TI PPC NOTES

NEWSLETTER OF THE TI PERSONAL PROGRAMMABLE CALCULATOR CLUB

P. O. Box 1421, Largo, FL 34649

Volume 13, Number 4 Fourth Quarter 1988/1989

This issue might appropriately be called a clean-up issue. There is not much entirely new material, but there are several extensions of previously published programs such as a faster smallest circle program for the TI-74 and a demonstration program for plotting on the HX-1000 with the TI-95. There is a review of the newly available TI-68 which is not really a programmable as we know it. The really encouraging news is the impending availability of peripheral hardware for the TI-74 such as the PC Interface Cable and a 32K RAM. Those items should be useable with the TI-95 as well. Finally I call your attention to the item at the bottom of page 9 which indicates the wide range of our members capabilities.

With this issue we have completed the ninth subscription "year" of our newsletter under the name TI PPC NOTES, and the thirteenth year since the first publication of our predecessor 52 NOTES. Quotation marks around the word year recognize that we typically use about 16 months to publish the four issues in a subscription year.

A subscription form for another set of four issues is attached. You will see that the price for a U.S. subscription has been raised from sixteen to seventeen dollars, the first increase since 1985. If you decide not to continue your membership for the future I would like to receive a note to that effect.

TABLE OF CONTENTS

Happy Holidays!

ERRATA

Double Exponential Evaluation of Data on the TI-59 - V13N3P15. Some readers have pointed out that that previous issues of our newsletter have typically defined equations of the form y = Ae^Bx as exponential curves and equations of the form y = Ax^B as power curves. This is consistent with references such as the documentation for the *Real Estate/Investment* and *Statistics* modules for the TI-59, the *TI-95 Statistics Library*, the *fx-7000G Owner's Manual*, and Kolb's *Curve Fitting for Programmable Calculators*. (Some references such as Spiegel's *Statistics* call the y = Ax^B equation a geometric curve.) If we had followed the typical useage then the solution in V13N3P15-17 should have been called a double power evaluation. Hopefully, anyone who read the documentation accompanying the program understood what we meant.

Thoughts About Curve Fitting - V8N1P27-31. The instructions in paragraph 3 on page V8N1P31 for finding the first unused location for X values are incorrect. The location can be found with the keyboard sequence RCL 10 + 20 = .

There is also an inconsistency between the discussion of weighting in the text and that used in the program on V8N1P30. The text defines the weighting as equal to Y_i^2. If you decipher the program you will find that the weighting is actually equal to Y_i^2 divided by the average of the squares of the Y values. In GWT-60 the normalization factor is obtained in program steps 058 through 077. This normalization reduces the magnitude of the sums and has the interesting property of making the sums look more like those in standard least-square normal equations; e.g., with the value of n, within calculator error, in R03. For the illustration from V8N1P29 the contents of registers R01 through R06 of the TI-59 without weighting and with and without the normalizing factor will be

	Weight	= 1	= Y_i^2	= $Y_i^2/((\Sigma Y_i^2)/n)$
R01	Σwx	22.97767431	54290.77409	33.11243389
R02	Σwx	77.04712422	227445.7842	138.7212399
R03	Σw	8	13116.71	7.999999999991
R04	Σwy	10.96127785	27836.82223	16.97792951
R05	Σwy	19.03588166	60009.4662	36.6003159
R06	Σwxy	38.14400257	116815.0022	71.24652582

Not surprisingly, the solution is not affected very much. For the weighted portion of the demonstration problem on V8N1P29 the values for a and b for the wieghts above are:

1.822732024	1.659924268	1.659924270
1.658107993	1.711531317	1.711531316

where the applicstion of the normalization factor only changes the solution in the ninth decimal place. A revised version of the program which permits the solution with or without weighting appears on pages 6 and 7 or this issue. The use of a normalizing factor is also mentioned in paragraph 67 of Deming's *Statistical Adjustment of Data* (page 201 of the Dover edition).

LATEST TI-95 PRICES - The #46 issue of the EduCALC catalog offers the following prices for the TI-95. $49.95 for the TI-95 and $5.00 for each of the three library cartridges, Mathematics, Statistics, or Chemical Engineering. You must buy one cartridge with each calculator, and you may buy up to three cartridges with each calculator. You cannot buy cartridges without buying a calculator. The prices are available only as long as their stock lasts. You can order by telephone using a credit card by calling 1-800-633-2252, extension 356.

Page 14 of DAMARK's Christmas 1989 catalog offers the TI-95 for $59.00 plus $6.00 for insurance, shipping and handling. The offer includes the "BONUS mathematics cartridge, statistics, & chemical engineering library cartridge at NO EXTRA CHARGE". I assume that means that you get all three. Ask for Item No. B-452-130583. You can call toll-free at 1-800-729-9000.

If you order, please mention our club. These sales, of course, are only more confirmation that the TI-95 is being phased out. TI continues to indicate that they will provide support for repairs and supplies.

MAKE YOUR OWN PAPER FOR THE PC-324 - Carl Rabe writes: I recently purchased a PC-324 printer and immediately noticed the paper problems that you wrote of in V13N1 (short rolls and high cost). So, I took a large roll of TP-30250 paper for the PC-100 and put it in the vise. With a new fine toothed hack saw blade I carefully cut off the end of the roll to reduce it to a width of 2 1/4 inches. After cutting it off I rubbed the new edge on a cloth to remove dust and smooth it a little. I used a small hairpin to start a roll. I rolled the paper tightly into aa roll about the right size to just fit in the back of the PC-324. I was able to put about fifteen feet on a roll. So far it has worked fine in the PC-324.

Editor's Notes: The contrast on some sample printouts received from Carl is about the same as that obtained with normal PC-324 paper. The contrast is decidedly inferior to that obtained with Radio Shack Catalog No. 26-3592B paper. As noted in V13N3P24 the improved contrast only occurs with the catalog number with a "B" suffix. There is still some "no suffix" paper and some "A" suffix paper still on the shelves of Radio Shack stores, so be sure to look for and get the "B" suffix. The price is the same.

Carl also wrote that he was going to try to suspend the rest of the PC-100 paper roll horizontally in a small cardboard box to put behind the PC-324. He will feed the paper into the back of the printer with the cover slightly ajar. He hasn't reported on the results yet. Of course, that approach is similar in concept to the retractable spindle provided with the HX-1000 Printer/Plotter.

I have searched for larger rolls of 2 1/4 inch wide thermal paper, but so far have only been able to find the blue printing version. Has any member found a source of black printing 2 1/4 inch wide paper on larger rolls?

PAPER FOR THE PC-100 - Some colleagues from the company where I worked before retirement called me to ask about a source of paper for the PC-100. I suggested that they call 1-800-TI-CARES. I understand that they obtained very fast service by using a credit card to order.

PLOTTING WITH THE TI-95 AND HX-1000 - P. Hanson.

In V12N3P23 I reported that I had been able to use Maurice Swinnen's cable from V12N3P13 to connect the TI-95 to the HX-1000 Printer/Plotter but had not been able to successfully send graphics mode commands. I repeated the story in V13N3P23 and asked if anyone had successfully used a TI-95 to plot on the HX-1000. Only a few days after sending out V13N3 I tried the TI-95 and HX-1000 combination again and was able to successfully perform graphics. I do not even know what my problem was on earlier attempts.

A key issue in plotting with the HX-1000 is that the coordinates for the end points of the line segments cannot be variables per se, but must be inserted into the proper position in the text for the command as alphanumerics. This is relatively easy in BASIC where one can use the STR$ function effectively as illustrated in the sample application on pages 30-31 of the HX-1000 *Printer/Plotter Users Manual* or in the example in V10N2P14. It is not so easy with the TI-95.

You might think that one could easily accomplish the equivalent of the STR$ function with the MRG instructions. The problem is that the MRG instruction will include a decimal point after an integer as noted on page 3-19 of the *TI-95 Programming Guide*. In addition, the MRG instruction will insert preceding blanks if the merged value does not fill allotted space. Blanks and decimal points are not acceptable in the graphics commands for the HX-1000. The program at the right provides the same 8 petal rose plot used in the demonstration of the CC-40 and HX-1000 in V10N2P14.

```
0000 `Plotting with the`
0017 ` TI-95 and HX-100`
0034 `0 - 31 July 1989`
0050 DEV 010 WID 24 CMS
0056 CLR CHR 019 PRT CLR
0062 `L(0,-107):1:3,(21`
0079 `5,-107)` PRT CLR
0088 `L(107,0),(107,-21`
0105 `5):1:0` PRT CLR
0113 `M(107,-107)` PRT
0125 CLR `O` PRT CLR `C`
0130 `4` PRT CLR
0133 LBL AA (4* RCL T )
0142 SIN *100= STO R *
0151 RCL T COS = STO X -
0158 RCL U + RCL M FRC =
0165 STO M RCL R * RCL T
0172 SIN = STO Y - RCL V
0179 + RCL N FRC = STO N
0186 7.1 STO I 48
0193 LBL BB STO IND I
0199 DSZ I GTL BB 0 x~t
0206 RCL M INV IF< 2079
0212 GTL CC 3 ST- A
0218 LBL CC RCL N INV
0224 IF< 2079 GTL DD 3
0231 ST- E
0233 LBL DD RCL M ABS /1
0241 00= ST+ B FRC *10=
0251 ST+ C FRC *10=
0258 ST+ D RCL N ABS /10
0266 0= ST+ F FRC *10=
0275 ST+ G FRC *10=
0282 ST+ H CLR `R(0,0),`
0292 `(` CHR IND A
0296 CHR IND B CHR IND C
0302 CHR IND D `,`
0306 CHR IND E CHR IND F
0312 CHR IND G CHR IND H
0318 `)` PRT 2 ST+ T
0323 RCL X STO U RCL Y
0329 STO V 360 INV IF< T
0337 GTL AA CLR CHR 017
0344 PRT DEV 012 HLT
```

Steps 0050-0059 match the characteristics of the TI-95 and HX-1000. The CHR 019 PRT sequence at steps 0057-0060 places the HX-1000 in the graphics mode.

Steps 0062-0112 include two Draw Line commands which plot the X and Y axes. The :1:3, sequence at steps 0071-0075 select the black pen and a dotted line. The :1:0 sequence at steps 0107-0110 return the HX-1000 to the condition which draws solid lines.

Steps 0113-0124 move the pen to the intersection of the X and Y axes.

Steps 0125-0132 designate the intersection of the axes as the new origin and select the red pen for the plot.

Steps 0133-0185 calculate points on the curve (X,Y variables), and then calculate the parameters for use by the Draw Relative Line commands to the HX-1000 (M,N variables). The variables U and V are the end points of the previous line segment of the plot. The RCL M FRC sequence at steps 0161-0163 and the RCL N FRC sequence at steps 0181-0183 carry the fractional parts of line segment components forward from one calculation to the next to avoid truncation effects in the plots.

