y(M)-
K)^2
720 IF D2>R2 THEN R2=D2
725 NEXT M
730 IF R2<M2 THEN M2=R2:
HH=H:KK=K
735 NEXT L
755 NEXT J
760 NEXT I
800 PAUSE ALL
810 PRINT #PN," h = ";HH
820 PRINT #PN," k = ";KK
830 PRINT #PN," r = ";SQ
R(M2)
840 IF PN=1 THEN PRINT #
1
850 PAUSE 0
999 END
```

LABELON PAPER AVAILABILITY - V13N1P3 reported that PC-100 paper was available from L. E. Muran Co. William Gorman writes: "I can specifically say that you should NOT call L. E. Muran Co. because they claim they only deal with large companies ... . I did obtain the toll-free number for Labelon, which is 800-428-5566, and through that obtained the local distributor so I could find out who were their stocking stationery stores for this product. (In the St. Petersburg area Calco Office Supply at 5028 66th St. N., St. Petersburg, FL, telephone 813-544-6285 is the distributor).

The Labelon thermal calculator paper type CR-025 does carry the following statement 'Approved for use in Texas Instruments PC-100 and SR-60 and Kodak Q-700 System.' Labelon indicated that they used to manufacture the material for TI distribution and hence this is the same paper, although it seems to give a blacker print that the TI paper that I have personally used in the past. There are three rolls of the CR-025 paper in each pack. It is priced at approximately $10.00-$12.00 per pack"

FIVE FUNCTION CURVE FIT FOR THE TI-95 - A five function curve fit program for the TI-74 appeared in V12N4P24-25. Scott Garver translated the program from BASIC for use on the TI-95. A printout for a sample problem appears at the right. The problem is the same one used to demonstrate the BASIC program on V12N4P24. The program listing appears on page 17. A complete set of prompts are provided with the program so no user instructions are given here. Features of interest to the user include:

1. The partitioning must accomodate 1430 program steps.

2. The program can accept 50 data points.

3. TF74 is used to determine if a printer is connected. With a printer the results are automatically printed. Without a printer the calculator stops at each output.

4. The "Display Inputs?" option at program entry allows non-printer use to either echo the input or to increase input speed by omitting the echo of the input.

5. Steps 369-371 demonstrates a use of an assembly language subroutine from page C-14 of the *TI-95 Programming Guide*.

6. If the determinant is zero the program stops and displays the messages "DETERMINANT = 0" and "RESTART PROGRAM".

```
5 CURVE FITTER

X( 1)=         1.
Y( 1)=       3.2
X( 2)=         2.
Y( 2)=       7.4
X( 3)=         3.
Y( 3)=       12.
X( 4)=         4.
Y( 4)=      16.8
X( 5)=         5.
Y( 5)=       22.

OPTIONS:
1) LINEAR Y=a+bX
2) EXP. Y=ae^bX
3) POWER Y=aX^b
4) LOG Y=a+bLnX
5) HYPER Y=a+b/X
6) FIND BEST FIT

FOR Y = a + bX
a =        -1.82
b =          4.7
r =  .9992132427
mn =          0.
rms= .2638181192

FOR Y = ae^bX
a =   2.48354533
b =  .4675682173
r =  0.972298671
mn =-.1669169783
rms= 1.985107914

FOR Y = aX^b
a =  3.211745293
b =  1.196427406
r =  .9999840273
mn =-0.005131201
rms= .0427736287

FOR Y = a + bLnX
a =  1.481255066
b =  11.27808205
r =  .9637126812
mn =       4.-13
rms= 1.775706102

FOR Y = a + b/X
a =  21.54522175
b = -20.28880676
r = -.8849129677
mn =    -1.04-11
rms= 3.098229048

BEST CURVE IS 3.

*** RESIDUALS ***
D 1=-0.011745293
D 2= .0395963302
D 3= .0441063981
D 4=-0.067944759
D 5=-.0296686814

PREDICT y FOR x ...
a =  3.211745293
b =  1.196427406
x =           3.
y =   11.9558936
```

Editor's Notes:

The SBA 226 command can be adequately replaced with a PAU if you do not worry about the slight delay, or can be replaced with an ADV PRT if you simultaneously delete the ADV OLD TF74 PRT sequence at steps 0599-0603 of the present listing.

I worried about the possibility of the determinant being zero after the bad experiences with divide-by-zero effects in the smallest circle program. I wondered if there might be a reason that the determinant could not be zero, but I coudn't work it out. I referred the problem to our statistics expert, George Wm. Thomson. In just a few days he provided an analysis which showed that the determinant will be zero only if all the x values are the same. Consider the sum of the squares of the deviations from the mean $\bar{x}$:

$$SS = \sum(x_i - \bar{x})^2 \quad \text{where} \quad x = (\sum x_i)/n \text{ . Expand to get:}$$

$$SS = \sum x_i^2 - 2x\sum x_i + n\bar{x}^2 \qquad \text{Substitute } (\sum x_i)/n \text{ for } x \text{ :}$$

$$SS = \sum x_i^2 - 2(\sum x_i)^2/n + n(\sum x_i)^2/n^2$$

Combining the secondand third terms, and rearranging yields:

$$SS = \frac{n\sum x_i^2 - (\sum x_i)^2}{n}$$

where the numerator is the equation for the determinant. Since SS must be positive or zero and n is positive then SS can be zero only if all the x values are equal to the mean, which is the same as saying that all the x values are the same, which is the same as saying that all data points lie on the vertical line with x as the x intercept.

Five Function Curve Fit for the TI-95 - (cont)

Program Listing

