PPX Exchange

Vol. 5 Number 6 Copyright 1981 November/December 1981

potpourri

1. **IMPORTANT NOTICE:** Our office will be closed for the holiday season beginning Monday, December 21. We will be back at our desks Monday, January 4. The staff at PPX wishes you a joyful and safe holiday season.
2. In order to help us serve you better, please be sure to include your membership number on all correspondence and orders. If you have misplaced your number, it can be found on the mailing label on this newsletter.
3. In checking many of the orders received for programs and accessories through the PPX Department, we find that many members are failing to include the necessary state taxes. Taxes must be paid for every state with exception of the following five states: Alaska, Delaware, Montana, New Hampshire, and Oregon. Since Texas Instruments is responsible to pay these taxes back to each member's state, we are asking that the tax be included with all orders. We have been instructed to return any orders on which tax is not included. So, to insure prompt delivery of your orders, do not overlook this important item.
4. Very often there is a two week lag between the time you mailed your order and the time PPX receives it. This delay may also occur when a filled order is returned to you. For this reason, please allow four to six weeks for your order to be delivered from the time that you mailed it.
5. For those members who joined PPX through the "Free Membership" offer in the fall of 1980, it is time to renew your memberships. You should be receiving a notice to renew from our membership coordinator. If you fail to receive this notice by January 1, 1982 please let us know, so that you won't miss any issues of the Exchange.
6. We would like to remind our international members who reside outside of the United States that all payments to PPX must be in American currency in the form of a United States Postal Money Order or a check from an American bank.

An Introduction to Continued Fractions

By John W. Buchwald

Recently I began the study of continued fractions. This subject is new to me, and I have found the mathematics and associated TI-59 programming very interesting. It seems to be a large subject, that some of you may have already explored. I am hoping that you will find a few of my observations about these continued fractions of interest.

My reference is the Handbook of Mathematical Functions edited by Milton Abramowitz and Irene A. Stegun, U.S. National Bureau of Standards Applied Mathematics Series, June 1964 Edition. This reference gives theorems and many examples of continued fractions.

A continued fraction is a fraction of the form

$$f_n = \frac{A_n}{B_n} = b_0 + \cfrac{a_1}{b_1 + \cfrac{a_2}{b_2 + \cfrac{a_3}{b_3 + \cdots}}}$$

And is written in standard notation as

$$f_n = \frac{A_n}{B_n} = b_0 + \frac{a_1}{b_1+}\,\frac{a_2}{b_2+}\,\frac{a_3}{b_3+}\,\cdots\frac{a_n}{b_n}$$

Where a portion to the right is added to the denominator of the fraction to the left, except for the first term, b_o, which is added directly. If the limit of A_n/B_n as n approaches infinity exists, the infinite continued fraction is said to be convergent.

Continued fractions may be used to approximate many mathematical functions. In this respect, continued fractions are similar to infinite series. I find that while infinite series can be used to approximate the same functions as continued fractions, the methods of application vary greatly. For example, a continued fraction for tan x is:

$$\tan x = \frac{x}{1-}\,\frac{x^2}{3-}\,\frac{x^2}{5-}\,\frac{x^2}{7-}\,\frac{x^2}{9-}\,\cdots\frac{x^2}{(2m\text{-}1)}$$

$$\text{for } x \neq \frac{\pi}{2} \pm k\,\pi$$

Where m is the term number, and k is an integer. This

continued on page 4

Program Usability

By Charles Cole

Dear Editor,

After my article "What is Optimization?" appeared in the Jan/Feb issue of the PPX Exchange I began receiving varied responses to the ideas I discussed, including requests for programming assistance. Unfortunately, I am unable to answer these requests individually, but I would like to expand upon my suggestions for program-interface design via the accompanying article.

Additionally, I would like to apologize for the poorly written introduction involving a conversation between two fictitious students. The intent of this dialogue was to attack a "pet peeve" of mine by initiating the idea that an optimized program is not necessarily one written using the fewest number of steps possible. Squeezing the size of a program is sometimes necessary, but I still contend that when program space is available it should be used to improve the quality of the program by giving special attention to features such as usability and maintainability.

The dialogue was not intended to imply that polishing your programming skills by reworking an existing program is a poor practice. In fact, this activity can be an invaluable learning experience as well as an enjoyable exercise.

Sincerely,
Charles Cole

One of the most important characteristics of a good program is usability. Unless the user-interface is designed well, the program might as well not be written because it is not likely to be used as often as it should. Fortunately, good human engineering is easy to achieve in most cases.

When designing a program, it may be possible to develop an appropriate user-interface by adhering to the following guidelines.

1. Data entry should be simple and direct, independent of order if possible.
2. Editing capabilities should be provided, including the ability to correct incorrect inputs without re-entering correct inputs.
3. The user should be able to repeat execution of the program by re-entering only the variables that changed between the two runs.
4. If a printer is available, input and output should be appropriately labeled.

Of course, it is not always possible to achieve each of these goals; but the following design rules can be used to meet these guidelines in a majority of calculator applications.

1. A separate set of registers should be assigned to hold the input values. The contents of these registers should not be altered by program execution.
2. Each input should be assigned to a single key that is not used in any other way by the program.
3. Program execution should not result from an input key. Execution should begin only after all of the variables have been entered and the user has had the opportunity to review his inputs.

Following these rules obviously achieves the first goal. Assigning a single input to a key simplifies the user interface and delaying manipulation of the variables until all of the data has been entered avoids the need of sequencing the input.

The second guideline is also met using these rules assuming the correction is made before execution is begun. Independent entry of each of the variables allows for retaining correct inputs. However, if the program has been executed, good editing procedures are normally possible only if one of the following conditions is met.

If the results of running the program can simply be discarded, the correct input can be entered and the program run once more. The second part of rule 1 is of primary importance here. If the contents of the original input registers are altered by program execution then all of the original data will have to be re-entered, including data that was entered correctly.

If the results of executing the program are cumulative (build upon previous runs), the process must be reversable in order to have a successful editing routine. In cases such as these the user should not have to re-enter the incorrect data in order to back out the results if the error is discovered immediately. A designated key should provide access to a routine that retrieves the values from the input registers and automatically reverses the calculations. Then, the user should only have to correct the incorrect entries and proceed. If the error is discovered after additional runs have been made and the process is still reversable then the user should be able to enter the complete set of data input at the time of the incorrect entry and back out the results before continuing. Unfortunately, if the process is not reversable, the program usually has to be restarted from the beginning.