Plotting with the TI-95 and HX-1000 - (cont)

Steps 0186-0319 assemble the Draw Relative Line commands using the values of M and N. Steps 186-203 form a loop which places the value 48 in data registers 000 through 007 (A through H). With an integer value in the DSZ register a value would not be placed in data register 000. The use of the non-integer value 7.1 in the DSZ register causes the number 48 to be placed in data registers 000 through 007. A demonstration of the effects of DSZ gegister values which contain fractional parts appears on page 5-10 of the *TI-95 Progamming Guide*. The specific technique in the program is borrowed from the solution to George Vogel's fourth TI-59 programming puzzle in V5N9/10P22.

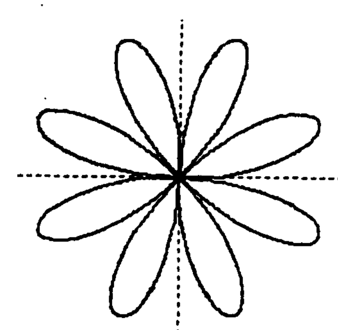
```
0186 10 STO A 0 x~t
0192 RCL M INT IF= 2079
0198 GTL XX IF< 2079
0204 INC A ABS LOG ST+ A
0210 LBL XX RCL A INT
0216 STO B 2 ST+ B RCL N
0223 INT IF= 2079 GTL YY
0230 IF< 2079 INC B ABS
0236 LOG ST+ B
0239 LBL YY RCL M INT
0245 `R(0,0),(`
0253 COL IND A MRG =
0258 RCL N INT OLD
0262 COL IND B MRG =
0267 COL IND A `,`
0271 COL IND B `)` PRT 2
0277 ST+ T RCL X STO U
0283 RCL Y STO V 360 INV
0291 IF< T GTL AA CLR
0297 CHR 017 PRT DEV 012
0304 HLT
```

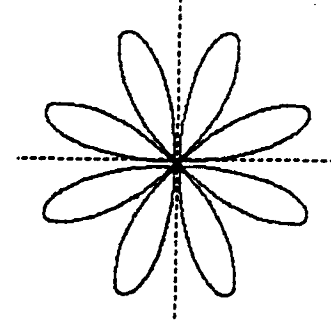
You might think that it would be necessary place integer values in data registers A through H in order for the CHR IND instructions at program steps 293 through 317 to convert the values to alpha characters. However, page 6-3 of the *TI-95 Programming Guide* states that only the integer portion of a numeric field in the pointer register is used.

Steps 0320-0338 increment the angle in the plotted equation (T), redesignate the old end points (X,Y) as the new starting points (U,V), and return to step 0133 to calculate and assemble the next line segment.

Steps 0339-0348 return the HX-1000 to the text mode and return the printer to parameters to that used with the PC-324.

The listing on page 4 included my first ideas for obtaining the string required by the HX-1000 graphic mode. That mechanization assembled the required string piece by piece starting from the right. The listing at the right shows another mechanization where the first 186 steps are the same as in the listing above. The values for the end point for the Draw Relative Line command are assembled using MRG = commands. Then, the decimal points inserted by the MRG commands are replaced by selecting the appropriate columns and writing over with a comma and a close parenthesis as appropriate. This mechanization saves 44 program steps. Yet another mechanization could be obtained using MRG IND commands to save a few more steps.





Two plots of the resulting 8 petal rose appear at the right. The upper plot was obtained with the program as pres ented here. The lower plot was obtained before the calculations to carry forward the fractional parts of line segments from one line segment calculation to the next were included. Note that the petals of the rose are shorter and that the bases of the petals wander away from the origin.

TI-59 SOLUTION FOR POWER FUNCTION - V8N1P27-31 presented George Thomson's program for fitting the power function with weighting. Some correspondence in response to Marcel Bogart's double exponential evaluation in V13N3P15-17 revealed that weighting might not be desired in some cases. In the process of evaluating different cases George's program was modified to include options for operation with or without weighting at the user's discretion. The resulting program is on page 7. Changes relative to the original program include:

* Setting the partitioning is included as a part of mode A.
* An option to solve without weighting is added (Mode D).
* A printed notation is added to indicate the source of the solution; i.e., weighted, unweighted, or user selected.
* The user does not have to calculate the address for the next X value prior to entering the add data mode (now mode A').

User Instructions:

1. To input data: Press A and see 45 in the display. That is the address of the first Y storage location. Enter the value of the first X value and press R/S. The X value is printed and displayed. Enter Y and press R/S. The Y value is printed, and the calculator stops with the address of the next Y storage location in the display. Enter additional pairs as required.

2. To solve with weighting press E. The notation "$W = Y^2$" will be printed when the solution is complete. The calculator will stop with the unrounded solution for a in the display. You have the option to round it or not. Then, press R/S. The value you selected for a will be printed. The calculator will stop with the unrounded solution for b in the display. Again, you may change the value or not. Then, press R/S. The value you selected for b will be printed and the calculator will proceed to print a table of the X input value, the Y input value, the calculated Y value, and the Y difference for each data pair. At the end of the table the mean error and the standard error of the fit are printed. The calculator stops with a flashing zero in the display.

3. To solve without weighting press D. The notation "W = 1" will be printed and the unrounded solution for a will apear in the display. Proceed as in step 2 above to either round the solutions for the coefficients and print the table. Note that you do not need to reenter the data.

4. To add data pairs to the input press A'. The address of the Y storage location to be used appears in the display. Proceed to enter the X and Y values as in step 1 above.

5. To correct an input data pair enter the storage location for the X value in the display. Press D'. The address of the companion Y storage location will be displayed. Re-enter the X and Y values as in step 1.

6. To evaluate residuals for a user selected set of coefficients enter the selected value of a and press B. The value of a appears in the display. Enter the selected value for b and press R/S. The calculator proceeds with a printout of the residuals as in step 2 except that the notation "USER COEF" is printed.

7. To perform single value calculations, but not residuals, enter a and press C. Enter b and press R/S. Enter X and press R/S. The calculator will display and print the value for X and the calculated value for $Y = aX^b$. For additional solutions with the same a and b, enter a new value for X and press C'. See the value of X in the display. Press R/S for the solution.

TI-59 Solution for Power Function - (cont)

8. Change the Fix command at locations 231-232 to select different printout resolution for printout of the input, residuals and mean error. Change the Fix command at locations 295-296 to select a different resolution for the printout of the standard error.

9. Note that the program does not check for zero or negative input values.

000	76	LBL	067	00	00	134	11	11	201	04	4	268	14	14	335	03	3
001	16	A'	068	69	OP	135	95	=	202	00	0	269	95	=	336	02	2
002	43	RCL	069	28	28	136	44	SUM	203	00	0	270	99	PRT	337	01	1
003	10	10	070	97	DSZ	137	01	01	204	04	4	271	94	+/-	338	07	7
004	85	+	071	09	09	138	73	RC*	205	05	5	272	85	+	339	02	2
005	02	2	072	30	TAN	139	07	07	206	07	7	273	73	RC*	340	01	1
006	00	0	073	43	RCL	140	23	LNX	207	08	8	274	08	08	341	69	OP
007	95	=	074	10	10	141	42	STO	208	69	OP	275	95	=	342	03	03
008	61	GTO	075	55	÷	142	12	12	209	03	03	276	99	PRT	343	69	OP
009	10	E'	076	43	RCL	143	65	×	210	76	LBL	277	44	SUM	344	05	05
010	76	LBL	077	00	00	144	43	RCL	211	50	I×I	278	16	16	345	98	ADV
011	11	A	078	95	=	145	11	11	212	69	OP	279	33	X^2	346	43	RCL
012	07	7	079	42	STO	146	95	=	213	05	05	280	44	SUM	347	14	14
013	69	OP	080	18	18	147	44	SUM	214	98	ADV	281	17	17	348	99	PRT
014	17	17	081	61	GTO	148	04	04	215	69	OP	282	69	OP	349	43	RCL
015	47	CMS	082	59	INT	149	65	×	216	12	12	283	27	27	350	15	15
016	02	2	083	76	LBL	150	43	RCL	217	22	INV	284	98	ADV	351	99	PRT
017	00	0	084	14	D	151	13	13	218	23	LNX	285	97	DSZ	352	98	ADV
018	76	LBL	085	86	STF	152	95	=	219	91	R/S	286	09	09	353	61	GTO
019	10	E'	086	01	01	153	44	SUM	220	42	STO	287	39	COS	354	17	B'
020	42	STO	087	01	1	154	06	06	221	14	14	288	43	RCL	355	76	LBL
021	07	07	088	42	STO	155	43	RCL	222	99	PRT	289	16	16	356	13	C
022	85	+	089	11	11	156	11	11	223	32	X:T	290	55	÷	357	42	STO
023	02	2	090	76	LBL	157	65	×	224	91	R/S	291	43	RCL	358	14	14
024	05	5	091	59	INT	158	43	RCL	225	42	STO	292	10	10	359	91	R/S
025	95	=	092	25	CLR	159	12	12	226	15	15	293	95	=	360	42	STO
026	42	STO	093	36	PGM	160	33	X^2	227	99	PRT	294	99	PRT	361	15	15
027	08	08	094	01	01	161	95	=	228	98	ADV	295	58	FIX	362	76	LBL
028	92	RTN	095	71	SBR	162	44	SUM	229	76	LBL	296	04	04	363	18	C'
029	98	ADV	096	25	CLR	163	05	05	230	17	B'	297	43	RCL	364	91	R/S
030	99	PRT	097	43	RCL	164	43	RCL	231	58	FIX	298	17	17	365	99	PRT
031	72	ST*	098	10	10	165	11	11	232	02	02	299	55	÷	366	45	Y^x
032	07	07	099	42	STO	166	65	×	233	98	ADV	300	53	(	367	43	RCL
033	69	OP	100	09	09	167	43	RCL	234	43	RCL	301	43	RCL	368	15	15
034	27	27	101	02	2	168	13	13	235	10	10	302	10	10	369	65	×
035	91	R/S	102	00	0	169	33	X^2	236	42	STO	303	75	-	370	43	RCL
036	99	PRT	103	42	STO	170	95	=	237	09	09	304	02	2	371	14	14
037	72	ST*	104	07	07	171	44	SUM	238	02	2	305	95	=	372	95	=
038	08	08	105	76	LBL	172	02	02	239	00	0	306	34	$\sqrt{X}$	373	99	PRT
039	01	1	106	38	SIN	173	69	OP	240	42	STO	307	99	PRT	374	98	ADV
040	44	SUM	107	43	RCL	174	27	27	241	07	07	308	22	INV	375	61	GTO
041	10	10	108	07	07	175	97	DSZ	242	25	CLR	309	58	FIX	376	18	C'
042	43	RCL	109	10	E'	176	09	09	243	42	STO	310	61	GTO	377	76	LBL
043	07	07	110	01	1	177	38	SIN	244	16	16	311	03	03	378	19	D'
044	61	GTO	111	87	IFF	178	98	ADV	245	42	STO	312	99	99	379	32	X:T
045	10	E'	112	01	01	179	69	OP	246	17	17	313	76	LBL	380	01	1
046	76	LBL	113	58	FIX	180	00	00	247	76	LBL	314	12	B	381	22	INV
047	15	E	114	73	RC*	181	04	4	248	39	COS	315	42	STO	382	44	SUM
048	22	INV	115	08	08	182	03	3	249	43	RCL	316	14	14	383	10	10
049	86	STF	116	33	X^2	183	03	3	250	07	07	317	91	R/S	384	32	X:T
050	01	01	117	65	×	184	07	7	251	10	E'	318	42	STO	385	61	GTO
051	25	CLR	118	43	RCL	185	69	OP	252	73	RC*	319	15	15	386	10	E'
052	42	STO	119	18	18	186	02	02	253	07	07	320	98	ADV			
053	00	00	120	95	=	187	06	6	254	99	PRT	321	69	OP			
054	43	RCL	121	42	STO	188	04	4	255	23	LNX	322	00	00			
055	10	10	122	11	11	189	00	0	256	42	STO	323	04	4			
056	42	STO	123	76	LBL	190	00	0	257	12	12	324	01	1			
057	09	09	[illegible]	58	FIX	191	00	0	258	73	RC*	325	03	3			
058	02	2	125	44	SUM	192	02	2	259	08	08	326	06	6			
059	00	0	126	03	03	193	00	0	260	99	PRT	327	01	1			
060	10	E'	127	73	RC*	194	00	0	261	73	RC*	328	07	7			
061	76	LBL	128	08	08	195	69	OP	262	07	07	329	03	3			
062	30	TAN	129	23	LNX	196	03	03	263	45	Y^x	330	05	5			
063	73	RC*	130	42	STO	197	87	IFF	264	43	RCL	331	69	OP			
064	08	08	131	13	13	198	01	01	265	15	15	332	02	02			
065	33	X^2	132	65	×	199	50	I×I	266	65	×	333	01	1			
066	44	SUM	133	43	RCL	200	06	6	267	43	RCL	334	05	5			