```
0000 CMS DEC CFG `DISPL`
0008 `AY INPUTS ?` Y/N
0020 SF 14 DFN CLR CE
0025 `5 CURVE FITTER`
0039 PRT ADV 1 STO A 50
0046 STO B 100 STO C CLR
0054 `# DATA PAIRS ?`
0068 DFN F4:#▸r@C  RTN
0076 LBL C  STO D
0081 LBL AA RCL A `ENTE`
0090 `R X(  ` MRG A
0098 COL 11 `)`
0101 DFN F4:Xn @CA RTN
0109 LBL CA STO IND B
0115 `X` SBL Cn `ENTER `
0125 `Y(  ` MRG A COL 11
0133 `)` DFN F4:Yn @CB
0141 RTN
0142 LBL CB STO IND C
0148 `Y` SBL Cn INC A
0154 INC B INC C RCL D
0160 INV IF< A GTL AA
0166 DFN CLR `OPTIONS:`
0176 ADV PRT CE `1) LIN`
0185 `EAR Y=a+bX` PRT CE
0197 `2) EXP. Y=ae^bX`
0212 PRT CE `3) POWER Y`
0224 `=aX^b` PRT CE `4)`
0233 ` LOG Y=a+bLnX` PRT
0247 CE `5) HYPER Y=a+b`
0262 `/X` PRT CE `6) FI`
0271 `ND BEST FIT` PRT
0283 LBL Cb CE `SELECT `
0294 `OPT'N 1-6` DFN CLR
0305 DFN F4:OPT@CC RTN
0313 LBL CC STO E 6
0319 IF= E SBL CD
0324 LBL CT DFN CLR `FO`
0331 `R Y = a` 1 IF= E
0341 SBL Ch 2 IF= E
0347 SBL Ci 3 IF= E
0353 SBL Cj 4 IF= E
0359 SBL Ck 5 IF= E
0365 SBL Cl OLD SBA 226
0372 0 STO I STO J STO K
0379 STO L STO M 1 STO N
0386 50 STO B 100 STO C
0395 LBL CE RCL IND B
0401 STO G RCL IND C
0406 STO H 2 IF= E
0411 SBL CF 3 IF= E
0417 SBL CF 3 IF= E
0423 SBL CG 4 IF= E
0429 SBL CG 5 IF= E
0435 SBL CH RCL G ST+ I
0442 x^2 ST+ J RCL H
0447 ST+ K x^2 ST+ L (
0453 SQR * RCL G ) ST+ M
0460 INC N INC B INC C
0466 RCL N INV IF> D
0471 GTL CE ( RCL D *
0478 RCL J - RCL I x^2 )
0485 STO D 0 IF= D
0490 GTL CI SBL Co ((
0498 RCL K * RCL J -
0504 RCL M * RCL I )/
0511 RCL D ) STO IND P (
0518 ( RCL D * RCL M -
0525 RCL I * RCL K )/
0532 RCL D ) STO IND Q 2
0539 IF= E SBL CJ 3
0545 IF= E SBL CJ ((
0552 RCL M - RCL I *
0558 RCL K / RCL D )/((
0567 RCL J - RCL I x^2 /
0574 RCL D )*( RCL L -
0582 RCL K x^2 / RCL D )
0589 ) SQR ) STO IND R 1
0596 STO Y SBL CK ADV
0602 OLD TF 74 PRT
0606 RCL IND P `a `
0611 SBL Cm RCL IND Q
0617 `b ` SBL Cm
0622 RCL IND R `r `
0627 SBL Cm RCL IND T
0633 `mn ` SBL Cm
0639 RCL IND U `rms`
0645 SBL Cm 4 IF< F
0651 IF< E GTL CS INC E
0658 GTL CT
0661 LBL CS 6 INV IF= F
0668 GTL CU RCL 035 ABS
0675 STO V 1 STO W 2
0681 STO A 36 STO R
0687 LBL CV RCL IND R
0693 ABS IF> V SBL CW
0699 INC A INC R 5 INV
0705 IF< A GTL CV RCL W
0712 STO E `BEST CURVE `
0725 `IS  ` MRG E ADV
0732 PRT
0733 LBL CU CE `SEE RES`
0744 `IDUALS ?` Y/N
0753 GTL Cc GTL CX
0759 LBL Cc COL 01 `***`
0767 COL 15 `***` ADV
0773 TF 74 PRT 2 STO Y
0779 SBL CK
0782 LBL CX CE `PREDICT`
0793 ` y FOR x?` Y/N
0803 GTL Cd GTL Cf
0809 LBL Cd COL 16 ` ..`
0817 `.` ADV TF 74 PRT
0822 SBL Co `a =` COL 16
0830 MRG IND P PRT CE
0835 `b =` COL 16
0840 MRG IND Q PRT
0844 LBL Ca CE `ENTER x`
0855 `:` DFN CLR
0858 DFN F1:ESC@Cf
0865 DFN F4: X @CY RTN
0873 LBL CY STO G `x =`
0881 COL 16 MRG = TF 74
0887 PRT 0 STO H 3 STO Y
0894 SBL CZ RCL S +/-
0900 `y =` COL 16 MRG =
0907 PRT RTN
0909 LBL Cf CE `ANOTHER`
0920 ` OPTION ?` Y/N
0930 GTL Ce GTL Cg
0936 LBL Ce 0 STO F
0942 GTL Cb
0945 LBL CD 1 STO E 6
0952 STO F RTN
0955 LBL CF RCL H LN
0961 STO H RTN
0964 LBL CG RCL G LN
0970 STO G RTN
0973 LBL CH RCL G 1/x
0979 STO G RTN
0982 LBL CI `DETERMINAN`
0995 `T = 0` PRT CLR `R`
1003 `ESTART  PROGRAM`
1018 PAU ` ***` COL 01
1025 INS `*** ` TF 74
1032 PRT ADV ADV
1035 GTO 0000
1038 LBL CJ RCL IND P
1044 INV LN STO IND P
1049 RTN
1050 LBL CK 0 STO I
1056 STO J 1 STO N 50
1063 STO B 100 STO C
1070 LBL CL RCL IND B
1076 STO G RCL IND C
1081 STO H
1083 LBL CZ 1 IF= E
1089 GTL CM 2 IF= E
1095 GTL CN 3 IF= E
1101 GTL CO 4 IF= E
1107 GTL CP ( RCL H -
1114 RCL 044 - RCL 049 /
1122 RCL G )
1125 LBL CQ STO S 3
1131 IF= Y RTN 1 IF= Y
1137 GTL CR RCL S `D  `
1145 MRG N COL 04 SBL Cm
1152 LBL CR RCL S ST+ I
1159 x^2 ST+ J INC N
1164 INC B INC C RCL N
1170 INV IF> D GTL CL (
1177 RCL I / RCL D )
1183 STO IND T ( RCL J /
1190 RCL D ) SQR
1194 STO IND U RTN
1198 LBL CP ( RCL H -
1205 RCL 043 - RCL 048 *
1213 RCL G LN ) GTL CQ
1220 LBL CO ( RCL H -
1227 RCL 042 * RCL G y^x
1234 RCL 047 ) GTL CQ
1241 LBL CN ( RCL H -
1248 RCL 041 *( RCL 046
1256 * RCL G ) INV LN )
1263 GTL CQ
1266 LBL CM ( RCL H -
1273 RCL 040 - RCL 045 *
1281 RCL G ) GTL CQ
1287 LBL CW STO V RCL A
1294 STO W RTN
1297 LBL Cg DFN CLR CLR
1303 HLT
1304 LBL Ch OLD ` + bX`
1313 RTN
1314 LBL Ci OLD `e^bX`
1322 RTN
1323 LBL Cj OLD `X^b`
1330 RTN
1331 LBL Ck OLD ` + bLn`
1341 `X` RTN
1343 LBL Cl OLD ` + b/X`
1353 RTN
1354 LBL Cm `=` COL 16
1360 MRG = TF 74 PRT INV
1366 TF 74 BRK RTN
1370 LBL Cn `(  ` MRG A
1378 COL 05 `)=` COL 16
1384 MRG = TF 74 PRT
1389 TF 14 PAU 0 RTN
1394 LBL Co RCL E (+24)
1404 STO T (+5) STO U (+
1414 5) STO R (+5) STO P
1424 (+5) STO Q RTN
```

TI-59 PROGRAMS FOR CHEMICAL ENGINEERING - Marcel Bogart has compiled a bibliography of TI-59 programs which were published in references such as *Chemical Engineering* and *Oil & Gas Journal*. The ten page list includes 66 references covering the period from March 1979 through August 1982. Each entry includes a one or two sentence abstract. He does not have copies of the articles or of the programs so you will have to find them in an engineering library. If you would like a copy of the list send one dollar to cover copying and postage.

MORE ON TI-95 INTERNAL PROGRAMS - Robert Prins has compiled a printout of the keystroke programmed functions for the TI-95. The seventy page document includes listings of the code with comments. He will provide a copy for ten dollars which includes shipping. Use an International Postal Money Order or equivalent - no personal checks, please. Write to Robert A. H. Prins, Alfred Nobellaan 112, 3731 DX DE BILT, NETHERLANDS.

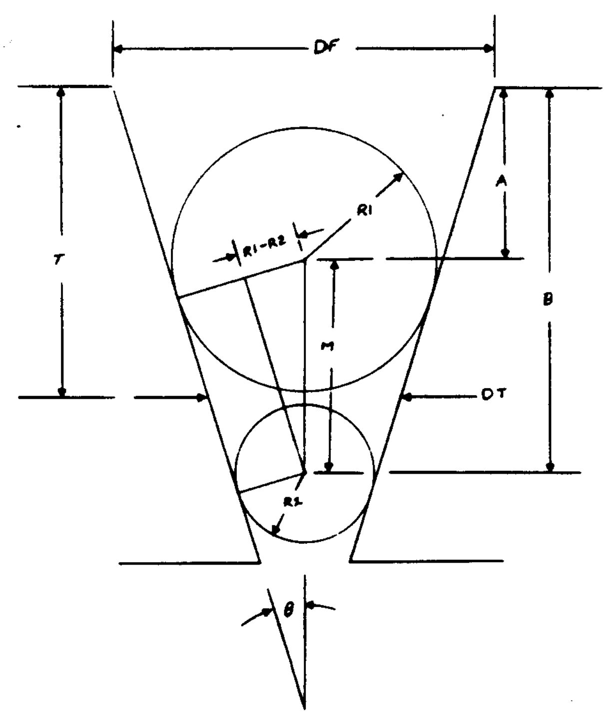
PALINDROMIC CHARACTERISTICS - Larry Leeds writes: In working with random number generators I decided to investigate lengthy repeating decimals. My interest was in the frequency count for the integers zero through nine. Much to my surprise I found that in a great many cases the frequency count tabulation was palindromic in character. A few examples:

Digit Frequency for

Fraction	0	1	2	3	4	5	6	7	8	9
1/29	2	3	3	3	3	3	3	3	3	2
1/47	4	5	5	4	5	5	4	5	5	4
1/61	6	6	6	6	6	6	6	6	6	6
1/73	1	1	0	2	0	0	2	0	1	1
1/89	5	5	5	2	5	5	2	5	5	5
1/329	16	14	14	11	14	14	11	14	14	16
1/1861	186	186	186	186	186	186	186	186	186	186

where, for example, 1/73 = 0.0136986301369863... . Has anyone seen anything written on this effect?

TAPER BORE CHECK - Don Laughery. This is one of the options in the "menu and module" TI-74 demonstration program on pages 19-21. The program listing is lines 300 through 450 on page 21. The program solves for taper bore parameters given input of the diameters of two precision balls and the dimensions from the face to the top of the balls when placed in the taper bore. See the diagram at the right. The larger ball's diameter must not exceed the face diameter times the secant of the half-angle. The smaller ball's diameter must be larger than the exit diameter times the secant of the half-angle. The dimension from the face to the top of the larger ball is negative if the ball projects above the face. To check your entry of the program use the following sample problem:



Large Ball Diameter = 1.0625
Small Ball Diameter = 1.000
Face to Large Ball = 0.410
Face to Small Ball = 2.950

Then, the Face Diameter is 1.0860 and the Included Angle is 1.4274 degrees.

A "MENU AND MODULE" PROGRAM FOR THE TI-74 - D. Laughery and P. Hanson

One of the attractive features of the TI-95 is the built-in file system capability which permits the user to store programs and data files in either the calculator's file space or in a Constant Memory Cartridge. The result is that the user can build a library of programs which can be easily accessed. The Radio Shack Model 100 offers a similar capablity where one can maintain up to nineteen data files or BASIC programs in the user RAM at one time. A menu screen provides immediate access to any of the files or programs. The software cartridges provided for the TI-74 provide a similar kind of menu driven access to a variety of programs, but no such capability is built-in for user-generated BASIC programs. How might we develop a capability to store a several BASIC programs simultaneously in user RAM of the TI-74? We could simply write the various routines in sequence in user RAM and access the desired routine by

* using RUN statements with line numbers to enter the program at the appropriate location, or by

* providing a menu in user RAM with access to the appropriate routine through the use of ON-GOTO or ON-GOSUB statements.

Both of those methods have the disadvantage that the user must use care to keep track of the variables and the dimensioning of variables in the various routines to avoid conflicts. A better solution is to use menu driven subprograms. A sample program was written to demonstrate the concept. The sample program:

* Displays or prints a directory of the subprograms available which was modeled on that used in the software cartridges.

* Demonstrates the use of argument lists with CALL statements and parameter lists with SUB statements to provide transfer of data between several subprograms.

* Demonstrates the call of one subprogram from within another.

* Runs Don Laughery's confidence limits and taper programs which are described elsewhere in this issue.

* Provides subprograms to enter a series of numbers, calculate some statistics on the numbers, and sort the numbers using a heap sort technique.

The program listing appears on page 21. Comments on the details of the program follow:

Lines 100-195 mechanize the menu function. The dimension statement at line 100 is needed to permit transfer of array data between three of the module programs. The accept statement at line 120 converts all alphanumeric input to uppercase to ensure compatibility with the call statements to follow. The call statements for the SORT, STATS and INPUT subprogram modules include argument lists to permit transfer of data between the subprograms. Additional modules can be added by adding similar IF...THEN CALL... statements between lines 120 and line 195. A printout from the DIR subprogram appears at the right.