Implementing the third guideline is useful when the program may be used in "what-ifing" situations. An example of a program of this type is one for evaluating a home mortgage. In this instance the user could hold the period of the loan, the price of the home and the interest rate at constant values and experiment with the size of the down payment in order to find an acceptable monthly payment. Obviously, simple re-entering of one variable instead of all four is preferable. Again, following the rules outlined above will almost always guarantee compliance with this goal. However, this technique will not always work where multiple steps are required, especially those involving cumulative operations.

Fortunately, designing programs to be usable is not a difficult task. You simply have to put a little more time into the planning stage of developing your program. Adhering to the suggestions described here can be helpful in many cases; but remember that each program represents a different situation and that rules such as these cannot be applied universally. You will probably want to modify these ideas to meet your taste or even replace them entirely. But keep in mind the fact that someone will be using your program. So make it easy, even if it's just for yourself.

ADDRESS CHANGES

In order to ensure uninterrupted service, please submit address changes to PPX at least six weeks prior to the effective date of the change. Send your name, membership number, old and new addresses to:

PPX
P.O. Box 53
Lubbock, TX 79408

PROGRAMMING CORNER

(This column serves to inform programmers of what software our members would like to see and also what new non-PPX software is currently available.)

A MODEST PROPOSAL

Because of several requests of recent days from PPX members to know just who is marketing what in terms of software and programming aids for the TI-59, PPX will attempt to compile such a list and make it available to the membership. In order to collect this information, we will need your help. If you know of any programs, books, or other programming aids relating to the TI-59, please send us a short note detailing the content and availability of these materials. Please mail all such information to:

Texas Instruments - PPX - Software Search
P.O. Box 53
Lubbock, Texas 79408

Watch future "Programming Corner" columns for details on how to obtain a copy of this list.

OTHER SOURCES

Listed below is some of the software and programming aids that are currently available from sources other than PPX.

Electronic Charts for Air Navigation

Allows data for 54 VOR points to be loaded on a single magnetic card. Data cards with pre-stored data for every VOR on a sectional are available, or the data load program allows you to create your own data cards or to tailor purchased cards. Features of the program used in flight are:

- Calculates position using two VOR bearings (and/or visual or RDF bearings).
- Calculates position using DME range plus radial (and/or VOR radial plus estimated range).
- Calculates "TO GO" distance and magnetic course to any point in your data bank.
- Calculates "MADE GOOD" distance, magnetic course and speed between fixes.
- Dead reckons in real time, continously giving aircraft's (intended) latitude and longitude.
- Uses VOR frequencies in lieu of waypoint numbers. Non-VOR points are assigned "pseudo" frequencies, i.e., waypoint numbers.

This program is the result of more than a year of programming, de-bugging, revising, testing, and evaluating feed-back from other pilots. It was written by and for the VFR pilot who is alone in the airplane and who can only spend a minimum of "HEAD DOWN" time.

For more information on this program write to:

Liston & Associates
890 Saratoga Avenue
San Jose, CA 95129

Edu CALC MAIL STORE

This mail order book distributor lists forty books devoted to calculator programming and applications. To receive their catalog write to:

Educalc Mail Store
27963 Cabot Road
South Laguna, CA 92677

PROGRAMS WANTED

In order to help PPX members obtain the software they need, we publish program requests. Members who respond to these requests by submitting a PPX program are rewarded with special incentives. All such submissions should be on standard PPX submission forms. The author of the program found to be most suitable to fill each request will receive a Solid State Software™ module of their choice. Runners-up will receive a Speciality Pakette. When submitting a program to fill a "Programming Corner" request, please attach a note stating which request the submission is intended to fill.

The program requests for this issue are listed below. All submissions to fill these requests should be postmarked no later than February 28, 1982.

- A program that will calculate a target date based on a standard five day work week (given a date and number of days, it would calculate the future date). Ideally, this program could be designed to also neglect holidays.
- An internal rate of return program that will handle regular and irregular positive and negative cash flows of differing amounts not necessarily at regular intervals. It should be able to handle the switching of negative or positive cash flows as often as necessary. The solution should be the interest rate at which the present value of the revenues equal the present value of the cost.
- A program for use in the area of revolver and target shooting. Such a program would include, but not be limited to, the areas of ballistic flight of the bullet, determination of the correct weight of powder to use in loading cartridges with bullets of different weights and ballistic characteristics, and a method of project the velocity, ballistic path, and powder load to achieve various predetermined levels of impact.

Membership Renewals

If your membership about to expire? To ensure that you will miss no newsletters, catalogs, or ordering privileges, check the renewal table to find out if your membership will soon expire. (If your number is not included in the range of the table, it is not time for you to renew). The next issues of the Exchange will list additional renewal dates.

A renewal card and reminder will be sent to each member before the time to renew. Return the card to PPX with your check or money order for $20.00. Be sure to include your membership number on both your card and your check and mail to: Texas Instruments PPX Department, P.O. Box 109, Lubbock, TX 79408.

MEMBERSHIP NUMBER	RENEWAL MONTH
901983-903100	January
912577-913803	January
922788-923688	January
929149-929538	January
903101-903931	February
913804-914841	February
923689-924325	February
929539-930286	February

Continued Fractions (continued from page 1)

form appears to be simpler and to have broader limits than the usual power series form.

In writing programs for calculating continued fractions on a TI-59 calculator there seems to be two general ways of doing it.

A. Calculating and combining the terms from right to left starting at a preselected term n, where n is the number of terms minus one. The accuracy of this method may be checked by comparing the result with the result obtained by another method, or by comparing it with the result obtained by increasing n slightly and repeating the calculation. If the result does not change when n is increased, the function should have reached its limit.

B. Calculating and combining the terms from left to right and stopping the program automatically when the succeeding results reach a constant value within a small error, say 1×10^{-11}.

As an example of the "right to left" method let us write a program for the above continued fraction for the special case where b_o is a constant, and $a_1 = a_2 = \ldots a_n$, and $b_1 = b_2 = b_n$ are constants during the calculation (using the power up partition 6 *OP 17).