<u>SCOTT GARVER'S CHALLENGE</u> - V13N3P11 challenged members to write a routine for the TI-95 which would place the nomenclature A⁻¹, θ , X÷Y ,∝←ϑ , and Σxy in the windows associated with function keys F1 through F5, where the characters such as ⁻¹, θ , ÷ , etc., are not available directly from the keyboard. The user can bring those characters to the display through use of the CHR instruction as described on page 3-7 of the *TI-95 Programming Guide*. The user cannot use the CHR instruction to bring those characters directly into the function key (DFN) instructions. The responses to the challenge demonstrate that the characters can be entered into a function key definition through access to system RAM, but only after the related DFN instructions have been completed.

Gordon Wilson of St. Croix, V.I. has the honor of submitting the first solution. His routine appears at the left below. The solution provided by Scott with his challenge appears the center, and a solution by the editor, admittedly obtained only after seeing Gordon's and Scott's routines, appears at the right. Comments follow.

Wilson

```
0000 DFN F1:    @AA
0007 DFN F2:    @BB
0014 DFN F3:    @CC
0021 DFN F4:    @DD
0028 DFN F5:    @EE 8390
0039 STO A 15 STO B .121
0049 120246226 SBL AA .1
0063 27224089253 SBL AA
0077 .088160242160
0090 SBL AA .233065160
0103 SBL AA SF 24 SF 25
0110 COL 07 `WOW!` HLT
0117 LBL AA STO C
0122 LBL BB 0 IF= C RTN
0129 RCL C (*3 INV LOG -
0137 FRC STO C )
0141 STB IND A DSZ B
0146 DSZ A GTL BB
```

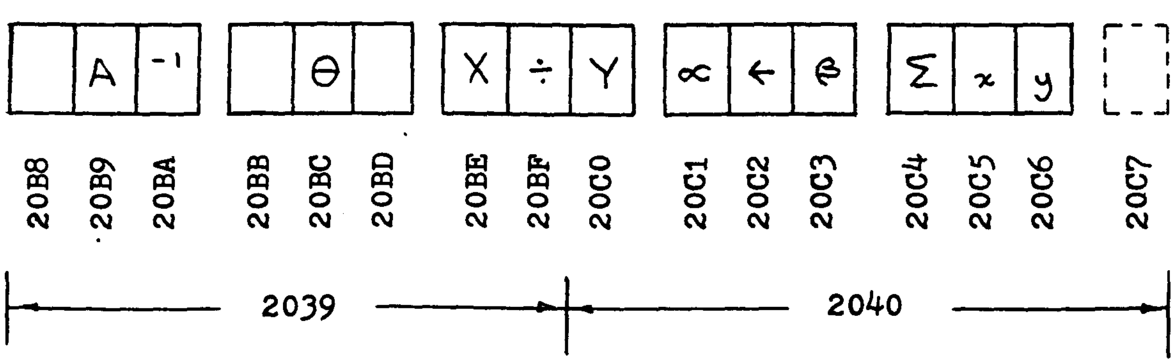
Garver

```
0000 DFN F1: A @AA
0007 DFN F2:    @BB
0014 DFN F3:X/Y@CC
0021 DFN F4:    @DD
0028 DFN F5: xy@ZZ 233
0038 STB 20BA 242
0044 STB 20BC 253
0050 STB 20BF 224
0056 STB 20C1 127
0062 STB 20C2 226
0068 STB 20C3 246
0074 STB 20C4 `      WO`
0085 `W!` HLT
```

Hanson

```
0000 DFN F1:    @AA
0007 DFN F2:    @BB
0014 DFN F3:    @CC
0021 DFN F4:    @DD
0028 DFN F5:    @EE CLR
0036 RCB 20C7
0039 CHR IND 2007 `yx`
0045 CHR 246 CHR 226
0051 CHR 127 CHR 224 `Y`
0058 CHR 253 `X `
0063 CHR 242 ` ` CHR 233
0070 `A ` RCL 2065
0075 STO 2040 RCL 2064
0081 STO 2039 CLR COL 07
0087 `WOW!` HLT
```

Each of the solutions involves entry of character codes directly into the system RAM assigned for storage of the function key definitions (see page C-19 of the *TI-95 Programming Guide*). The following diagram provides a cross reference between the window locations and the corresponding byte addresses and system registers. Note that byte 20C7 is actually part of the flags and addresses for the function key definitions, but is also the least significant byte of the second system register used for the user messages.



Scott Garver's Challenge - (cont)

Gordon Wilson's routine uses STB instructions to place the required three digit character codes in the appropriate bytes. The value 8390 stored in data register A in steps 35-39 is the decimal equivalent of hexadecimal 20C6, the location of the byte which controls the right hand character of the windows. Subroutine AA takes a decimal number from the display, extracts the three digit character codes in sequence, and stores them in the appropriate bytes, starting at the byte associated with the right hand character of the function key definitions. Thus, the number .121120246226 (steps 45-57) includes the character codes for y, x , Σ , and @ . Gordon's routine is amenable to "cookbook" use by simply entering the required character codes in reverse order in segments of up to four character codes each, and remembering to include the leading decimal point. The routine is relatively slow -- the window display does not appear until about three seconds after the routine is called. This is apparently due to the calculations in subroutine AA.

Scott Garver's solution uses the message portion of the DFN instruction to enter those characters which can be entered from the keyboard. It only uses the STB instructions to enter the character codes for the characters which are not available from the keyboard. His routine is much faster than Gordon,s but is not as amenable to a "cookbook" approach.

In the editor's solution the messages are assembled in the display (and in the alpha register) using the alpha mode. Keyboard entered characters are used where possible and CHR instructions where necessary. Then, the contents of the two pertinent alpha registers are transferred to the system registers assigned for the function key messages. It turns out that one byte of one of those system registers (20C7) contains information which results from execution of the DFN instructions. To carry this information forward an RCB instruction retrieves the information in byte 20C7. Subsequenct STO A and CHR IND A instructions insert the equivalent information into the display and the alpha register. Note that the characters must be loaded into the display register in reverse order. I don't yet know why, but it is necessary. The routine responds as rapidly as Scott's and is more amenable to a cookbook approach -- the user simply mechanizes the alpha mode sequence which will place the desired messages in reverse order in the display, while remembering to include steps 0036-0043 at the beginning to retain the information in byte 20C7.

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

ROBERT PRINS SETS A RECORD

Club members know of Robert as a "superprogrammer". The item at the right from the *Hitch-hiker's Guide to Europe* shows that Robert has been busy with other things as well.

3 (Most miles in any 24-hour period) This record, held by Ashok Gupta for several editions, has finally been beaten by *Robert Prins of De Bilt, Netherlands*. Robert hitched an incredible 1297 miles, from 30 km. south of Nis in Yugoslavia to 2 km. east of De Bilt in Holland in just 20 hrs. 47 mins!

THE TI-68 - Palmer Hanson. The TI-68 is one of the newer scientific calculators from TI. It is available from EduCalc for $49.95 plus $1.00 shipping. It provides some impressive capability to solve specific problems such fifth order simultaneous equations and polynomial evaluation. You can also enter formulas into memory, evaluate the formulas, call back the formulas and change them, and obtain solutions to the new formulas. But there are no comparison tests available to the user. So, if you are looking for a *programmable* calculator such as the TI-59, TI-66 or TI-95 then you will be disappointed.

Benchmark Tests:

I tested the TI-68's arithmetic with some of the benchmarks that we have used to test other machines in earlier issues:

1. e x π was equal to π x e indicating that multiplication was commutative. The non-commutative multiply with the TI-59 was discussed in V8N2P15.

2. sin(45) - cos(45) = -1E-13. This type of discrepancy is not as important for machines such as the TI-68 where the user cannot implement comparison tests.

3. The square root - squared test: V8N3P13/14 described this test vwhich is a derivative of the $(\sqrt{2})^2$ test by Brian Hayes on page 136 of the January 1981 issue of *BYTE*. For our test we start with an integer, take the square root five times, take the square five times and compare the result to the original integer. I tested selected integers from 2 through 17. The display returned the starting integer in each case. The actual values before truncation to the display were:

2	1.99999 99999 83	12	12.00000 00001 3
3	3.00000 00000 04	13	12.99999 99998 1
5	4.99999 99999 70	15	15.00000 00002 9
7	7.00000 00000 71	17	17.00000 00000 7

where those results are identical with those reported for the TI-95 on V11N3P16, and all answers are better than those obtained with the TI-59. To get those results I used the 13 digit precision mode. I also had to perform the calculations with the keyboard sequence

$\sqrt{N}$ = $\sqrt{ANS}$ = $\sqrt{ANS}$ = $\sqrt{ANS}$ = $\sqrt{ANS}$ = x^2 = x^2 = x^2 = x^2 = x^2 =

When I tried to enter all the calculations at one time as in

$\sqrt{}\sqrt{}\sqrt{}\sqrt{}\sqrt{}N\ x^2\ x^2\ x^2\ x^2\ x^2$ =

the calculator combined the calculations and return the exact input integer. The same sort of combining of the calculations occurred if I used a lot of parentheses such as

$(((((\sqrt{}(\sqrt{}(\sqrt{}(\sqrt{}(\sqrt{N})^2)^2)^2)^2)^2)$ =

4. 1.0000001 squared 27 times: V9N2P11 described this test from the "Computer Recreations" column of the April 1984 issue of *Scientific American*. The results were:

Exact	674530.47074 10845 59...
Mode A (repeated x^2)	674530.31804 26
Mode B (repeated ANSxANS)	674530.31804 26
Mode C (N^134217728)	674530.46681 15

The Mode A and Mode B results are the same as with the TI-95, CC-40 and TI-74. The Mode B result is better than the TI-59, but much worse than the TI-95.