```
CONF  Confidence Bounds
TAPER Bore check w balls
AREA  Find polygon area
PRINT Printer Control
INPUT Vector Entry
STATS Vector Statistics
SORT  Vector Sorting
DIR   Directory
```

"Menu and Module" Program for the TI-74 - (cont)

Lines 200-295 are a subprogram which calculates confidence bounds based on an article on page 59 of the February 1989 issue of Quality. The routine is described in detail with a sample printout on page 26. Note that the variable X can be used in the subprogram even though the variable X was defined as an array in the main program. This could not be done if the module had been implemented with ON GOTO or ON GOSUB statements.

Lines 300-450 are a subprogram by Don Laughery for performing taper bore check calculations. The routine is described in more detail on page 18.

Lines 500-580 are a subprogram which calculates the area enclosed by a number of vertices. The routine is essentially the same as that on V11N2P22.

Lines 600-650 are a subprogram which is called by five other subprograms (CONF, SORT, STATS, INPUT and DIR) to provide printer setup if desired. The argument PN is needed to pass the variable which defines whether to print or display between the calling subprogrfams and the PRINT subprogram.

Lines 700-790 are a subprogram which provides the ability to build a one dimensional array which can be accessed by other subprograms. Note that it would not be necessary to use the same array name as in the main program.

Lines 800-895 are a subprogram which calculates elementary statistics for an one dimensional array. The program was adapted from the one on V11N3P21. The elements of the array must be provided by another subprogram, in this demonstration by the INPUT subprogram. It is not necessary to use the same array name as in the main program or as in the INPUT subprogram.

Lines 900-990 are a subprogram which performs a heap sort on a one dimensional array. The routine is essentially the same as that on V10N3P13-16 which was adapted from a program on page 137 of the September 1980 issue of Creative Computing. Again, it is not necessary to use the same array name as in the main program or as in the INPUT subprogram.

The sample printout at the right illustrates the output from use of the INPUT, STATS and SORT subprograms.