000	76	LBL	014	91	R/S	028	43	RCL
001	11	A	015	76	LBL	029	04	04
002	42	STO	016	14	D	030	85	+
003	01	01	017	42	STO	031	97	DSZ
004	91	R/S	018	04	04	032	05	05
005	76	LBL	019	43	RCL	033	00	00
006	12	B	020	01	01	034	23	23
007	42	STO	021	42	STO	035	43	RCL
008	02	02	022	05	05	036	02	02
009	91	R/S	023	43	RCL	037	95	=
010	76	LBL	024	03	03	038	42	STO
011	13	C	025	95	=	039	06	06
012	42	STO	026	35	1/X	040	91	R/S
013	03	03	027	65	×			

As an example of the use of this program, I have found that the Continued Fraction for $(1 + x)^{1/2}$ is

$$(1+x)^{1/2} = 1 + \frac{x}{2+} \; \frac{x}{2+} \; \frac{x}{2+} \cdots \quad \text{for } -1 < x < \infty$$

If we enter 20 for n, press A, enter 1 for b_o, press B, enter 2 for b_n, press C, and if we enter 1 for a_n (=x), and press D, the result is $1.414213562 = \sqrt{2}$. If we enter 2 for x, press D, the result is $1.732050808 = \sqrt{3}$. The accuracy of the result seems to decrease as the value of x is increased, or n must also be increased for full accuracy which in turn takes longer running time.

As an example of the "left to right" method let us write a program for the same continued fraction used above.

We may use the matrix formula:

$$\begin{bmatrix} A_n \\ B_n \end{bmatrix} = \begin{bmatrix} A_{n-1} & A_{n-2} \\ B_{n-1} & B_{n-2} \end{bmatrix} \begin{bmatrix} b_n \\ a_n \end{bmatrix}$$

We may wish to start with:

$$\begin{bmatrix} A_1 \\ B_1 \end{bmatrix} = \begin{bmatrix} b_0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} b_n \\ a_n \end{bmatrix}$$

We will find this to be correct as:

$$f_1 = \frac{A_1}{B_1} = b_0 + \frac{a_n}{b_n} = \frac{b_0 b_n + a_n}{b_n}$$

000	76	LBL	036	22	INV	072	48	EXC
001	12	B	037	52	EE	073	05	05
002	42	STO	038	69	OP	074	42	STO
003	02	02	039	21	21	075	06	06
004	91	R/S	040	53	(	076	43	RCL
005	76	LBL	041	43	RCL	077	10	10
006	13	C	042	03	03	078	48	EXC
007	42	STO	043	65	×	079	07	07
008	03	03	044	43	RCL	080	42	STO
009	91	R/S	045	05	05	081	08	08
010	76	LBL	046	85	+	082	53	(
011	14	D	047	43	RCL	083	53	(
012	42	STO	048	04	04	084	53	(
013	04	04	049	65	×	085	43	RCL
014	43	RCL	050	43	RCL	086	09	09
015	02	02	051	06	06	087	55	÷
016	42	STO	052	54	)	088	43	RCL
017	05	05	053	42	STO	089	10	10
018	01	1	054	09	09	090	54	)
019	42	STO	055	53	(	091	48	EXC
020	06	06	056	43	RCL	092	11	11
021	42	STO	057	03	03	093	55	÷
022	07	07	058	65	×	094	43	RCL
023	00	0	059	43	RCL	095	11	11
024	42	STO	060	07	07	096	54	)
025	01	01	061	85	+	097	75	-
026	42	STO	062	43	RCL	098	01	1
027	08	08	063	04	04	099	54	)
028	42	STO	064	65	×	100	50	I×I
029	11	11	065	43	RCL	101	77	GE
030	01	1	066	08	08	102	00	00
031	52	EE	067	54	)	103	38	38
032	01	1	068	42	STO	104	43	RCL
033	01	1	069	10	10	105	11	11
034	94	+/-	070	43	RCL	106	91	R/S
035	32	X:T	071	09	09			

Using this program as before for the example, we enter 1 for b_o, press B, 2 for b_n, press C, and if we enter 1 for x (=a_n), and press D, we get $1.414213562 = \sqrt{2}$. If we enter 2 for x and press D, we get $1.732050808 = \sqrt{3}$. After a calculation, the number of terms required may be found by pressing RCL 01. If values greatr than 35 are used for x an overflow error may result as the terms A_n and B_n become very large. Also, the running time of the program gets quite long.

For a more advanced example let us write a program using the "right to left" method for the continued fraction for the exponential integral.

$$E_n(z) = \lim_{p \to \infty} e^{-z} \left[\frac{1}{z+} \left(\frac{n}{1+} \; \frac{1}{z+} \right) \left(\frac{n+1}{1+} \; \frac{1+1}{z+} \right) \left(\frac{n+2}{1+} \; \frac{1+2}{z+} \right) \cdots \left(\frac{n+p-1}{1+} \; \frac{p}{z+} \right) \right]$$

Where n is the order of the exponential integral and p is the term number minus 1, when the terms are grouped in pairs as indicated above.

```
000  76 LBL   020  95  =    040  95  =
001  11  A    021  35 1/X   041  85  +
002  42 STO   022  65  ×    042  97 DSZ
003  01  01   023  43 RCL   043  04  04
004  91 R/S   024  04  04   044  00  00
005  76 LBL   025  95  =    045  18  18
006  12  B    026  85  +    046  43 RCL
007  42 STO   027  01  1    047  03  03
008  02  02   028  95  =    048  95  =
009  91 R/S   029  35 1/X   049  35 1/X
010  76 LBL   030  65  ×    050  65  ×
011  13  C    031  53  (    051  43 RCL
012  42 STO   032  43 RCL   052  03  03
013  03  03   033  01  01   053  94 +/-
014  43 RCL   034  85  +    054  22 INV
015  02  02   035  43 RCL   055  23 LNX
016  42 STO   036  04  04   056  95  =
017  04  04   037  75  -    057  42 STO
018  43 RCL   038  01  1    058  05  05
019  03  03   039  54  )    059  91 R/S
```

Some table values for the exponential integral are: $E_1(1)$ = .219383934, $E_2(1)$ = .1484955, $E_3(1)$ =, .1096920, $E_4(1)$ = .0860625, $E_1(2)$ = .048900511, $E_2(2)$ = .0375343, $E_3(2)$ = .0301334, $E_4(2)$ = .0250228.

To run the program, enter n, press A, enter p, press B, enter z and press C.

If we use a value of 40 for p we will obtain sufficient accuracy to get the above results. It seems, however, when a value of z = .5 is used then p must be increased to 80 or 60 to obtain full accuracy.