The TI-68 - (cont)

5. The Bob Fruit Benchmark: Bob proposed a compound interest problem as a benchmark in V8N4P4. The appropriate equation is that for the sum of a geometirc series S = [(1 + i)^n -1]/i . An annual interest rate of ten percent (i = 0.10/12) and compounding monthly for thirty years (n = 360) yields

Exact	2260.48792 47960 86067 ...
TI-68 using the y^x function	2260.48792 4527

where that answer is slightly better than that from the TI-95, but not as good as that from the TI-59.

6. Speed of calculations:

69!	one second
100p50	one second
328c164	three seconds

where the results for the combination test are a factor of three faster than the TI-95 and as fast as the TI-74.

7. I tested the simultaneous equation solution (SIMUL) using a sub-Hilbert matrix (the first row is 1/2, 1/3, 1/4, ..., the second row is 1/3, 1/4, ..., etc.) with ones on the right hand side as suggested by George Thomson in V8N6P18. The previously published results were for the 7 x 7 sub-Hilbert which could be solved by TI-59 programs which called the ML-02 library program as a subroutine and by most other simultaneous equation solutions on the TI-59. Since the TI-68 can only accommodate a fifth order solution the 5 x 5 sub-Hilbert solutions were recalculated for the TI-59 using ML-02 and for the TI-95 using the Linear Equations solution from the Mathematics cartridge. The results were

Exact	TI-59	TI-95	TI-68
30	30.00000352680	29.99999464660	29.9999951204
-420	-420.0000381192	-419.9999399311	-419.999945126
1680	1680.000124591	1679.999799175	1679.999816279
-2520	-2520.000158255	-2519.999741040	-2519.999762855
1260	1260.000068682	1259.999886407	1259.999895898
Rel Error	1.18E-7	1.78E-7	1.63E-07

The relative error is is calculated as (answer - exact)/exact for each answer and the largest of the five errors entered in the table above. In V9N2P18 James Walters suggested that a better figure of merit might be obtained if the solution vector is multiplied by the original matrix and the result compared with the unity vector. The results for the 5 x 5 sub-Hilbert for the various machines are:

1	1.	0.9999999999	1.
1	1.0000000013	1.0000000001	1.0000000003
1	0.9999999997	0.9999999998	0.9999999999
1	1.0000000004	1.0000000001	1.0000000001
1	1.0000000005	0.9999999999	1.
Max error	13E-10	2E-10	3E-10
RMS error	6.6E-10	1.3E-10	1.5E-10

So, the TI-68 provides results which are as accurate as either the TI-59 or TI-95 for this problem. Execution time is another matter. The TI-59 solves the problem in 82 seconds, 57 seconds to find the determinant and 25 seconds to find the solution. The TI-95 needs 16 seconds. The TI-68 solution is available within a second after the last input value is entered!

The TI-68 - (cont)

8. Finally, I tested the accuracy of the polynomial solution (POLY) using Peter Messer's cubic from V11N4P16

$$x^3 - 2x^2 + (4/3)x - 2/9 = 0$$

which has two complex roots and one real root. The following table compares the TI-68 result with the first fifteen digits of the exact solution:

Source	Re	Im	R3
Exact	0.87665 35083 15812	0.36370 78786 57240	0.24669 29833 68375
TI-68	0.87665 35083 160	0.36370 78786 564	0.24669 29833 683

Those results are as accurate as any that we have obtained with various cubic solutions. The solution is returned in less than a second.

There are two TI-68 features that I don't like:

* The internal capabilities such as the simultaneous equation solution and the polynomial evaluation do not place the answers into memory locations where they can be easily used in subsequent calculations. Thus, if you want to use the results in subsequent calculations it is necessary to transfer the results to user-available locations. That's inconvenient at best.

* The "Last Answer" function used to recall a previous result for subsequent calculations is a 2nd function key. That's inconvenient for a frequently used function.

ANOTHER COLLECTOR - New member Jon Dattorro writes: I am interested in acquiring all brands of working hand held calculators that are no longer commercially available. This includes prototypes such as the TI-88 which were never released to the public, and goes all the way back to 1972 with the appearance of the first models from TI and Bowmar. I am most interested in the early TI and HP models, but I also am interested in scientific models from National (NOVUS) and Commodore or any other short-lived manufacturer. I'm also interested in "4-bangers" from the early seventies. Units with bad batteries or power supplies are acceptable as long as they were last known to be working. I am not interested in any machine which was primarily designed to interpret the BASIC language. Write to Jon Dattorro, 2 Beaumont Lane, Devon PA 19333 .

A 32K RAM FOR THE TI-95 - Scott Garver writes: I am now the owner of a fully working 32K RAM cartridge for my TI-95. The conversion is rather straightforward. The greatest problem was in trying to obtain the needed RAM chip. I currently have two additional chips on hand and can order more. I am offering to supply converted cartridges to club members at a cost of $65.00 each for the conversion and the new chip. I will either (1) modify a cartridge that they send to me or (2) purchase a new cartridge and modify it, passing the additional cartridge cost on to them. Since I hate things that fail, I will guarantee the conversion of each cartridge with a moneyback guarantee for 9 months from the date of purchase. What I cannot guarantee though is the life of the existing battery. For those who want it I will install Robert Prins' TI-95 Utilities Cartridge programs onto the 32K cartridge free of charge. It has not been tested but the 32K should also work with the TI-74. Write to Scott L. Garver, 1113 Woodlawn Ct., Pekin, IL 61554 if you are interested.

Editor's Note: Cartridge battery life and the Robert Prins Utility Cartridge programs are discussed elsewhere in this issue.

A NAVIGATION PACKAGE FOR THE TI-95 - Hewlett Ladd has been working on this package in collaboration with Al Mackenzie for many months. Members may recall that several navigation-related programs were in his list of available programs in V12N3P3. The navigation package includes five access menus:

1. Sight reductions for the sun moon, stars and planets.

2. Fixes: Two object fix, fix from the sun at Local Apparent Noon (Transit), longitude fix when the sun is on the prime verticle, and fix from the moon at meridian transit.

3. Times and azimuths of sunrise and sunset, moonrise and moonset. Times of morning and evening twilight. Three object fix. Running fix from unknown position.

4. Utilities including sextant corrections, horizontal parallax and semi-diameter for the moon, semidiameter for the sun, date and zone time to G-date and GMT, and star identification.

5. Dead reckoning aids for latitude and longitude including Mercator or rhumb-line computations, the same with speed and time, and great circle courses.

A main feature of the collection of programs is that the Nautical Almanac is not needed for problems dealing with the Sun or Moon. This feature is contained in three "Core Programs", subroutines which compute celestial coordinates and other elements of the Sun and Moon to a precision of Almanac accuracy, automatically, by direct calls from those navigation programs requiring such information as of a given date and instant of GMT.

The complete program uses 4712 bytes in user memory (MEM) and 8048 bytes in an 8K Memory Module (labeled XXX). The locations and the names are prescribed because of frequent inter-program calls. The 55 page documentation includes user instructions, comments on methods used, program descriptions, program listings, and a set of sample problems.

Hewlett will loan his copy of the documentation to members so that they can make a copy for their own use. Send four dollars. He will send his copy to you, and expect you to make a copy and return the original to him promptly. To avoid keying in the programs you can send two memory modules plus four dollars ($4.00). He will transfer the programs on your modules and return them to you. Write to Hewlett F. Ladd, 12 Ridgewood Avenue, Rowayton CT 06853.

SENIORNET - An article in the November 1989 issue of *DATA BUS* describes a network for senior citizens. The founder of the network is Dr. Mary Furlong at the University of San Francisco. She describes the network as an "intellectual amusement park" with information on health-care legislation, recreation, travel, stock quotations, news and a mail order service. The network has 2000 members who pay a $10.00 per year membership fee plus $12.60 per hour online. A night rate of $6.75 will go into effect in May. SeniorNet also maintains 25 locations where senior citizens can attend workshops on word processing, spreadsheets, etc. If you are interested and want more information write to SeniorNet, University of San Francisco, School of Education, San Francisco, CA 94117 or call 415-666-6505. Please mention our club if you write.

<u>A FASTER SMALLEST CIRCLE PROGRAM</u> - Carl Rabe. The problem of finding the smallest circle which will just enclose n random points was proposed by Don Laughery in V12N4P26. A workable solution was published in V13N2P12; however, for large numbers of points the program was painfully slow. Carl Rabe tackled the problem of reducing the solution time. His method involved identifying a subset of four points which are near the periphery of the set of points. A solution is found which will enclose the four points. That solutions is checked against the remaining points. Any points which fall outside the circle defined by the subset are added to the subset and the process is repeated. Don Laughery has noted that this process is similar to the one he uses graphically, that is, he ignores the points which are obviously not on the perimeter.

Carl reports the following improvement in solution times on the TI-74:

Pairs	V13N2P12	This Program
12	301 sec	9 to 44 sec
21	2940 sec	11 to 47 sec

```
SMALLEST CIRCLE
# 1 X= 9.8 Y= 3.9
# 2 X= 6.1 Y=-9.8
# 3 X= 11.3 Y= 2.1
# 4 X= 8.4 Y=-8.3
# 5 X=-9.8 Y= 7.5
# 6 X= 5.5 Y= 13.9
# 7 X= 5 Y= 8.7
# 8 X= 9.6 Y= 8.1
# 9 X= 16.5 Y= 1.1
# 10 X= 11.8 Y= 12.5
# 11 X=-9.8 Y=-7.2
 H = 1.84275743
 K = 1.725961397
 R = 14.67060283
```

where the variation in the solution time with this program depends upon the difficulty of solving a particular case. The shorter times are for cases which require a single cycle through the compute and check-for-outside of circle sequence. The longer times are for cases which require three cycles. A sample printout appears at the right.

Carl has noted that the program as written sometimes uses more cycles than would be expected due to the interaction of truncation errors in comparisons such as those at lines 675, 685, 725 and 745. We will work on that and try to publish a modification in the next issue. Meanwhile, the program on page 15 provides by far the fastest smallest circle solutions obtained to date.

Note: The constants in lines 725 and 745 must be selected to match the input data resolution.

<u>Program Description</u>:

Lines 10-290: This is the input section. It is similar to the input section of the program on V13N2P15. Lines 45-55 have been made inoperative and the device code number is set to that for the PC-324 in line 60. This may be changed to provide a choice of printers if that is desired. Additions provide a for optional recheck of inputs or a change of inputs if desired (PC-324 printer paper is at a premium due to the cost and the small amount of paper in each roll.)

Lines 295-440: This section sorts out four pairs which will fall near the periphery of the plot of all the input pairs. It places the four pairs in an array for use by the main program. If the number of input pairs is 4 or less then no sorting is done.