```
X(1)  .8225408469
X(2)  .163164479
X(3)  .4352797351
X(4)  .3233738121
X(5)  .1733442624
X(6)  .5765405918
X(7)  .6272445458
X(8)  .4822732445
X(9)  .9196005246
X(10) .29828705

Min  =  .163164479
Max  =  .9196005246
Mean =  .4821649092
S.D. =  .2572322652
RMS  =  .5404022521
RSS  =  1.708901969

Sorted List:

 .163164479
 .1733442624
 .29828705
 .3233738121
 .4352797351
 .4822732445
 .5765405918
 .6272445458
 .8225408469
 .9196005246
```

Lines 1000-1099 are a subprogram which either prints or displays a directory of the subprograms in the menu. Additional DATA statements should be added between lines 1040 and 1098 to match any subprograms which are added. The null string for the data in line 1098 is used by line 1020 to detect the end of the directory.

"Menu and Module" Program for the TI-74 - (cont)

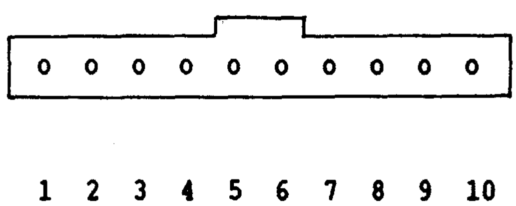
Program Listing