Using the "left to right" method for the same exponential integral and its continued fraction which is given above, we may write the matrix:

$$\begin{bmatrix} A_m \\ B_m \end{bmatrix} = \begin{bmatrix} A_{m-1} & A_{m-2} \\ B_{m-1} & B_{m-2} \end{bmatrix} \begin{bmatrix} b_m \\ a_m \end{bmatrix}$$

Where m is the term number minus 1. (Note: we have used m here instead of n for the term number minue 1, in order to avoid confusion with n which is used for the order of the exponential integral in this case.) We may start with:

$$\begin{bmatrix} A_3 \\ B_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ (z+n) & z \end{bmatrix} \begin{bmatrix} z \\ 1 \end{bmatrix} \text{ next term } \begin{bmatrix} 1 \\ (n+1) \end{bmatrix}$$

Here we have considered $b_o = 0$, $A_1/B_1 = 1/z$, $A_2/B_2 = \frac{1}{z+} \quad \frac{n}{1} = \frac{1}{z+n}$. We ignore the grouping of the terms but instead substitute the alternate different terms involved, adding 1 to the numerator, a_m, of each term each time it is used.

```
000  76 LBL   011  03  03   022  42 STO
001  11  A    012  42 STO   023  07  07
002  42 STO   013  04  04   024  53  (
003  01  01   014  42 STO   025  43 RCL
004  91 R/S   015  08  08   026  02  02
005  76 LBL   016  42 STO   027  85  +
006  13  C    017  09  09   028  43 RCL
007  42 STO   018  43 RCL   029  01  01
008  02  02   019  02  02   030  42 STO
009  01  1    020  42 STO   031  10  10
010  42 STO   021  06  06   032  54  )
```

```
033  42 STO   070  07  07   107  53  (
034  05  05   071  85  +    108  53  (
035  01  1    072  43 RCL   109  53  (
036  52 EE    073  06  06   110  43 RCL
037  01  1    074  65  ×    111  11  11
038  01  1    075  43 RCL   112  55  ÷
039  94 +/-   076  08  08   113  43 RCL
040  32 X:T   077  54  )    114  12  12
041  22 INV   078  42 STO   115  54  )
042  52 EE    079  12  12   116  48 EXC
043  00  0    080  43 RCL   117  13  13
044  42 STO   081  11  11   118  55  ÷
045  00  00   082  48 EXC   119  43 RCL
046  42 STO   083  03  03   120  13  13
047  13  13   084  42 STO   121  54  )
048  69 DP    085  04  04   122  75  -
049  20  20   086  43 RCL   123  01  1
050  53  (    087  12  12   124  54  )
051  43 RCL   088  48 EXC   125  50 IxI
052  03  03   089  05  05   126  77 GE
053  65  ×    090  42 STO   127  00  00
054  43 RCL   091  06  06   128  48  48
055  07  07   092  43 RCL   129  53  (
056  85  +    093  09  09   130  43 RCL
057  43 RCL   094  48 EXC   131  13  13
058  04  04   095  07  07   132  65  ×
059  65  ×    096  42 STO   133  43 RCL
060  43 RCL   097  09  09   134  02  02
061  08  08   098  01  1    135  94 +/-
062  54  )    099  44 SUM   136  22 INV
063  42 STO   100  10  10   137  23 LNX
064  11  11   101  43 RCL   138  54  )
065  53  (    102  10  10   139  42 STO
066  43 RCL   103  48 EXC   140  14  14
067  05  05   104  08  08   141  91 R/S
068  65  ×    105  42 STO
069  43 RCL   106  10  10
```

This program seems to do the job of finding the values of the exponential integral, but unfortunately it seems to be rather slow. It will calculate .2193839344 for E_1 (1) in about 7 minutes 30 seconds. Of course, the process could be speeded by decreasing the number of iterations and, thereby, decreasing the accuracy. Perhaps there are other techniques for computing continued fractions that are fast yet do not appreciably decrease the accuracy of the results. If any of you other PPX members come up with such a program, I would be interested in seeing it.

Letters to the Editor

Do you have comments or questions on the Exchange or other aspects of PPX that might benefit other members? We have always welcomed letters from our membership, and, therefore, we provide this space in the newsletter for you to share your views with your fellow members. Approximately 2-4 letters dealing with issues of general interest will be featured as space permits.

Dear Sir:

In your "Exchange" Vol. 5, Number 4 certain illegal key sequences found through experiments are mentioned in the editor's note to the "Fast Mode" article by Palmer O. Hansen, Jr. I would like to have some more information about it, such as "fractured display graphics", "access to hardwired functions", and "printer interrupts". Perhaps the articles are already published in earlier copies of the "Ex-

Letters To The Editor

(Continued from page 5)

change". I would appreciate it, if I could get this information from you.

Sincerely,
Edgar Schleupner

Dear Mr. Schleupner and other PPX members:

Since the key sequences that produce the above mentioned peculiarities are not endorsed by TI, we have not been the ones to discover or research these sequences. The most extensive coverage of these subjects can be found in the "TI PPC NOTES". The "TI PPC NOTES" is an independent club for users of TI programmable calculators which is in no way affiliated with TI. Subscriptions to the "TI PPC NOTES" are available for $20.00 per year for US bulk rate mail delivery, and the 1980 back issues are also available for $20.00 (First class and overseas subscriptions are higher). All orders for subscriptions to the "TI PPC NOTES" should be sent to:

TI PPC NOTES
9213 Lanham Severn Road
Lanham, Maryland 20706

In order to help PPX members know what is in the past newsletters, we are currently compiling a topical index. Look for this item in an upcoming issue.

Editor

The Editor Replies

My response to G.B. Stanton's query (Letters to the Editor, PPX Exchange, July/August 1981) provoked a barrage of letters from helpful PPXers. In his letter, Mr. Stanton asked if there was any way to get more than 100 data registers on the TI-59. Since it is a common misconception to think that a module can be used to expand the data register capacity, I printed Mr. Stanton's letter with the technically correct answer of "no". Several PPX members responded by reminding me that, although the number of registers cannot be increased, there are techniques that can be used to more efficiently deploy the available memory space. In order to exonerate myself in this matter, we will review the use of data packing and introduce the subject of pseudo-registers.

The idea behind data packing is to put more than one piece of numeric data in a data register. Since each data register occupies 64 bits of memory space, regardless of whether it contains a four digit or a twelve digit number, it makes sense that one method of effectively increasing the number of data registers is to utilize as much of the available memory as practical. There are several methods of packing data in a register, and no one method can be declared as being the best. The method of data packing that is best for a particular situation depends on the format of the data and the amount of available memory space. In general, the first criterion is that the packing and unpacking subroutines should be short enough that they are profitable. In other words, if the subroutines to pack and unpack two numbers in a register require 480 program steps, it would be better to use those 480 program steps as 60 data registers and just store the data in unpacked form. Secondly, it is desirable to have the packing and unpacking routine execute as fast as possible.