Lines 445-700. These lines are from V13N2P15 lines 200-760 with minor changes.

Lines 705-760. This section checks all points to see if any are outside the circle obtained from the reduced set of points. If any points are found outside the circle they are added to the reduced array and a new cycle is initiated. The process is continued until no points are found to be outside the computed circle, which is then the smallest circle.

Lines 765-815 provide for display and/or printout of the results.

Lines 820-865 are a subroutine used by the sorting process.

A Faster Smallest Circle Program - (cont)

Program Listing

```
10 A$="SMALLEST CIRCLE":
PRINT A$:PAUSE 1
15 !FROM PPC NOTES V13N2
P15
20 !MODIFIED 10/15/89 C.
RABE
25 DIM X(52),Y(52),U(52)
,V(52)
30 INPUT "USE PRINTER? Y
/N ";Z$
35 IF Z$="Y"OR Z$="y"THE
N PN=1 ELSE 80
40 !PRINT "PRINTER NUMBE
RS:":PAUSE 1
45 !PRINT "FOR THE HX-10
00 ENTER 10":PAUSE 1
50 !PRINT "FOR THE PC-32
4 ENTER 12":PAUSE 2
55 !INPUT "ENTER PRINTER
 NUMBER ";D$
60 D$="12"
65 OPEN #1,D$,OUTPUT
70 IF D$="10"THEN PRINT
#1,CHR$(18)
75 PRINT #1:PRINT #1,A$
80 INPUT "USE CANNED DAT
A? Y/N: ";ZZ$
85 IF ZZ$="Y"OR ZZ$="y"T
HEN 90 ELSE 155
90 PRINT "(DATA STATEMEN
T BEGINS LINE 125)":PAUS
E 1
95 FOR I=1 TO 50
100 READ X(I),Y(I)
105 IF X(I)=999 THEN 215
110 N=N+1
115 NEXT I
120 PRINT "NUMBER OF DAT
A PAIRS IS:";N:PAUSE 1
125 DATA 9.8,3.9,6.1,-9.
8,11.3,2.1
130 DATA 8.4,-8.3,-9.8,7
.5,5.5,13.9
135 DATA 5,8.7,9.6,8.1,1
6.5,1.1
140 DATA 11.8,12.5,-9.8,
-7.2
145 DATA 999,999
150 GOTO 215
155 PRINT "ENTER INPUT X
 & Y VALUES":PAUSE 1
160 PRINT "(TO END INPUT
 ENTER A NULL)"&CHR$(255
):PAUSE 1
165 N=1
170 X$="X = "
175 Y$="Y = "
180 INPUT X$:XX$:IF XX$=
""THEN 210
185 INPUT Y$;YY$:IF YY$=
""THEN 210
190 X(N)=VAL(XX$)
195 Y(N)=VAL(YY$)
200 IF PN=0 THEN 205
205 N=N+1:GOTO 180
210 N=N-1
215 PRINT "NUMBER OF DAT
A PAIRS IS:";N:PAUSE 2
220 INPUT "WISH TO RECHE
CK INPUT? Y/N:";ZZZ$
225 IF ZZZ$="Y"OR ZZZ$="
y"THEN 235 ELSE 275
230 PRINT "TO CHK INPUTS
 PRESS:ENTER  ":PAUSE
235 FOR I=1 TO N
240 PRINT "PAIR#";I;"  X
:";X(I);" Y:";Y(I):PAUSE
245 NEXT I
250 INPUT "CHANGE AN INP
UT? Y/N:";ZZ$
255 IF ZZ$="Y"OR ZZ$="y"
THEN 260 ELSE 220
260 INPUT "WHICH PAIR #?
: ";I
265 INPUT "ENTER NEW X:"
;X(I)
270 INPUT "ENTER NEW Y:"
;Y(I):GOTO 250
275 IF Z$="Y"OR Z$="y"TH
EN 280 ELSE 295
280 FOR I=1 TO N
285 PRINT #PN,"#";I;"X="
;X(I);"Y=";Y(I)!:PAUSE
290 NEXT I
295 PRINT "NOW SORTING..
PLEASE WAIT":PAUSE 1
300 DIM XX(21),YY(21),XX
X(21),YYY(21)
305 FOR I=1 TO N:U(I)=X(
I):V(I)=Y(I)
310 NEXT I
315 TOTAL_N=N
320 FAC=4
325 IF FAC>=N THEN FAC=N
:GOTO 430
330 PSN,I=1:LOWX=X(I):PT
R=0
335 FOR I=I TO N
340 IF X(I)<LOWX THEN LO
WX=X(I):PSN=I
345 NEXT I
350 CALL PLACE(X(),Y(),P
SN,PTR)
355 PSN,I=2:HIX=X(I):PTR
=1
360 FOR I=I TO N
365 IF X(I)>HIX THEN HIX
=X(I):PSN=I
370 NEXT I
375 CALL PLACE(X(),Y(),P
SN,PTR)
380 PSN,I=3:LOWY=Y(I):PT
R=2
385 FOR I=I TO N
390 IF Y(I)<LOWY THEN LO
WY=Y(I):PSN=I
395 NEXT I
400 CALL PLACE(X(),Y(),P
SN,PTR)
405 PSN,I=4:HIY=Y(I):PTR
=3
410 FOR I=I TO N
415 IF Y(I)>HIY THEN HIY
=Y(I):PSN=I
420 NEXT I
425 CALL PLACE(X(),Y(),P
SN,PTR)
430 FOR I=1 TO FAC
435 XX(I)=X(I):YY(I)=Y(I
)
440 NEXT I
445 PRINT "NOW SOLVING..
PLEASE WAIT":
450 CYCLES=CYCLES+1
455 FOR I=1 TO FAC
460 X(I)=XX(I):Y(I)=YY(I
)
465 NEXT I
470 N=FAC
475 M1=0
480 FOR I=2 TO N
485 FOR J=1 TO I-1
490 D2=(X(I)-X(J))*(X(I)
-X(J))+(Y(I)-Y(J))*(Y(I)
-Y(J))
495 IF D2>M1 THEN M1=D2:
S=I:T=J
500 NEXT J
505 NEXT I
510 HH=(X(S)+X(T))/2
515 KK=(Y(S)+Y(T))/2
520 M2=M1/4
525 Z=0
530 FOR I=1 TO N
535 IF I=S OR I=T THEN 5
50
540 D2=(X(I)-HH)*(X(I)-H
H)+(Y(I)-KK)*(Y(I)-KK)
545 IF D2>M2 THEN M2=D2:
Z=1
550 NEXT I
555 IF Z=0 THEN 765
560 FOR I=N TO 3 STEP -1
565 FOR J=(I-1)TO 2 STEP
 -1
570 FOR L=(J-1)TO 1 STEP
 -1
575 A=X(I):B=Y(I)
580 C=X(J):D=Y(J)
585 E=X(L):F=Y(L)
590 A=A-E:C=C-E
595 B=B-F:D=D-F
600 G=C*B-A*D:G=G+G
605 IF G=0 THEN 690
610 H1=C*C+D*D
615 X1=B:B=H1
620 B=X1*B:H1=A*H1
625 X1=X1*X1+A*A
630 D=X1*D:C=X1*C
635 B=B-D:C=C-H1
640 B=B/G:C=C/G
645 E=E+B:F=F+C
650 R2=B*B+C*C
655 H=E:K=F
660 FOR M=1 TO N
665 IF M=L OR M=J OR M=I
 THEN 680
670 D2=(X(M)-H)*(X(M)-H)
+(Y(M)-K)*(Y(M)-K)
675 IF D2>R2 THEN R2=D2
680 NEXT M
685 IF R2<M2 THEN M2=R2:
HH=H:KK=K
690 NEXT L
695 NEXT J
700 NEXT I
705 N=TOTAL_N
710 FOR I=1 TO N
715 X(I)=U(I):Y(I)=V(I)
720 NEXT I
725 M2=INT(M2*1.E+07+.5)
/1.E+07
730 CNT=0
735 FOR I=1 TO N
740 D2=(X(I)-HH)*(X(I)-H
H)+(Y(I)-KK)*(Y(I)-KK)
745 D2=INT(D2*1.E+07+.5)
/1.E+07
750 IF D2>M2 THEN CNT=CN
T+1:FAC=FAC+1:XX(FAC)=X(
I):YY(FAC)=Y(I)
755 NEXT I
760 IF CNT<>0 THEN 450
765 PRINT "SOLUTION COMP
LETED. ":PAUSE 2
770 PRINT "NUMBER OF CYC
LES: ";CYCLES:PAUSE
775 PRINT "PAIRS FOR FIN
AL COMPUTE:";FAC:PAUSE
780 PRINT "CIRCLE DATA F
OLLOWS":PAUSE 2
785 PAUSE ALL
790 PRINT #PN," H = ";HH
795 PRINT #PN," K = ";KK
800 PRINT #PN," R = ";SQ
R(M2)
805 IF PN=1 THEN PRINT #
1
810 PAUSE 0
815 PRINT "PROBLEM COMPL
ETED":PAUSE 1
820 SUB PLACE(X(),Y(),PS
N,PTR)
825 PTR=PTR+1
830 TMPX=X(PTR)
835 TMPY=Y(PTR)
840 X(PTR)=X(PSN)
845 Y(PTR)=Y(PSN)
850 X(PSN)=TMPX
855 Y(PSN)=TMPY
860 SUBEND
865 END
```

Editor's Comments: Carl implemented some other features in his program which are worthy of mention:

1. The definition of lines 40 through 55 as comments saves the user some time if he is only using the PC-324. Note that line 60 sets the device code number to that for the PC-324. If the option to select the printer from the keyboard is to be implemented the exclamation points in lines 40 through 55 must be removed and line 60 must be deleted or changed to comment status by the insertion of an exclamation point.

A Faster Smallest Circle Program - (cont)

2. Lines 80 through 145 provide for entry of data pairs from DATA statements rather than from the keyboard. Lines 125 through 140 provide a sample problem which can be used to check operation of the program. Line 105 operating with the 999's in the DATA statement in line 145 provide for termination of the input routine. The printout on page 14 was obtained using the problem in the data statements. Other methods of accomplishing the same effect are discussed elsewhere in this issue.

3. Line 315 illustrates the use of an underline in a variable name. This was previously demonstrated in Stephen Gutknecht's arrow game program in V13N3P12.

4. The first statement in line 330 illustrates a capability which is unique to TI BASIC, where

PSN,I=1 in TI BASIC is equivalent to the statements PSN=1:I=1

which must be used in most other BASIC implementations. The technique is described in the discussion of the LET keyword on page 2-63 of the *TI-74 Programming Reference Guide*. Similar statements appear in lines 380 and 405.

5. Several statements in this program illustrate a feature of the IF THEN ELSE statement that is not widely used, namely that actions can include groups of statements separated by colons. Thus, for line 495:

```
495 IF D2>M1 THEN M1=D2:S=I:T=J
```

if D2>M1 then M1, S and T are set to new values. If D2<=M1 no actions are taken and control passes directly to line 500. Several examples illustrating the effect are discussed on page 2-46 of the *TI-74 Programming Reference Guide*.