```
100 DIM X(100)
105 DISPLAY "Enter DIR t
o display directory"
110 PAUSE 3
115 DISPLAY "Program Nam
e?";
120 ACCEPT AT(15)VALIDAT
E(UALPHA),A$
125 IF A$="TAPER"THEN CA
LL TAPER
130 IF A$="CONF"THEN CAL
L CONF
135 IF A$="AREA"THEN CAL
L AREA
140 IF A$="SORT"THEN CAL
L SORT((N),X())
145 IF A$="STATS"THEN CA
LL STATS((N),X())
150 IF A$="INPUT"THEN CA
LL INPUT(N,X())
155 IF A$="PRINT"THEN CA
LL PRINT
190 IF A$="DIR"THEN CALL
 DIR
195 GOTO 115
200 SUB CONF
205 REM Calculation of c
onfidence, D. Laughery,
2/8/89
210 REM Adapted from pag
e 59 of Feb 1989 Quality
215 CALL PRINT(PN):PAUSE
 ALL
220 INPUT "Pop. % bounde
d by sample? ";P
225 IF PN=1 THEN PRINT #
1,P;" percent confidence
":PRINT #1
230 IF PN=1 THEN PRINT #
1,"Sample          Conf"
:PRINT #1
240 P=P/100
245 X=1-P
250 FOR N=3 TO 100
260 IF N>15 THEN N=N+4
270 CONF2=(INT(1000*(1-(
P^N+N*X*(P^(N-1))))))/10
00
280 IF PN=0 THEN DISPLAY
:"Sample = ";N,"Conf = "
;CONF2:GOTO 290
285 PRINT #1,N,CONF2
290 NEXT N
295 SUBEND
300 SUB TAPER
305 REM Taper bore check
, Don Laughery, 2/16/89
310 DEG:PAUSE ALL
315 INPUT "Large Ball Di
a.=? ";D1:R1=D1/2
320 INPUT "Small ball di
a.=? ";D2:R2=D2/2
325 INPUT "Dim from face
 to large ball=? ";A:A=A
+R1
330 INPUT "Dim from face
 to small ball=? ";B:B=B
+R2
335 M=B-A:ANG=ASIN((R1-R
2)/M)
340 DF=2*((A+R1*SIN(ANG)
)*TAN(ANG)+R1*COS(ANG))
345 DISPLAY USING 410;DF
350 DISPLAY USING 420,2*
ANG
355 DISPLAY "1=Dia~Depth
/2=Depth~Dia/3=New ";
360 ACCEPT VALIDATE("123
")SIZE(1),X
365 ON X GOTO 370,390,32
5
370 INPUT "Enter Specifi
ed Depth ";T
375 DT=DF-2*(T*TAN(ANG))
380 DISPLAY USING 430;DT
;T
385 GOTO 355
390 INPUT "Enter Specifi
ed Diameter ";DT
395 T=((DF-DT)/2)/TAN(AN
G)
400 DISPLAY USING 440,T,
DT
405 GOTO 355
410 IMAGE FACE DIA=###.#
###
420 IMAGE INCL ANGLE= ##
#.#### DEG
430 IMAGE DIA=###.#### A
T ###.#### DEPTH
440 IMAGE DEPTH= ###.###
# AT ###.#### DIA
450 SUBEND
500 SUB AREA
505 REM Area Finder from
 V11N2P22, TI PPC Notes
510 DIM X(31),Y(31)
515 INPUT "Number of ver
tices? ";N
520 IF N<3 OR N>30 THEN
515
525 FOR I=1 TO N
530 INPUT "X("&STR$(I)&"
)=? ";X(I)
535 INPUT "Y("&STR$(I)&"
)=? ";Y(I)
540 NEXT I
545 X(N+1)=X(1):Y(N+1)=Y
(1)
550 S=0
555 FOR I=1 TO N
560 S=S+X(I)*Y(I+1)-X(I+
1)*Y(I)
565 NEXT I
570 PRINT "Area = ";S/2
575 PAUSE
580 SUBEND
600 SUB PRINT(PN)
605 REM Printer Control
Routine
610 DISPLAY "Use printer
? ";
615 ACCEPT AT(14)VALIDAT
E(UALPHA),A$
620 IF A$="Y"THEN PN=1 E
LSE 650
625 PRINT "PC324, use 12
; HX1000, use 10":PAUSE
2
630 INPUT "Enter device
number ";D$
635 OPEN #1,D$,OUTPUT
640 IF D$="10"THEN PRINT
 #1,CHR$(18)
645 PRINT #1
650 SUBEND
700 SUB INPUT(N,X())
705 REM Vector Entry Rou
tine
710 CALL PRINT(PN)
715 INPUT "Number of ele
ments? ";N
720 IF N>100 THEN 710
725 FOR I=1 TO N
730 A$="X("&STR$(I)&") "
735 INPUT A$&"= ? ";X(I)
740 IF PN=1 THEN PRINT #
1,A$;X(I)
745 NEXT I
750 IF PN=1 THEN PRINT #
1:CLOSE #1
755 PAUSE 0
790 SUBEND
800 SUB STATS(N,X())
805 REM Statistics for V
ector Elements - adapted
 from V11N3P21 of TI PPC
 Notes
810 CALL PRINT(PN)
815 MIN=X(1):MAX=X(1)
820 FOR I=1 TO N
825 S1=S1+X(I)
830 S2=S2+X(I)*X(I)
835 IF X(I)<MIN THEN MIN
=X(I)
840 IF X(I)>MAX THEN MAX
=X(I)
845 NEXT I
850 PAUSE ALL
855 PRINT #PN,"Min  = ";
MIN
860 PRINT #PN,"Max  = ";
MAX
865 PRINT #PN,"Mean = ";
S1/N
870 PRINT #PN,"S.D. = ";
SQR((S2-S1*S1/N)/(N-1))
875 PRINT #PN,"RMS  = ";
SQR(S2/N)
880 PRINT #PN,"RSS  = ";
SQR(S2)
885 IF PN=1 THEN PRINT #
1:CLOSE #1
890 PAUSE 0
895 SUBEND
900 SUB SORT(N,X())
902 REM Heap sort adapte
d from page 137 of Sept
1980 Creative Computing
904 REM Also see V10N3P1
3-16 of TI PPC Notes
906 CALL PRINT(PN)
908 M=N
910 FOR L=INT(N/2)TO 1 S
TEP -1
912 B=X(L)
914 GOSUB 960
916 NEXT L
918 L=1
920 FOR M=N-1 TO 1 STEP
-1
922 B=X(M+1):X(M+1)=X(1)
924 GOSUB 960
926 NEXT M
928 PAUSE ALL
930 PRINT #PN,"Sorted Li
st:"
932 IF PN=1 THEN PRINT #
1
934 FOR I=1 TO N
936 PRINT #PN,X(I)
938 NEXT I
940 IF PN=1 THEN PRINT #
1:CLOSE #1
942 PAUSE 0
944 SUBEXIT
960 REM Subroutine
962 I=L
964 J=I+I
966 IF J>M THEN 978
968 IF J=M THEN 972
970 IF X(J+1)>X(J)THEN J
=J+1
972 IF B>=X(J)THEN 978
974 X(I)=X(J):I=J
976 GOTO 964
978 X(I)=B
980 RETURN
990 SUBEND
1000 SUB DIR
1005 REM Directory of Pr
ograms
1010 CALL PRINT(PN)
1015 PAUSE ALL
1020 READ A$:IF A$=""THE
N 1030
1025 PRINT #PN,A$:GOTO 1
020
1030 IF PN=1 THEN PRINT
#1:CLOSE #1
1035 PAUSE 0:SUBEXIT
1040 REM Listing of Subp
rograms
1051 DATA CONF  Confiden
ce Bounds
1052 DATA TAPER Bore che
ck w balls
1053 DATA AREA  Find pol
ygon area
1054 DATA PRINT Printer
Control
1055 DATA INPUT Vector E
ntry
1056 DATA STATS Vector S
tatistics
1057 DATA SORT  Vector S
orting
1097 DATA DIR   Director
y
1098 DATA ""
1099 SUBEND
```

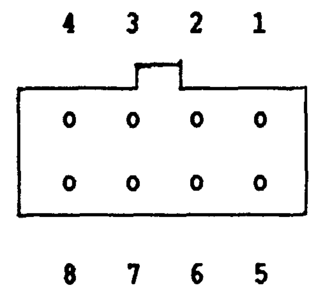
Real Estate/Investment Library Module for the TI-59. The pocket-sized Quick Reference Guide is available. The full-sized book is not. Send five dollars to TI PPC Notes.

<u>INTERFACE INFORMATION FOR THE TI-74 AND CC-40</u> - B. V. Tackach of Wahroonga, New South Wales, Australia. The enlightening discussion of cabling to connect the TI-74 with the CC-40 peripherals in V12N3P13 was a little confusing. The numbering and lettering of the TI-74 and CC-40 hex bus ports was not consistent with the pin numbering used by TI in its various applications notes and user manuals. In addition some additional information is needed for successful use of the CC-40 peripherals with the TI-74, or with the CC-40 for that matter.

<u>Relative Pin Position Diagrams</u>



TI-74 DOCK-BUS Interface



CC-40 HEX-BUS Interface

<u>DOCK-BUS/HEX-BUS Interface Cable Connections</u>

TI-74 DOCK-BUS			CC-40 HEX-BUS	
Description	Signal	Pin	Signal	Pin
System power distribution - output	PO	1*		
System power distribution - input	PI	2*		
Data bit - least significant bit	D0	3	D0 Data-LSB	1
Data bit	D1	4	D1 Data	2
Data bit	D2	5	D2 Data	7
Data bit - most significant bit	D3	6	D3 Data-MSB	8
Handshake - I/O timing control line	HSK	7	HSK	5
Bus Available - I/O traffic control line	BAV	8	BAV	3
System reset line	RESET	9*		
Common ground line	GND	10	GND	4
			Protect Gnd	6*

<u>Notes:</u>

1. Pin numbers marked with an asterisk are not connected when DOCK-BUS and HEX-BUS are interfaced.

2. CAUTION! Pin #1 of the DOCK-BUS is at +6 volts, supplied by the TI-74. Pin #2 is used to supply all DOCK-BUS peripherals from a common power supply. This is not supported by the CC-40 system.

3. The Protective Ground (pin #6 of the CC-40 HEX-BUS should not be joined to the floating reference power line (pin #10 of the TI-74 DOCK-BUS).

Interface Information for the TI-74 and CC-40 - (cont)

Device Numbers for CC-40 Peripherals.

Model	Device	Remarks	Device Numbers
HX-1000	Printer Plotter	Switch Selectable	10 to 11
HX-1010	Printer 80	Internally changed	16 or 17
HX-2000	Wafertape	Was not released	1 to 7
HX-3000	RS232 & Parallel	RS232 - Internally selectable	20 to 23
		Parallel - Internallly selectable	50 to 53
QD-01	Disk Drive	by Mechatronic	8

Note: If the TI-99/4 Impact Printer Model PHP2500 is used on the parallel port of the HX-3000, the RS-232 board in the printer must be removed. Refer to Appendix D of the printer manual.

Device Numbers for TI-74 Peripherals

Model	Device	Remarks	Device Numbers
		Reserved for RESET-ALL	0
PC-324	Printer		12
QD-02	Disk Drive	by Mechatronic	8

Editor's Note: The TI-74 DOCK-BUS pin assignments are consistent with page 17 of my *TI-74 BASICALC Technical Data Manual*. The CC-40 HEX-BUS pin assignments are consistent with those in my HX-3000 *RS232 Users Manual*. Some of the confusion resulted from Maurice Swinnen's definition of the hex-bus interface by "looking into the cable" not by "looking into the device". Other information which is helpful in the construction of a cable between the the TI-74 and a CC-40 peripheral includes the color coding of the wiring in a Hex-bus cable:

Pin	Color	Pin	Color
1	grey	5	brown
2	yellow	6	green
3	red	7	black
4	orange	8	blue

I still have four of the sixteen inch cables with a hex-bus connector at one end. Send one dollar if you want one.

Finally, I remind our members that the connection of the TI-74 to CC-40 peripherals has not been approved by TI, and may result in loss of warranty.

EXTENDED PRECISION PROGRAMS FOR THE TI-59 - V11N2P25 announced that Robert Prins had compiled a 98 page treatment on extended precision programs for the TI-58 and TI-59. Robert reports that he has three copies remaining. He will sell each copy for ten dollars including shipping. He suggests that you write first to confirm that copies are available. His address is in the item at the bottom of page 17. If you are into extended precision calculations for the TI-59 you should get a copy of this.

SOME MORE HISTORY: HEX CODES - Newer members have asked for material on some of the older TI-59 special techniques. One such technique was the implanting of hexadecimal codes to implement fast mode (h12) and high resolution graphics (h25); however, the hexadecimal code method for entering fast mode has largely been supplanted with the Stflg at the end of the current partition technique. The principal investigators of hexadecimal codes (such names as Michael Sperber of the Federal Republic of Germany, Patrick Acosta of San Antonio, Texas, Dejan Ristanovic of Yugoslavia and Dave Leising of Grand Rapids, Michigan come to mind) found other unusual effects with hexadecimal codes. Patrick found that h24 provided the ability to write a program for the TI-58C (but not for the TI-59) which can interpret keycodes in reverse order. An example appeared in V8N1P17. Dejan found some unusual and largely unexplained printouts in List and Trace modes. Examples included such instructions as OSS*, *GT, and *ST. The editor found other curious instructions such as *SB, V2N, TLR and NR/. The TLR printout, which was generated when the Trace mode encountered an h32, appeared to clear all memory.

Often, the results from these experiments yielded erratic results. For example, in V7N6P8 Dejan found that the response to a hex code might be different depending on whether or not the PC-100 was attached. In mid-1981 Patrick, who used a TI-58C, found that an h01 following an Op 00 to Op 07 command would act as a GO* using the display value. He estimated that the use of the technique might save another second in the great calendar race. However, we were unable to demonstrate the same effect with the TI-59 connected to the PC-100.

A thorough discussion of the techniques for implanting hexdecimal codes and of the results achieved would extend to many pages. Newer members who would like to experiment with some of these techniques can obtain a twelve page compilation by sending two dollars. Stamps or cash. No two dollar checks, please.

WHAT DOES IT DO? - Several years ago, when the first TI-66's came out, Dave Leising reported that the units all had a particular instruction sequence in memory. He postulated that the sequence was some sort of self-check or pre-shipment check, but he couldn't determine the purpose. At the time I had already erased (2nd CP) the memory in my TI-66. Recently, I had access to new TI-66, fresh out of the box. I connected the unit to a PC-200, pressed 2nd List, and got the printout at the right. I still can't see what useful function is mechanized. Any ideas out there?