Two previous PPX articles have discussed several of the more involved methods of data packing, but they have failed to mention what is probably the most used technique. "The Art of Data Packing" (Vol. 2 No. 8, November 1978) explained a scheme to pack non-integer values, and "The Maximization of Data Packing for Advanced Programmers" (Vol. 3 No. 4, July 1979) examined the use of base conversions and logarithmic representations to pack data. Back issues of the PPX Exchange are available for $1.00 apiece plus $2.00 per order for postage and handling plus applicable state sales tax.

Often times data packing involves the packing of a pair of values into one register. One of the simplest methods of accomplishing this task is to separate the values by the decimal point. If, for instance, two positive integers of six digits or less are to be stored in one data register, Routine A (steps 000-017, listed below) will pack the data into the specified register. Routine B (steps 018-037) will unpack a specified register.

```
000   76 LBL      013   52 EE       026   65  ×
001   11  A       014   91 R/S      027   01  1
002   42 STO      015   74 SM*      028   52 EE
003   00  00      016   00  00      029   06  6
004   91 R/S      017   91 R/S      030   95  =
005   55  ÷       018   76 LBL      031   22 INV
006   01  1       019   12  B       032   52 EE
007   52 EE       020   42 STO      033   91 R/S
008   06  6       021   00  00      034   73 RC*
009   95  =       022   73 RC*      035   00  00
010   72 ST*      023   00  00      036   59 INT
011   00  00      024   22 INV      037   91 R/S
012   22 INV      025   59 INT
```

To pack the numbers 123456 and 654321 in register 1, perform the following. Enter the register (any register except zero) to receive the data, in our case "1", and press A. Enter the first number to be packed, 123456, and press R/S. Enter the second number to be packed, 654321, and press R/S. The numbers are now stored in register 1 in the form 654321.123456. To unpack the data, enter the register to be unpacked, 1, and press B. The first number, 123456, will be displayed. Press R/S and the second number, 654321, will be displayed. This method of data packing is quite straightforward. If the values to be packed are of a more arbitrary form, i.e., positive or negative non-integer values, one of the methods described in the afore mentioned articles should be used.

As PPX member Gregory Stark reminded me, one can also use program memory to create pseudo-registers for the storage of constant values. This technique is most often employed in the coding of custom modules, but only in a few situations would it be applicable to main memory programming. Since this method is of little value in our context, it will not be further detailed. If you would like a copy of Gregory Stark's article, "Trading Program Steps for Registers" send me a self addressed, stamped envelope, and I will send you a copy.

Jay Claborn
Editor

Register Display

This program was designed for use when the PC-100A/C Print Cradle is not available to list the data register contents. It will sequentially display register number and contents for all registers with non-zero contents in a user selected range (from register 1-89). The program also contains options to briefly display the register number and/or contents without halting the program and to display the guard digits.

USER INSTRUCTIONS

1. Select the desired options (if any) as shown below.

Key	Option
B	Causes register number to flash (pause) instead of program halting to display register number.
C	Causes register contents to flash instead of program halting to display register contents.
D	Causes guard digits to also be displayed. Use of this option voids the use of option "C".

2. Enter the number of the first register to be displayed, and press A. Enter the number of the last register to be displayed, and press R/S.
   - If option "B" was not selected, the first register number will be displayed. Press R/S to continue. (If "B" was selected, the register number will be briefly displayed, and the program will continue without halting.)
   - If option "C" was not selected, the register contents will be displayed. Press R/S to continue (If "C" was selected, the contents will be briefly displayed, and the program will continue without halting.)
   - If option "D" was selected, the program will halt after the register number has been displayed or flashed with the first eight digits of the register contents displayed in scientific notation (EE). Pressing R/S will display the remaining five digits. (For example, if the contents of a register are $1.234567890123 \times 10^{24}$, the first display will be 1.2345678 24, and the second display will be 0.90123.) Pressing R/S will continue the program.
   - When all designated registers have been displayed, the display will flash.

SAMPLE PROBLEM

Put the following data in registers 30, 32, 34, and 35, respectively.

$1.234567890123 \times 10^{24}$
$1.234567890123 \times 10^{-16}$
$-1.234500000007 \times 10^{10}$
23.

To achieve this storing process, perform the following key sequence.

1.234567 EE 24 + 8.90123 EE 17 = STO 30 x 1 EE 40 +/− = STO 32 1.2345 EE 10 + .07 = +/− STO 34 23 STO 35 CLR

Now, use the program to look the contents of these registers.

Enter	Press	Display	Comments
	D	0	Select option to display guard digits.
30	A	30	Starting register number.
35	R/S	30	Enter ending register number. Display shows register to be displayed next.
	R/S	1.2345678 24	
	R/S	0.90123	Contents of register 30.
	R/S	32	Next register.
	R/S	1.2345678 -16	
	R/S	0.90123	Contents of register 32.
	R/S	34	Next register.
	R/S	-1.2345000 10	
	R/S	0.00007	Contents of register 34.
	R/S	35	Next register.
	R/S	2.3000000 01	
	R/S	0.00000	Contents of register 35.
	R/S	"9.9999999 99"	Flashing display indicates that all registers have been displayed.