ERROR CODE DIFFERENCES BETWEEN THE CC-40 AND TI-74 - P. Hanson. While trying to demonstrate an alternate method for determining the end of the items in DATA statements I tried to translate a routine which had previously been used successfully with the CC-40. I found that my program wouldn't work when directly translated. I eventually traced the problem to a difference in the error code assigned for the error message "DATA error" for the two machines. I proceeded to compare the lists of error codes on pages K-11 and K-12 of the *Compact Computer 40 User's Guide* and page A-27 of the *TI-74 Programming Reference Guide* and found additional differences in error code assignments. I then compared the detailed descriptions of the error codes in pages K-2 through K-11 of the *Compact Computer 40 User's Guide* and pages A-16 through A-26 of the *TI-74 Programming Reference Guide*. The diagram on page 17 presents the results of my work. Note that:

* Some error codes and error messages are the same; for example, "Bad Value" (code 04) and "Stack Underflow" (code 05).

* Some error codes are different even though the error codes and the detailed descriptions are the same; for example, "Division by zero" which is code 34 with the CC-40 but code 26 with the TI-74.

* In some cases, several error codes and error messages from the CC-40 have been combined into a single error code and message on the TI-74; for example, the "FOR/NEXT error" (code 6) on the TI-74 is the composite of three separate errors on the CC-40, "NEXT without FOR" (code 06), "Illegal FOR/NEXT nesting" (code 13) and "FOR without NEXT" (code 30). The obvious result of such combination is that the error indications from the TI-74 will not be as definitive as from the CC-40.

Error Code Differences between the CC-40 and TI-74 - (cont)

CORRESPONDENCE BETWEEN CC-40 AND TI-74 ERROR CODES

CC-40 Message	CC-40 Code	TI-74 Code	TI-74 Message
I/O error	00	0	I/O error
Illegal syntax	01	1	Syntax
Expression too complex	02	2	Complex
String-number mismatch	03	3	Mismatch
Bad value	04	4	Bad value
Stack underflow	05	5	Stack underflow
NEXT without FOR	06	6	FOR/NEXT error
Bad INPUT data	07	7	Bad data
Invalid dimension	08	8	Bad dimension
Variable previously defined	09	9	Previously defined
Can't do that	10	10	Can't do that
Illegal after SUBEND	11	11	Line number error
Line reference out of range	12	12	Missing statement
Illegal FOR/NEXT nesting	13	13	Not found
Missing RETURN from error	14	14	Bad program type
Program not found	15	15	Protection error
Line not found	16	16	In use
Bad line number	17	17	Not defined
Bad program type	18	18	Image error
Illegal in program	19	19	File error
Protection violation	20	20	Name Table Full
Subprogram in use	21	21	Parenthesis
Variable not defined	22	22	Too long
Error in image	23	23	Bad Argument
File error	24	24	Extension missing
Name table full	25	25	Overflow
Unmatched parenthesis	26	26	Division by zero
Line too long	27	27	Contents may be lost
Name too long	28	28	Truncation
Bad argument	29	29	Break
FOR without NEXT	30	30	Initialized
BASIC extension missing	31	31	No RAM
Bad subscript	32	32	DATA error
Overflow	33		
Division by zero	34		
Memory contents may be lost	35		
String truncation	36		
Break	37		
System initialized	38		
Must be in subprogram	39		
No RAM in catridge	40		
Statement must be first on line	41		
Missing SUBEND	42		
Data Error	43		
Must be in program	44		
RETURN without GOSUB	45		
System error	126	126	System error
Memory full	127	127	Memory full

The I/O Error Codes are the same for the two machines with the exception that the ILLEGAL IN SLAVE MODE (code 254) is not listed for the TI-74. See pages K-13 through K-16 of the *Compact Computer 40 User's Guide* and pages A-28 through A-31 of the *TI-74 Programming Reference Guide*.

THE USE OF DATA STATEMENTS TO STORE INPUT DATA

Carl Rabe included a routine for reading input data from DATA statements in his latest smallest circle program. This technique is valuable if user is planning to process the input data many times, or as a means to provide a "canned" test problem with a program. The first program at the right contains essentially the same routine. The routine senses the end of data by recognizing the 999's in line 520.

```
100 DIM X(50),Y(50)
105 FOR I=1 TO 50
110 READ X(I),Y(I)
115 IF X(I)=999 THEN 200
120 N=N+1
125 NEXT I
200 PRINT "Number of Dat
a Pairs is:";N:PAUSE 2
205 END
500 DATA 9.8,3.9,6.1,-9.
8,11.3,2.1
505 DATA 8.4,-8.3,-9.8,7
.5,5.5,13.9
510 DATA 5,8.7,99.6,8.1,
16.5,1.1
515 DATA 11.8,12.5,-9.8,
-7.2
520 DATA 999,999
```

Another way of accomplishing the same thing is with an error processing routine such as that used to provide a test case in the CC-40 sorting program on V10N3P14. The second program at the right is similar to the routine in the sorting program. The program uses error code 32 (DATA Error) to sense the end of DATA statements. It was during the of conversion of the routine from the CC-40 to the TI-74 that I discovered the differences between error codes described in more detail on pages 16 and 17. The idea for this method originally came from an article in *Codeworks*.

```
100 DIM X(50),Y(50)
105 ON ERROR 150
110 READ X(N+1),Y(N+1)
115 N=N+1
120 GOTO 110
150 CALL ERR(E,F)
155 IF E=32 THEN 200
160 RETURN 900
200 PRINT "Number of Dat
a Pairs is:";N:PAUSE 2
205 END
500 DATA 9.8,3.9,6.1,-9.
8,11.3,2.1
505 DATA 8.4,-8.3,-9.8,7
.5,5.5,13.9
510 DATA 5,8.7,99.6,8.1,
16.5,1.1
515 DATA 11.8,12.5,-9.8,
-7.2
900 PRINT "Error Code =
";E:PAUSE
910 END
```

While working on the review of the error codes I recognized a fundamental deficiency in this method. It assumes that error code 32 is the result of the "Out of data" condition, but page A-18 of the *TI-74 Programming Reference Guide* shows that other problems in DATA statements can generate a code 32. If you change the DATA statement in line 510 by inserting an alphabetic character after the first 5, say such that the statement reads "DATA 5A,8.7,99.6, ... " then code 32 will be generated, the input from DATA statements will terminate, and the routine will indicate that only six of the eleven data pairs will be read. Thus, the termination of input from DATA statements by use of the error processing routine can result in the omission of desired data if there are faults in the format of the data. This would seem to allow the use of the method for "canned data" for test problems; but, since there are better methods for reading input data from DATA statements there doesn't seem to be any good reason to use the error processing technique.

```
100 DIM X(50),Y(50)
105 FOR I=1 TO 50
110 READ X$,Y$
115 IF X$=""OR Y$=""THEN
 200
120 X(I)=VAL(X$):Y(I)=VA
L(Y$):N=N+1
125 NEXT I
200 PRINT "Number of Dat
a Pairs is:";N:PAUSE 2
205 END
500 DATA 9.8,3.9,6.1,-9.
8,11.3,2.1
505 DATA 8.4,-8.3,-9.8,7
.5,5.5,13.9
510 DATA 5,8.7,99.6,8.1,
16.5,1.1
515 DATA 11.8,12.5,-9.8,
-7.2
520 DATA "",""
```

A third way of terminating the input from DATA statements is to read the data as string expressions, convert to numeric variables using the VAL function, and sense the end of the data with null strings in the DATA statements. A similar technique was used to display the directory for the"Menu and Module" progrqam on V13N2P21. The third program at the right presents a conversion of that routine for the input of numerics. The advantage over Carl's method (the first program at the right) is that the user does not have to check the input for values which are equal to the termination value; for example, in Carl's routine a X value of 999 would cause the input to omit any subsequent data.

The bottom routine at the right adds the number of data pairs as the first value in the DATA statements (see the 11 as the first value in line 500). The single READ statement at line 105 finds that value, and assigns it to the variable N used to terminate the input of data via the following FOR-NEXT loop. The user instructions must emphasize that the first item in the data list is the number of input pairs.

```
100 DIM X(50),Y(50)
105 READ N
110 FOR I=1 TO N
115 READ X(I),Y(I)
120 NEXT I
200 PRINT "Number of Dat
a Pairs is:";N:PAUSE 2
205 END
500 DATA 11,9.8,3.9,6.1,
-9.8,11.3,2.1
505 DATA 8.4,-8.3,-9.8,7
.5,5.5,13.9
510 DATA 5,8.7,99.6,8.1,
16.5,1.1
515 DATA 11.8,12.5,-9.8,
-7.2
```

HOW DO BANKS CALCULATE INTEREST - Charlie Williamson writes: A national bank pays interest on checking accounts. About half the statements differ from my figures by a penny or more and it's always in the bank's favor. Nobody seems to know how balances are figured. They all say the computer does it and give different versions of how it does it.

A disclosure statement says that interest is calculated and compounded daily by applying an interest factor (the annual rate divided by 365, or 366 in a leap year) to the "available balance" each day. Here is an example for a 31 day period:

Previous balance	$ 1148.42
Interest at 4% rate	3.90
New balance	$ 1152.32

If we calculate the new balance using the classic calculation we obtain:

$1148.42 x (1 + 0.04/365)^31 = $1152.3279

which rounds $1152.33 but truncates to $1152.32. If banks really truncate in such cases then what kind of income can they realize? Consider a bank with 1000 accounts and monthly calculations of interest. Since the average truncation will be a half a cent the bank will realize sixty dollars per year.

Does anyone know what banks really do?

USING A TI-95 MODULE PROGRAM AS A SUBROUTINE - Robert Boykin of Midland, Texas called my attention to this two page TI-generated discussion of how to access one TI-95 program as a subroutine for another TI-59 program. Much of the material parallels that by William Hawes for the Math module in V12N1P17. The TI discussion involves the Statistics module, so if you are working with that module you might find the TI discussion to be helpful.

We had hoped to reprint the discussion in this issue, but have not yet obtained the required permission. So, if you want it write to Texas Instruments, Consumer Service, Box 53, Lubbock, TX 79408 and ask for PR001-1187.

A BOOK OF TI-59 PROGRAMS - In late 1988 Victor Chapman of Houston, Texas wrote that a Half Price bookstore had a book that may interest a member who is still using the TI-59. The book *Calculator Programs for the HYdrocarbon Processing Industries, Volume 2*, by S. Jagannath, includes 416 pages. The price was $19.96, half of the original retail price. Victor's letter was misplaced, so the likelihood that the book is still available seems small. If you are interested you might write to Half Price, 2537 University, Houston TX 77005.

GAME PROGRAMS FOR THE TI-74 - Steve Greenspon of Montreal, Quebec sent in a tape with several TI-74 game programs. The games include blackjack, crazy eights, hangman, and monopoly, where the monopoly program requires an 8K RAM. The tape also includes a matrix manipulation program which requires the additional RAM. If you would like to obtain these programs send one dollar (no checkws, please) to cover postage and packaging. I will send Steve's tape to you. You can copy the program as you see fit and promptly return the tape to me for use by other members.