```
     ST
000  INV
001  1/X
002  X2
003  INTG
004  LBL
005  D
006  1
007  2
008  4
009  8
010  STO
011  01
012  2
013  SUM
014  01
015  R/S
016  IX
017  0
018  0
```

ADDRESS CHANGE - V13N1P5 reported that Scott Garver had completed a number of conversions for programs from the TI-59 for the TI-95. He asked that users send a SASE for further information program availability. Scott's address has changed to 2013 Sierra Drive, Pekin IL 61554

CLEARANCE SALE - Newer members have asked about back issues of our newsletter. We have a limited number of copies of the original printing for the three years from 1982 through 1984. Each year includes about 150 typewritten pages. The 1982 issues contain about 20 pages of material on the TI-88 which was announced but never offered to the public. The 1983 and 1984 issues combined contain about 28 pages of material on the CC-40. The 1984 issues contain 11 pages of material on the TI-66. All three issues contain lots of TI-59 programs. We will sell the issues from those three years for ten dollars per year, including shipment. Complete sets of the original printings are NOT available for other years (1980-1981 and 1985-1987/8). Write for details and costs if you are interested in those years.

A PROGRAMMING CHALLENGE - Charlie Williamson writes: Classic number theory wants a remainder that is non-negative and less than the absolute value of the divisor. In other words, the result A/B must satisfy

$$A = B*Q + R \quad \text{where} \quad 0 \leqq R < B$$

Some examples of the results of using this definition are:

```
For:   9/4 :     9  =  4*2    +  1
     -10/3 :   -10  =  3*(-4) +  2
     -9/-4 :    -9  =  (-4)*3 +  3
```

The challenge is to find the quotient and the remainder without the use of direct comparison logic.

Editor's Note: While classic number theory may insist on non-negative remainders some recent programs in TI PPC Notes have worked perfectly well with negative values where we normally expect positive values. An example appeared in V12N3P21. Charlie, who is from Sacramento also notes that

F L O R I D A = R I D O F L.A.

a solution the local chamber of commerce would love.

HARDWARE WANTED

B. V. Takach at P.O. Box 114, Wahroonga 2076, New South Wales, Australia would like to buy a CC-40 Assembler Cartridge and manual.

Lars Herold Andersen at 5 Ravnsbjerg Hegn, DK 7400 Herning, Denmark would like to buy an SR-50A, an SR-52 and an SR-56.

Michael Sperber at Birkenallee 67, 8526 Bubenreuth, Federal Republic of Germany would like to buy a SR-52.

ADDRESS CHANGE - The instructions for access to former PPX programs on V13N1P10 show an out-of-date address for Code H, Thomas Wysmuller. His new address is 441 Wolcott Hill Road, Wethersfield, CT 06109.

CALCULATION OF CONFIDENCE BOUNDS - Don Laughery

This program is one of the options in the "menu and module" program for the TI-74 on page 19. The program is an adaptation of a program in Gerald Hurayt's article "Bounded by Extreme Values" which appeared on page 59 of the February 1989 issue of *Quality*. The program generates a table of sample size and confidence interval for a given confidence level. A listing of the adaptation of the program for the TI-74 appears as lines 200-295 of the "menu and module" program. The printout at the right is for a 95% confidence level.

95 percent confidence

Sample	Conf
3	.007
4	.014
5	.022
6	.032
7	.044
8	.057
9	.071
10	.086
11	.101
12	.118
13	.135
14	.152
15	.17
20	.264
25	.357
30	.446
35	.527
40	.6
45	.665
50	.72
55	.768
60	.808
65	.842
70	.87
75	.894
80	.913
85	.93
90	.943
95	.954
100	.962

Editor's Note: This program is of particular interest for the demonstration of two BASIC language features. The first feature is the changing of the increment in a FOR-NEXT loop while the program is in operation. You might think that one could simply do this by writing a statement FOR N = 3 TO 100 STEP J with J set at one at the start and changed to another value at an selected point in the loop. Unfortunately, the typical BASIC implementation does not respond to that sequence. One method of accomplishing the same thing is illustrated in line 260 of this program which advances the value of N an additional four digits each cycle when N is greater than 15.

The second feature is unique to the BASIC implemented in the TI-74 and the CC-40. Don had observed that the execution time seemed to increase as the program proceeded through the table. The effect can be readily observed during a the printout of the table, where the execution time for the rows with a sample size of 60 or greater is substantially longer than for rows with the sample size less than 60. At first I thought that a some sort of sneak loop had been mechanized which caused a time delay; however, once I had convinced myself that was not the case I proceeded to identify an interesting idiosyncrasy of the TI-74 and CC-40 BASIC implementation--the method of calculating a number raised to an integer power is different for integers of 59 and above than for integers of 58 and below! A similar characteristic is evident with the TI-95. Details appear on the next page.

MORE PPX PROGRAMS - Member John Trask donated the following two PPX programs which were not on the listing in V13N1:

788024 - Astronomical Time and Coordinate Manipulation
788034 - Local Mean Sidereal Time

John also donated his PPX Exchange Software Catalog including the addendums up through the H Update in June 1982. The catalog includes abstracts for each of the programs. I am making the catalog available for loan to interested members. The set weighs 2 lb 14 oz. Send money to cover shipment by Priority Mail (at that weight there is only a small increase over fourth class). I will forward the catalogs to you. I will expect reasonably prompt return so that the catalogs will be available for other members.

AN ANOMALY IN EXPONENTIATION WITH THE TI-74 AND CC-40 - Palmer Hanson. I suspected that the change in execution time of the confidence limits progam at higher values of sample size might be associated with the exponentiation function. I wrote the short program at the upper right to determine the execution time for 100 calculations of an argument raised to different powers. I ran the program with arguments of 0.95 and 2 with the TI-74, CC-40 and Radio Shack Model 100. I also ran a similar program with the TI-95. The execution times in seconds were:

	TI-74		CC-40		Model 100		TI-95	
N	2	.95	2	.95	2	.95	2	.95
2.5	30	34	48	48	33	34	34	38
10	3	4	3	5	2	4	5	7
30	5	11	7	16	3	6	8	14
50	10	18	13	26	5	7	12	21
58	11	21	16	30	5	8	13	23
59	31	36	48	48	5	9	14	24
70	30	36	47	49	5	9	16	28
100	30	36	46	46	6	9	34	38
200	30	35	45	47	7	10	35	39
300	31	36	45	47	OV	11	37	39