000	76	LBL	053	00	0	106	14	14
001	11	A	054	45	Y^x	107	95	=
002	29	CP	055	82	HIR	108	58	FIX
003	67	EQ	056	14	14	109	07	07
004	82	HIR	057	94	+/-	110	91	R/S
005	42	STO	058	95	=	111	25	CLR
006	00	00	059	52	EE	112	82	HIR
007	91	R/S	060	22	INV	113	15	15
008	82	HIR	061	52	EE	114	58	FIX
009	07	07	062	65	×	115	05	05
010	09	9	063	82	HIR	116	91	R/S
011	69	DP	064	15	15	117	22	INV
012	17	17	065	95	=	118	58	FIX
013	29	CP	066	77	GE	119	61	GTO
014	22	INV	067	00	00	120	01	01
015	86	STF	068	76	76	121	30	30
016	04	04	069	65	×	122	87	IFF
017	73	RC*	070	01	1	123	02	02
018	00	00	071	82	HIR	124	01	01
019	67	EQ	072	54	54	125	27	27
020	01	01	073	01	1	126	91	R/S
021	30	30	074	00	0	127	66	PAU
022	43	RCL	075	95	=	128	66	PAU
023	00	00	076	82	HIR	129	66	PAU
024	87	IFF	077	06	06	130	69	DP
025	00	00	078	75	-	131	20	20
026	00	00	079	53	(	132	43	RCL
027	29	29	080	29	CP	133	00	00
028	91	R/S	081	65	×	134	32	X:T
029	66	PAU	082	01	1	135	82	HIR
030	66	PAU	083	52	EE	136	17	17
031	73	RC*	084	07	7	137	77	GE
032	00	00	085	54	)	138	00	00
033	22	INV	086	22	INV	139	13	13
034	87	IFF	087	59	INT	140	25	CLR
035	03	03	088	82	HIR	141	35	1/X
036	01	01	089	05	05	142	91	R/S
037	22	22	090	55	÷	143	76	LBL
038	77	GE	091	01	1	144	12	B
039	00	00	092	52	EE	145	86	STF
040	44	44	093	07	7	146	00	00
041	86	STF	094	95	=	147	91	R/S
042	04	04	095	22	INV	148	76	LBL
043	50	I×I	096	87	IFF	149	13	C
044	82	HIR	097	04	04	150	86	STF
045	05	05	098	01	01	151	02	02
046	28	LOG	099	01	01	152	91	R/S
047	59	INT	100	94	+/-	153	76	LBL
048	82	HIR	101	65	×	154	14	D
049	04	04	102	01	1	155	86	STF
050	01	1	103	00	0	156	03	03
051	32	X:T	104	45	Y^x	157	91	R/S
052	01	1	105	82	HIR			

To key in the HIR and the two digit code following (XY) press STO 82 STO XY BST BST BST BST 2nd Del SST 2nd Del SST

CHECKSUM OUTPUT FOR PROGRAM

```
BANK?
          1.
INSERT CARD
         80.
PRESS LRN, INS(4x)
      LRN,     C
        108.
 118.1344168
```

More TI-59 Programming Seminars

Do you fully utilize the power of your programmable calculator? Or does the thought of actually writing your own software create a programming phobia syndrome? For the past two years Texas Instruments has provided programming seminars for individuals like yourself. In fact, several companies have realized the productivity gained by having their personnel trained on the TI-59 and have had us come to their company site to perform a seminar.

There may be a seminar coming to your area. These seminars are open to anyone with a TI-59 regardless of programming background. The seminars provide both beginning and intermediate programming training on the TI-59 in a "hands on" fashion. Tuition for the two day class is $150.00 per person. This includes the instruction, workbook, and luncheon for the two days. You should supply your own TI-59. To register send your check for $150.00 Payable to Texas Instruments to:

TI-59 Seminar
Texas Instruments
P.O. Box 10508 MS 5820
Lubbock, Texas 79408

If you have further questions regarding the seminars or if you would like information on setting up a company seminar, please contact Sherry Schroeder at 806-741-3277. The schedule of upcoming seminars is listed below.

Seminar Dates	Location
January 7-8	Dallas
January 14-15	Orlando
January 21-22	Chicago
January 25-26	Portland
January 28-29	Boston
February 4-5	Minneapolis
February 11-12	Houston
February 18-19	San Francisco
February 25-26	Denver
March 4-5	New Orleans
March 11-12	Philadelphia
March 18-19	Atlanta
March 25-26	Buffalo
March 29-30	New York

from the Analyst's Desk

•**OOPS!** The program listing of Misadventure in the September/October newsletter contained an error. Step 483 should be "4" instead of "0". The new Checksum number for bank three is shown below.

```
BANK?
          3.
INSERT CARD
        536.
        496.
        488.
PRESS LRN, INS (4x)
      LRN,     C
        532.
 124.6801746
```

•PPX would like to thank all program submitters for their continuing support. Fully realizing the amount of effort that goes into originating and documenting programs, we would like to outline some simple, but important, points to ensure your finished program reaches the user in its best form.

(1) If submitting a PC-100A/C printout as your program listing (which is something we recommend), please completely paste your printout to the submission. Listings secured only at the top and bottom have a tendency to tear during the photocopying process. We recommend the use of Glue Stick by Dennison to paste down listings. Simply taping your listing down is undesirable as the tape usually yellows the paper.

(2) To avoid damage to magnetic cards, submit them in a small envelope or plastic bag. Do not tape or paper clip magnetic cards to the submission forms. Also, be sure to label all magnetic cards with program title, partition, and card side.

(3) Mail your submission unfolded in a large envelope. Submissions which are folded and placed in small envelopes can be damaged by our automatic letter opener.

(4) If you use any program steps that cannot be keyed in directly, such as HIR (hierarchy) or Dsz nn (where nn is greater than 9), you should fully document how to enter these instructions. Also, if your program is keyed-in in one partition and recorded in another, it is helpful to include magnetic card recording procedures. Remember, not all of our members are as familiar with these steps as you are.

(5) Make sure your submissions are reproducable. Programs should be typed or neatly printed in black ink in order to produce clear readable copies. Please do not submit photocopies of your programs. We can only consider the original submission for acceptance.

(6) After completing your program, it is a good idea to let it sit for a few days. Then, pick it up and carefully proof read it. Check to see that register contents are documented (i.e., prestored constants and alphanumerics). Also, check for typographical errors.

(7) When submitting a program that is closely related or an improvement to another program, please include a note with your submission stating the differences and advantages of your program over other programs. This procedure will help our analysts not to mistake your program as a duplication.

Not only will following these simple steps help speed the

reviewing process of your program, but, more importantly, it makes your program of greater utility to the user.

On the other side of the coin, since so much time is spent documenting and reviewing PPX programs, you can almost bet that if you are having difficulties getting a sample problem to work on a purchased program, the difficulty is probably not due to a bug in the program. If you should encounter problems in running a PPX program, ask yourself the following questions.

(a) Is my calculator properly partitioned? The partition is stated on the User Instructions page. Further information regarding partitioning can be found in your Personal Programming Manual, pp. V-42 and VII-1.

(b) If a library module is required, is it inserted? The Submission Abstract and User Instructions pages note if a module is required.

(c) Are all alphanumeric codes and required constants stored in their appropriate registers? These requirements are usually noted in the User Instructions, Program Description, or Listing pages.

(d) Do all of my keystrokes match those of the author's listing? If you have trouble entering keycodes which have been produced from a PC-100A/C listing, please consult your Personal Programming Manual, pp. IV-44 and VI-6.