<u>STORAGE OF DATA IN THE TI-74</u> - Jim Nugent noted that the TI-74 implementation of the RAM cartridge leaves much to be desired. It would have been nice to have multiple program files and data files in RAM, similar to that in the TI-95 or in the Radio Shack Model 100. That would have allowed a user to go out into the field and collect data with only the hand-held computer, and then come in to the office and print it out, save it on tape or transfer it to a PC.

<u>Editor's Note</u>: Other users have echoed Jim's complaint about the inability to store data files in the TI-74's memory or in an installed RAM. That issue also affects the projected use of a TI-74 with a PC using the interface cable.

One solution is to "bite the bullet" and violate TI's instructions for use of the PC-324 Printer or the CI-7 Cassette Interface with the TI-74. You simply leave the calculator running in a BASIC program at the end of the field site note taking. You have to be sure to leave the computer in a mode where the automatic power down can not come in, or you can set up the block of the automatic power down function as described on page 1-7 of the *TI-74 User's Guide*. Then when you get to where a peripheral is located, you simply hook up the printer and continue on. You can do this by leaving the TI-74 in the BREAK mode (remember not to clear the display after the break or you will be subject to automatic shutdown), and then doing a Continue (CON) to proceed after you have made the connection. You can open the port to the printer after you have made the connection with an entry from the keyboard. Of course, you would want to make a substantial number of "dry runs" with this technique before you tried to use it in a case where it was critical to retain the data. You also have to recognize that you could be susceptible to loss of warranty if you use this procedure since instructions on pages 8 and 9 of the *PC-324 User's Guide* and on the instruction card for the CI-7 are very specific that the first step in connecting with the peripheral is to ensure that the calculator is off.

One way to store data which will survive exit from BASIC or shutdown is to put the data in DATA statements. The consideration of that option led to the examination of the various methods of retrieving data from DATA statements on page 18 of this issue.

If you have a limited number of data items to store through a shutdown you can use the "Assigning Text to a Key" idea from pages 3-22 and 3-23 of the *TI-74 User's Guide*. What you can do is bring the information to the display followed by a command which permits data entry such as an INPUT statement. Then make keyboard entries to match the displayed data and press SHIFT FN and one of the ten number keys. After turning off and back on again, and again setting up with an INPUT statement, you can retrieve the information stored in the SHIFT FN definitions by pressing FN and the appropriate number key. A bit cumbersome, perhaps, but it works.

Finally, there may be hope for a better solution. A contact at TI tells me that a program has been written which permits the establishment of data files in the TI-74 memory. So far, I haven't been able to obtain the program.

<u>THE PC TO TI-74 INTERFACE CABLE ALSO WORKS WITH THE TI-95</u> - V13N3P10 reported the coming availability of a cable which will provide the interface between a TI-74 and a TI or IBM compatible PC. I received an engineering model from TI and immediately found that the cable will also provide the interface between a TI-95 and a PC. Scott Garver currently has the cable. He promises to provide an evaluation for the first issue in the coming year.

STATUS REPORT ON THE ROBERT PRINS UTILITIES CARTRIDGE (UC) FOR THE TI-95

Old-timers will remember that a "club utility module" for the TI-59 was proposed in V5N9-10P2, and a preliminary set of routines was presented in V6N4-5P2. The module was never completed. One reason might have been that there were "too many cooks". Shortly after the TI-95 became available Robert Prins proposed that a utilities module might be a good idea. With minimal support and input from others he selected a set of appropriate routines and has completed the programming.

The resulting TI-95 UC is a collection of routines that perform useful tasks. Because all of the routines are written in assembly language they are very fast. Because the routines are in assembly language special procedures are needed to install the programs in a TI-95. Scott Garver is working with Robert to complete the documentation.

Some representative functions which appear in the UC are:

ON/OFF - These functions, used in conjunction with each other, provide the TI-95 with a true constant memory by bypassing the built-in routines which clear the T register, clear the AOS register, clear the SBR return stack, etc. The result is that a TI-95 will "wake up" in EXACTLY the same state as when it was turned off using the OFF function.

CHR - This function allows easy modification of CHR 000...004 with the result of the modification shown directly in the system alpha register. This capability is similar to that provided by the CHAR subprogram of the CC-40.

CRC - This function calculates a two byte checksum for the program currently in program memory. The checksum can be used to determine if a program has been keyed in correctly if the author has provided a CRC checksum with his program.

SGN - This function is a true -1 / 0 / +1 signum function.

CJ/JC - These functions allow conversion of a date to a Julian Day Number or a JDN back to a date. Provisions are included to work with both US (MMDD.YYYY) and European (DDMM.YYYY) formats. The user can also select between the Julian and Gregorian calendars.

SHL - This function provides a very fast sorting capability.

REN - This function can be used to rename files in MEM.

MRX - This function can be used as an alternative to MRG. Its advantage is that it allows merging in any fix mode without altering the current display mode.

MRF - This function can be used to merge a file from MEM to the end of the program currently in memory.

ROM - This function can be used to download the system ROMs into RAM, in chunks of 4 Kb. It is a copy of the program from V12N2P26 of TI PPC Notes.

ENT - This function is a mathematical integer function; e.g., -1.5 ENT gives -2.

COLUMNAR DATA OUTPUT WITH THE TI-95 AND PC-324 - Don Laughery observes that printing columnar data outputs is easily done with the TI-74 and PC-324 using the PRINT USING and IMAGE statements. It is not so easy with the TI-95. Readers are invited to submit routines which will provide printing of columns with the TI-95 and PC-324.

REGRESSION WITH 2 OR 3 VARIABLES AND USER DEFINED FUNCTIONS - P. Hanson

This program is another iteration in the development of a TI-74 regression program which provides versatility through the use of user defined functions for the regression. Earlier versions in the development appeared in V11N4P12-15, V12N1P14, V12N3P19 and V13N3P18/19. The impetus for this iteration was support of the discussion of curve fitting with George Thomson and Marcel Bogart as documented on page 25 of this issue. Changes relative to the program in V13N3P18/19 provide:

* User selection of two or three variables (one or two independent variables).
* The capability to run the solution unweighted, or to apply a user defined weighting as part of the user defined functions.
* The capability to use one function for the solution and another for the calculation of residuals.

Program Listing:

```
10 REM Linear Regression with 2 or 3 Variables
20 REM and User Defined Functions
30 REM 4 September 1989
100 DIM A(8,8),B(8),F(8),X(50),Y(50),Z(50)
105 INPUT "Use Printer <Y/N>? ";A$
110 IF A$="Y"OR A$="y"THEN PN=1 ELSE 123
115 INPUT "Device Code ? ";P$
120 OPEN #1,P$,OUTPUT
123 INPUT "Are functions correct <Y/N>? ";A$
127 IF A$="N"OR A$="n"THEN STOP
130 INPUT "Number of Data Points? ";K
133 INPUT "2 or 3 Variables? ";KK
137 IF KK=2 OR KK=3 THEN 140 ELSE 133
140 FOR I=1 TO K
150 A$="X"&STR$(I)&" = ":INPUT A$;X(I)
160 IF PN<>0 THEN PRINT #PN,A$;X(I)
170 A$="Y"&STR$(I)&" = ":INPUT A$;Y(I)
175 IF PN<>0 THEN PRINT #PN,A$;Y(I)
180 IF KK=2 THEN 185
181 A$="Z"&STR$(I)&" = ":INPUT A$;Z(I)
183 IF PN<>0 THEN PRINT #PN,A$;Z(I)
185 PRINT #PN:IF E$<>""THEN 700
190 NEXT I
200 INPUT "Order of the solution? ";N
205 W=1
210 FOR I=1 TO N:FOR J=1 TO N
220 A(I,J)=0:NEXT J
230 B(I)=0:F(I)=0:NEXT I
240 FOR L=1 TO K
250 GOSUB 800
300 FOR I=1 TO N:FOR J=1 TO N
305 A(I,J)=A(I,J)+F(I)*F(J)*W:NEXT J
310 B(I)=B(I)+F(I)*F(0)*W:NEXT I
315 NEXT L
320 FOR L=1 TO N
325 P=A(L,L)
330 FOR J=L TO N
335 A(L,J)=A(L,J)/P:NEXT J
340 B(L)=B(L)/P
345 FOR I=1 TO N
350 IF I=L THEN 375
355 G=A(I,L)
360 FOR J=L TO N
365 A(I,J)=A(I,J)-G*A(L,J):NEXT J
370 B(I)=B(I)-G*B(L)
375 NEXT I
380 NEXT L
400 FOR I=1 TO N
410 X$="A"&STR$(I)&" = "
420 PRINT #PN,X$;B(I)
430 IF PN=0 THEN PAUSE
440 NEXT I
450 PRINT #PN
500 INPUT "Display Residuals <Y/N>? ";A$
510 S1=0:S2=0
520 FOR L=1 TO K
530 GOSUB 900
570 IF A$="N"OR A$="n"THEN 610
580 P$="d"&STR$(L)&" = "
590 PRINT #PN,P$;D
600 IF PN=0 THEN PAUSE
610 S1=S1+D:S2=S2+D*D:NEXT L
620 PRINT #PN
630 PRINT #PN,"Mean = ";S1/K
640 IF PN=0 THEN PAUSE
650 PRINT #PN
660 PRINT #PN,"S.E. = ";SQR(S2/(K-N))
670 IF PN=0 THEN PAUSE
680 PRINT #PN
700 INPUT "Edit Input Data <Y/N>? ";E$
710 IF E$="N"OR E$="n"THEN 780
720 INPUT "Which Data Pair to Edit? ";I
730 IF I<1 OR I>K THEN 700
740 GOTO 150
780 INPUT "New Solution <Y/N>? ";A$
790 IF A$="Y"OR A$="y"THEN 200
799 STOP
800 REM USER DEFINED FUNCTIONS
810 F(0)=LN(Y(L))
820 F(1)=1
830 F(2)=LN(X(L))
840 F(3)=LN(Z(L))
850 W=Y(L)*Y(L)
860 RETURN
900 REM RESIDUAL CALCULATIONS
910 YF=EXP(B(1))*X(L)^B(2)*Z(L)^B(3)
920 D=Y(L)-YF
930 RETURN
```

Regression with 2 or 3 Variables and User Defined Functions - (cont)

The changes in the program relative to the listing on V13N3P19 are:

The Else address in line 110 is changed to agree with other changes in the program.

Line 125 is replaced by lines 123 and 127 to provide for an operator input to indicate that the functions are correct.

Lines 133, 137 and 180 are added to provide for the input of 2 or 3 variables.

Line 205 is added to provide a default value for uniform weighting.

Lines 305 and 310 are changed to include the weighting factor W.

Lines 530 to 560 are replaced by a new line 530 to provide for use of a different function for the calculation of residuals.

Lines 900-930 are added to provide the functions for the calculation of residuals.