```
1 INPUT "N= ";N
2 FOR I=1 TO 100
3 P=.95^N
4 NEXT I
5 GOTO 1
6 END
```

```
1 INPUT "N= ";N
2 P=1
3 FOR I=1 TO N
4 P=.95*P
5 NEXT I
6 PRINT P
7 PAUSE
8 GOTO 1
```

For integer exponents the execution time on the TI-74 and CC-40 increases more or less (within my timing errors) linearly over the range from 10 to 58. The execution time for non-integer exponents (2.5) and for integer exponents above 59 is a constant. By contrast, the execution time for the Model 100 and integer exponents increases linearly over the range shown (and continued over extended ranges as well), but showed a much longer execution time for non-integer exponents. The execution time for the TI-95 increases linearly up to an exponent of 100.

I then compared the values obtained with three methods of exponentiation, 0.95^N, EXP(LN(.95)*N), and the short program at the lower right which would find the product of N values of 0.95. I found that for integers of N of 58 and below the value of 0.95^N agreed with the product of N values of 0.95, but for integers of 59 and above the values of 0.95^N agreed with EXP(LN(.95)*N). These results support the idea that for integer values of N the calculation of A^N is calculated differently depending on the value of N. The product method of evaluating A^N where N is an integer was apparently used for small N to improve execution time. But, why does the change in method occur between N of 58 and 59? The table above shows that execution time is longer for an argument of 0.95 than for an argument of 2. With an argument of .987654321 the execution time with N of 58 or 59 are the same, 39 seconds with the TI-74 and 53 seconds with the CC-40. Thus, the change in calculation method corresponds with the point at which the EXP(LN(A)*N) solution becomes the faster method.

The Model 100 calculations do not follow the same pattern. The calculation of A^N by the three methods typically yields small differences between the three methods. For the TI-95 the situation is even more complex with agreement between various methods of calculation seeming to depend on the range of values used for the argument. I will try to cover that in more detail in the next issue.

In the process of this investigation I also found that for negative arguments and integer exponents the TI-74, the CC-40, and the TI-95 all calculate A^N properly. V8N3P16 reported that Charlie Williamson had noted that the HP calculators properly evaluated a negative number raised to an integer power. V11N4P19 discussed the different responses of the TI-59 and TI-95 for powers and roots with negative arguments, but only hinted at the capability to raise a negative number to an integer power. Somehow, I failed to make that observation in over five years of experience with the CC-40.

ON NEW MACHINES, ELIMINATION OF GOTO'S, STRUCTURED PROGRAMMING, AND THE LIKE

Have you noticed that computer system users often seem to be treated like a "poor relations." What I mean by that statement is that computer system designers chase the latest trends with nary a concern for inconvenience to the users. One aspect of that attitude is the promulgation of a never-ending procession of computer programming languages such as JOVIAL, PASCAL, ADA, C, and the like. We are told that BASIC is an obsolete and ineffective language (I love BASIC), and that the use of GOTO's should be banned from the surface of the earth (I love GOTO's). It's as if there is a cadre of programming language personnel out there somewhere, and we are obliged to keep them occupied, no matter what the cost to the user--a high technology version of the WPA, if you will.

I saw one example of the phenomenon many years ago. The company where I worked had a timeshare system known as the Honeywell Computer Network (HCN) in place. The system was seldom down, it had a powerful BASIC capability including options for double precision and matrix algebra , a FORTRAN capability, and a marvelous interactive program for learning the system. I had introduced many new users to the system by simply showing them how to sign on, entering the system command TEACH SYSPRN, and turning them loose. But the system was built around an older computer, and the time came when the "powers that be" decided that the system was obsolete and must be replaced. When users were introduced to the new system they found that BASIC programs written for the existing system would not run on the new system. As I recall, even the subscripting convention was changed. A group of users suggested that the new system be altered to be "transparent to the BASIC user." The system programmers said that wasn't possible--what that really meant was that they simply didn"t want to do it that way, and they didn't care about inconvenience to the end user. The result was that many end users avoided use of the new system if at all possible. Programmable calculators and personal computers were becoming available, and the users transferred their "workhorse" programs to systems over which they had at least a modicum of control.

More recently, I have been unhappy with some aspects of the TI-95 implementation; for example, the needless elimination of t register testing made translation of TI-59 programs more difficult than necessary. However, several correspondents chided me on my resistance to change, and I tended to write off my objections as the ramblings of a crotchety old man who was far to set in his ways. So I was particularly interested in the following well written comments of member William Hitt on his frustrations with another widely ballyhooed new computing system, the HP-28S:

"... The first and biggest disappointment was quickly evident; my library of calculator programs could not be translated to run on the 28S. Its programming system is so radically different that translation is out of the question; one has to completely rewrite the programs. This meant that my notebooks of programs, accumulated over several years, would be worthless if I chose to make the 28S my main machine. This was not a difficulty encountered with any other HP machine: e.g., 41C programs were easily translated into BASIC for the 71B.

The second difficulty is the programming language itself, RPL as it is called. I have done some programming in FORTRAN, BASIC, and Pascal as well as for HP hand-helds, and nothing I've encountered is quite as weird as RPL. It generates more error signals for me than all the other languages combined. I think the problem stems in part from the attempt to implement structured programming on a hand held machine. This is a little like assuming that because a 747 requires a large tail fin, one should also be put on a Volkswagon. Structured programming I know is a very trendy thing, in fact almost a sacred cow. I am ready to agree that for very large programs, particularly those composed by a team of programmers and maintained by still other programmers, structured programming is necessary. To extend that decision and say that it is the best kind of programming for all machines in all circumstances is ridiculous, and it is time somebody said so. Even the names are value laden: "structured" versus "unstructured." Try "free branching" versus "restricted branching." Personally I prefer the freedom to branch where I want to when I want to; and I don't like to be told "that is not permitted." I like the ability to exit from the middle of one loop and go to the middle of another, if that is what the problem requires. This may result in code that is complicated and difficult to read, but to paraphrase Rhett Butler, "Frankly, I don't give a damn." That my programs may not be crystal clear to someone else bothers me almost as much as what color gown Princess Di will wear to the next ball. I am concerned that my programs are efficient, compact, relatively bug free, and user friendly, but since I am the only one who uses them and since I never confuse them with a light novel, readability is the least of my worries. ..."

To which the editor can only add, Hallelujah, and Amen!