If you still have problems with a program after answering the above questions, please clearly document your problems on the PPX Program Memo form which is provided specifically for this purpose. As noted on the form, please include a duplicate set of your magnetic cards of the program to aid our analysts in locating the problem.

•PPX member Bob Sayre has written to remind PPXer's of an effect caused by Op 00 that is not mentioned in Personal Programming. The execution of Op 00 not only clears the print registers (heirarchy registers five through eight), but it also performs the integer operation on the value in the display.

•Stan Millwright has used an interesting number theory observation to create a nifty decimal to fraction converter program. Mr. Millwright realized that any positive, rational number can be formed from these three fractions: $\frac{0}{1}$, $\frac{1}{1}$, and $\frac{1}{0} = \infty$. For example, the fraction ¾ is formed as follows. Three-fourths is between $\frac{0}{1}$ and $\frac{1}{1}$; therefore, a new three number series is formed: $\frac{0}{1}$, $\frac{0+1}{1+1}$, and $\frac{1}{1}$, which is the same as $\frac{0}{1}$, $\frac{1}{2}$, and $\frac{1}{1}$. Performing this operation again produces $\frac{1}{2}$, $\frac{2}{3}$, $\frac{1}{1}$. And, finally, one more iteration yields $\frac{2}{3}$, $\frac{3}{4}$, $\frac{1}{1}$ and the desired value has been formed. This same method can be employed to find the fractional equivalent (to a user specified accuracy) in reduced form of a positive decimal number.

The familiar approximation for pi of 22/7 is accurate to the nearest hundreth. Using this program, one can find the approximation of 355/113 which is accurate to six decimal places. To use the program, enter the convergence criterion (ϵ), and press A. (The value of the calculated fraction will be within ϵ of the entered decimal number.) Enter the decimal number (>0), and press E. The numerater of the fraction will be returned in the display. Press x ↔ t, and the denominator will be displayed.

000	76	LBL	036	53	(	072	15	15
001	11	A	037	53	(	073	32	X:T
002	42	STO	038	53	(	074	43	RCL
003	18	18	039	43	RCL	075	14	14
004	92	RTN	040	10	10	076	77	GE
005	76	LBL	041	85	+	077	00	00
006	15	E	042	43	RCL	078	90	90
007	42	STO	043	12	12	079	43	RCL
008	14	14	044	54	)	080	16	16
009	32	X:T	045	42	STO	081	42	STO
010	01	1	046	16	16	082	12	12
011	22	INV	047	55	÷	083	43	RCL
012	67	EQ	048	53	(	084	17	17
013	00	00	049	43	RCL	085	42	STO
014	16	16	050	11	11	086	13	13
015	92	RTN	051	85	+	087	61	GTO
016	42	STO	052	43	RCL	088	00	00
017	10	10	053	13	13	089	36	36
018	42	STO	054	54	)	090	43	RCL
019	11	11	055	42	STO	091	16	16
020	42	STO	056	17	17	092	42	STO
021	12	12	057	54	)	093	10	10
022	42	STO	058	42	STO	094	43	RCL
023	13	13	059	15	15	095	17	17
024	77	GE	060	75	-	096	42	STO
025	00	00	061	43	RCL	097	11	11
026	33	33	062	14	14	098	61	GTO
027	00	0	063	54	)	099	00	00
028	42	STO	064	50	I×I	100	36	36
029	13	13	065	32	X:T	101	43	RCL
030	61	GTO	066	43	RCL	102	17	17
031	00	00	067	18	18	103	32	X:T
032	36	36	068	77	GE	104	43	RCL
033	00	0	069	01	01	105	16	16
034	42	STO	070	01	01	106	92	RTN
035	10	10	071	43	RCL			

•In order to learn more about his TI-59, PPX member L.M. Leeds devised a program to calculate the value of "e" to 440 decimals. Although the program is only of "one time" value, it is presented here since the fact that the TI-59 has this capability might be of interest to some of you.

The series for e as given in handbooks is:

$e = 1 + 1/1! + 1/2! + 1/3! + \ldots$

To obtain e to 440 decimals necessitates that the series be carried to where n! is greater than 10^{440}. This corresponds to n! = 229!. Obviously, it is rather impractical to use the series in the form stated above, but the series can be written as: $e = 1 + 1/1(1 + 1/2(1 + 1/3(1 + 1/4(1 + \ldots$ It is this equation that the program solves.

Each of the 229 divisions involves a dividend of 444+ digits and results in a quotient of 440+ digits. A special rapid division process was devised which operates in a 100,000,000,000 positional format. This was possible since the maximum divisor was a three digit number.

To run the program, repartition the calculator by pressing 10 Op 17 then press RST and R/S. The first display will be the integer value of 2. After pressing R/S the display will show the first ten decimals. Repeated pressing of R/S will display more groups of ten digits, until a display of .3333333333 indicates all the calculated values have been displayed. (If any display has less than ten digits, leading zeroes should be added.)

Since the run time to calculate all 440 decimal places is about 7.7 hours, listed below are the few simple program modifications needed to produce only 50, 100, or 200

continued on page 10

Analyst's Desk (continued from page 9)

decimals.

Program Steps To Be Changed	Number of Decimals Produced 50	100	200
008 009 010	4 1 Nop	7 1 Nop	1 2 1
073 074	4 Nop	9 Nop	1 9
129 130	5 Nop	1 0	2 0