Sample User Defined Functions and Residual Calculations

Lines 800 to 860 and 900 to 930 provide for a solution for a double power function evaluation similar to that in the program on V13N3P18/19. The name was changed from "double exponential to "double power" for consistency with other solutions in TI PPC Notes. The difference in calculations is that the residuals are in the coordinate system before the logarithmic transformation.

Two additional listings for lines 800 ff appear below. The left hand listing can be used for a single exponential solution with the residuals in the coordinate system before transformation. The right hand listing can be used for solution of polynomials with one independent variable in a manner similar to that in the program in V11N4P12-15.

```
800 REM USER DEFINED FUN
CTIONS
810 F(0)=LN(Y(L))
820 F(1)=1
830 F(2)=X(L)
840 W=Y(L)*Y(L)
850 RETURN
900 REM RESIDUAL CALCULA
TIONS
910 YF=EXP(B(1))*EXP(B(2
)*X(L))
920 D=Y(L)-YF
930 RETURN
```

```
800 REM USER DEFINED FUN
CTIONS
810 F(0)=Y(L)
820 F(1)=1
830 FOR NN=2 TO N
840 F(NN)=F(NN-1)*X(L)
850 NEXT NN
860 RETURN
900 REM RESIDUAL CALCULA
TIONS
910 GOSUB 800
920 YF=0:FOR NN=1 TO N
930 YF=YF+B(NN)*F(NN):NE
XT NN
940 D=Y(L)-YF
950 RETURN
```

The important things to remember when generating other user defined functions and residual calculations are that F(0) in the user defined function section is the independent variable (with or without transformation), W in the user defined function section is the weighting to be used (a default of W = 1 is provided elsewhere), and the equations for the residual calculations which follow line 900 must find the variable D as the difference between the dependent variable (or a transformation thereof) and the calculated value.

Regression with 2 or 3 Variables and User Defined Functions - (cont)

The pair of printouts at the left below are the solutions with and without y-squared weighting for the same double power function problem as on V13N3P18. For the unweighted solution the coefficients are the same as in V13N3P18, but the residuals are different since they are calculated in the coordinate system before transformation. Thus, the mean of the residuals and the standard error are residuals are larger. For the weighted solution, both the mean error and the standard error are reduced relative to the unweighted solution.

The pair of printouts at the right below are the solutions with and without y-squared weighting for an exponental fit to the same data used for the demonstration of the five function curve fit in V12N4P24/25. In this case the first coefficient of the solution is actually ln(a).

W = 1

```
X1 =  21
Y1 =  .201064
Z1 =  2

X2 =  40
Y2 =  1.43996
Z2 =  7

X3 =  100
Y3 =  4.34868
Z3 =  5

X4 =  500
Y4 =  32.3097
Z4 =  3

X5 =  50
Y5 =  2.67693
Z5 =  10

A1 = -6.725432649
A2 =  1.499999478
A3 =  .8000012245

d1 = -4.88015E-09
d2 =  3.28122E-07
d3 = -7.60033E-07
d4 =  2.117956E-06
d5 = -2.52659E-07

Mean =  2.857012E-07

S.E. =  1.617855E-06
```

$W = Y^2$

```
X1 =  21
Y1 =  .201064
Z1 =  2

X2 =  40
Y2 =  1.43996
Z2 =  7

X3 =  100
Y3 =  4.34868
Z3 =  5

X4 =  500
Y4 =  32.3097
Z4 =  3

X5 =  50
Y5 =  2.67693
Z5 =  10

A1 = -6.725433786
A2 =  1.499999639
A3 =  .8000014045

d1 =  9.969297E-08
d2 =  6.02523E-07
d3 = -3.11868E-07
d4 =  1.5396E-08
d5 = -1.0859E-08

Mean =  7.897699E-08

S.E. =  4.850718E-07
```

W = 1

```
X1 =  1
Y1 =  3.2

X2 =  2
Y2 =  7.4

X3 =  3
Y3 =  12

X4 =  4
Y4 =  16.8

X5 =  5
Y5 =  22

A1 =  .9096871079
A2 =  .4675682173

d1 = -.7640067736
d2 =  1.073016751
d3 =  1.901450901
d4 =  .6816241667
d5 = -3.726669937

Mean = -.1669169783

S.E. =  2.562763297
```

$W = Y^2$

```
X1 =  1
Y1 =  3.2

X2 =  2
Y2 =  7.4

X3 =  3
Y3 =  12

X4 =  4
Y4 =  16.8

X5 =  5
Y5 =  22

A1 =  1.364904852
A2 =  .3513039132

d1 = -2.36339629
d2 = -.5051360299
d3 =  .7674412557
d4 =  .8394427791
d5 = -.6786605439

Mean = -.3880617659

S.E. =  1.591122751
```

CAN YOU USE A TI-99/4? - A set of used TI-99/4 has been donated to our club. It does have documentation, a TV interface and a cassette interface but does not have a disc drive. Write for details if you are interested.

A 32K RAM FOR THE TI-74 FROM TI - Page 12 of this issue describes a 32K RAM capability for the TI-95 as developed by Scott Garver. A recent brochure from TI on TI-74 applications indicated that a 32K CRAM Module is available for the TI-74. I called TI for information and was told that the module had been delivered for use in some special applications but was not yet available to the public. For most TI-74's in circulation it will be necessary to load a special MEMADD subprogram prior to use of the 32K Module. TI will send an engineering model for evaluation. I will report on it in the first issue of the coming year.

Other literature received from TI indicates that they been succcessful in marketing sizeable numbers of TI-74's for use special field applications with activities such as the Farmers Home Administration, Cummins Engine Company, Kodak and the Canadian Department of Revenue. Those applications should lend assurance to the continued availability of the TI-74/TI-94 peripherals such as the PC-324 and CI-7.

STORAGE LIFE FOR THE 8K RAM CARTRIDGE - V13N3P22 presented Scott Garver's review of the documentation on the life of the battery in the 8K RAM cartridge, and noted that there were some unanswered questions. We have obtained some answers from TI:

1. When the RAM is installed in TI-74 or TI-95 the power to support the RAM comes from the batteries in the parent machine. Thus, RAM storage life will be extended if the RAM is in a calculator or computer.

2. The battery in the RAM is not recharged by the calculator or computer.

Manufacturers of carbon-zinc and alkaline cells provide a "Best If Installed by ..." note on the package to alert users to shelf life for the cells. There are also well established degradation modes for lead-acid and nickel-cadmium cells. So another question on the RAM battery is:

Doesn't the RAM battery age and loses capability gradually even when the RAM is installed in a calculator or computer?

THE PARALLEL PORT OPTION FOR THE HX-3000 - The discussion of interfacing with peripherals in V13N3P23 asked if anyone had modified an HX-3000 to implement the parallel port option. Jim Nugent of Peoria, Illinois wrote: "I opened up my HX-3000 back in 1983 and installed a short (12 inch) cable that ran out of the case over or under the RS-232 connector. When I wanted to print I took the cable off the back of my IBM PC and connected it to the cable stub hanging out of the back of the HX-3000.

To add a parallel port to an HX-3000 you need a 20 contact ribbon header socket (IDM20 from JDR - see page 327 in the August 1989 issue of BYTE), a foot or two of 20 conductor ribbon cable, and a male D-subminiature 25 pin connector. I hand wired my D-sub connector, so I must have used the circuit diagram in the HX-3000 data sheet that came with the unit. As I remember it, the 20 pins on the HX-3000 were half grounds. All of the pins on the bottom row, I think, were grounds giving you a ribbon cable with every other wire a ground. My cable out the back was crude and not very portable. A better soloution would be to bite the bullet and mount the D-sub connector on the side or top of the HX-3000 case."

Jim also addressed the other interface issue raised in V13N3P23. He tried to use the CI-7 with the CC-40 without success (the same experience as the editor). He agreed that the CC-40 had some great features which were not carried forward to the TI-74 such as the CHAR subprogram, the DEBUG Monitor subprogram, and a better RAM file system.

MORE THOUGHTS ABOUT CURVE FITTING - George Thomson's treatise on curve fitting in V8N1P27 observed that that the use of the logarithmic transformation to enable an easy least squares solution for an equation of the form y= ax^b does not minimize error in y but rather minimizes proportionate or percentage error in y. He noted that the weighting of each transformed data pair by y^2 would provide a solution which would minimize error in y. The publication of Marcel Bogart's "Double Exponential Evaluation of Data" in V13N3P15 caused George to write to re-emphasize the need to use caution when using transformations. George, Marcel and the editor then embarked on some extensive correspondence relative to when weighting should or should not be used. Some results of the correspondence are the corrections on page 2 of this issue, the revised TI-59 program for solution of a power function on pages 6 and 7 of this issue, and another iteration in the development of a least squares solution with user defined functions on pages 22 to 24 of this issue.

When should we use weighting, and when should we not? Elememtary discussions of least squares fitting assume that the observations are all equally accurate, as they might be when we count something, weigh something, etc. But, in many electrical instrumentation applications the measurement accuracy is a function of the magnitude of the measurement. An example occurs when the user switches the range of a voltmeter, ammeter, or ohmmeter. In such cases, the fitting routine should weight the observation depending on the accuracy of the observation. For electrical measurements that is often exactly the proportional weighting that is obtained by using the logarithmic transformation. Marcel Bogart says that one clue is if the data is stated in terms of a qualifier such as parts per million. So the answer is that the decision on whether to weight the data or not depends on the nature of the data itself, which often depends on the physics of the process or the physics of the instrumentation. George Thomson said essentially the same thing in V8N1P28:

> "The use of a transformation ... (such as a logarithmic transformation for a power function Y = aX^b) ... no longer minimizes the error in Y when the method of least squares is used. Instead it minimizes errors in the logarithm of Y, that is, proportionate or percentage errors in Y. For many problems, this may be the right thing to do, but the user should be aware that the standard process does not minimize absolute errors in Y."

The discussion of weighting in Chapter 10 of Deming's *Statistical Adjustment of Data* says much the same thing:

> "... Of course, in some lines of work, the weight of y is approximately inversely proportional to y^2, whence the weight of log y is practically constant, independent of y. When this is so, the weighting factor y^2 is to be omitted."

George has also noted that the selection of Y-squared weighting (with or without normalization as discussed on page 2 of this issue) only depends on the use of the ln(Y) function on the right hand side of the equation. This means that similar weighted solutions can be obtained for equations such as

$$Y = Ae^{Bx} \quad , \qquad Y = Ae^{Bx^2} \qquad \text{and} \qquad Y = Ae^{B/x}$$

An example of a weighted solution with an exponential equation appears as one of the options for the TI-74 user-defined function program on pages 22 to 24. Finally, user's should remember that while the Y-squared weighting will yield a smaller value for the sum of the squares of the residuals from the untransformed solution, the Y-squared weighting will not result in a zero sum of the residuals. For example, in the solutions to George's original problem from V8N1P29 the average residual with weighting appears to be zero due to the truncation of the solution. The actual value of the average residual is 0.00268, substantially larger than the average error which occurs in solutions with polynomical curves, or when the residuals are calculated in the transformed coordinates.