000	47	CMS	038	69	DP	076	00	00	114	00	0
001	71	SBR	039	22	22	077	43	RCL	115	42	STO
002	01	01	040	69	DP	078	55	55	116	02	02
003	07	07	041	23	23	079	55	÷	117	42	STO
004	43	RCL	042	69	DP	080	43	RCL	118	08	08
005	06	06	043	24	24	081	06	06	119	01	1
006	42	STO	044	97	DSZ	082	95	=	120	01	1
007	10	10	045	05	05	083	75	-	121	42	STO
008	02	2	046	00	00	084	59	INT	122	03	03
009	02	2	047	15	15	085	91	R/S	123	05	5
010	09	9	048	73	RC*	086	95	=	124	05	5
011	42	STO	049	07	07	087	65	×	125	42	STO
012	01	01	050	72	ST*	088	43	RCL	126	04	04
013	42	STO	051	08	08	089	06	06	127	42	STO
014	09	09	052	69	DP	090	95	=	128	07	07
015	73	RC*	053	27	27	091	91	R/S	129	04	4
016	02	02	054	69	DP	092	73	RC*	130	04	4
017	55	÷	055	28	28	093	07	07	131	42	STO
018	43	RCL	056	97	DSZ	094	91	R/S	132	00	00
019	01	01	057	00	00	095	69	DP	133	42	STO
020	95	=	058	00	00	096	27	27	134	05	05
021	59	INT	059	48	48	097	97	DSZ	135	25	CLR
022	72	ST*	060	68	NOP	098	00	00	136	92	RTN
023	04	04	061	97	DSZ	099	00	00	137	71	SBR
024	65	×	062	09	09	100	92	92	138	01	01
025	43	RCL	063	01	01	101	03	3	139	13	13
026	01	01	064	37	37	102	35	1/X	140	69	DP
027	75	-	065	43	RCL	103	91	R/S	141	31	31
028	73	RC*	066	06	06	104	61	GTO	142	43	RCL
029	02	02	067	44	SUM	105	00	00	143	06	06
030	95	=	068	55	55	106	69	69	144	44	SUM
031	94	+/-	069	05	5	107	01	1	145	10	10
032	65	×	070	06	6	108	52	EE	146	61	GTO
033	43	RCL	071	42	STO	109	01	1	147	00	00
034	06	06	072	07	07	110	00	0	148	15	15
035	95	=	073	04	4	111	42	STO			
036	74	SM*	074	03	3	112	06	06			
037	03	03	075	42	STO	113	01	1			

Précis

This column presents the abstracts of some of the new PPX programs which have been recently accepted. The programs were selected by our analysts as being ones that would be of special interest to our members. You can purchase these programs at a cost of $4.00 each. Send your order to: Texas Instruments: PPX Department, P.O. Box 109, Lubbock, TX 79408. Include an additional $2.00 for postage and handling plus applicable state tax.

If you have a need for a specific program, send a note to PPX. There is a chance that the program may have already been written. If it has, we will put the abstract in the next issue of the Exchange. Requests for programs not yet written will be placed in the "Programming Corner" column.

038013H Inventory Program for up to 158 Materials
Up to 999,999 items of 158 materials can be handled by this program. Items can be put in or taken out using User Keys A & B respectively. The current inventory of any material can be obtained by User Key C. The inventory of each material in any size group of consecutively numbered materials can be obtained by User Key D. User Key E will print out the entire inventory. The Data Pack Program 08 in the Math/Utility module must be used. The Program is easily modified to 316 materials using four magnetic cards.
R.F. Fulton, Friendswood, TX
201 Steps, PC-100A, Mod 10

038014H Inventory Tally File: 100 Item
Generates and manipulates (retrieves, increments, decrements, updates, deletes) an inventory file of up to 100 items. Items are accessed by user-defined stock code numbers. Ten calculator display digits may be partitioned as needed between stock code number digits and quantity digits. Worst-case retrieval time is 46 seconds.
J. David Cogdell, Dunlap, IL
156 Steps

098023H Unit Price Construction Contract-Bid
Given: Material, labor, equipment, and subcontractor costs per unit, and the desired % profit for each category, this program calculates: material, labor, equipment, and subcontractor prices based on prorating the % desired profit figures, and makes the extended price calculation. The program also calculates and prints the total costs, prices and profit by category and total job. Profit % and unit price may be varied.
Don L. Normoyle, Rock Island, IL
702 Steps, PC-100A

128021H Loan Amortization
Given the loan amount, number of periods over which it is to be repaid, number of periods per year, nominal or effective annual interest rate, and balloon if any (joint or separate), the program produces: normal level payment, final level payment to adjust for rounding in previous payments, total payments to be made, total interest to be paid, interest rate per period, effective annual interest rate if the nominal rate was entered, or nominal rate if the effective rate was entered. Given the number of elapsed periods, the program produces: current payment, payments to date, and payments remaining. Each of these three is further divided between principal and interest. Calculates interest on the full loan for a partial period.
Herman Burstein, Wantagh, NY
476 Steps

188048H Bond Yield - ROR
Computes the net rate of return of a bond during the period of ownership. The program accounts for broker fees both at purchase and at sale of the bond. It also computes the coupon adjustments typically made at transfer of ownership for both purchase and sale, and incorporates these cash flows into the ROR calculation.
P.E. Teeter, Des Peres, MO
711 Steps

208077H Nonlinear Least Squares
This program performs a nonlinear least squares fit of up to 15 data points to a user defined function: y = mf(s,c,d) + b

where m,c, and b are the fit parameters, and d is a specified constant.
John R. Long, Madison, NJ
543 Steps, Mod 1

208078H Hyperbolic Curve Fitting
Data that is asymptotic to an axis can be represented by a hyperbolic curve, $Y=(A+BX)/(1+CX)$. Using least squares analysis, constants A, B, and C are calculated. Program contains a routine for calculating values of Y based in a given X and a routine for calculating standard deviation between given data and calculated data.
George F. Poland, Providence, RI
421 Steps, PC-100A

228070H t-Test of Means Drawn From Two Samples
Tests differences between two means drawn from two separate samples when given a standard error. Useful for research data when testing a mean in one sample versus a mean in other sample. Allows user to enter means rather than raw data and computes t-score and level of significance either 1-tailed or 2-tailed test.
Martin P. Roth, Valhalla, NY
159 Steps

248006H Markov Chain Simulation
Let Q be an n by n probability matrix, n less than or equal to 8, and let p be a 1 by n probability vector defining an n-state Markov chain C with initial distribution p and transition matrix Q. The program provides runs of C.
H. Jurgensen, Reinheim 1, Germany
233 Steps, Mod 1

268046H Three Constant Weibull Distribution
The user enters three data points. He then enters guesses on the threshhold parameter and on each guess obtains an error value, showing how well the curve fit the points. Having obtained about the lowest error value from this method, extrapolation can be made to other points. All entered parameters are printed and labeled.
Gregory L. Stark, Pittsford, NY
389 Steps, PC-100A, Mod 1

348033H Sine, Cosine, Exponential Integrals
This program is designed to calculate the sine, cosine, and exponential integral where x is entered by the user and falls between 0 and 23.
Richard Kirchhofer, Fremont, CA
227 Steps

398242H Square Root - Double Precision
Program extracts the square root of a 24 digit (max.) number and returns a 24 digit result. The program is intended for practical calculations rather than a demonstration of capability.
Laurance M. Leeds, Sun City, AZ
476 Steps

398244H Error Propagation
Takes any equation of the form $V=K(x^a)(y^b)(z^c)\ldots$ where the user supplies x, y, z, ..., their errors, and the powers they are raised to (positive, negative, or fractional), and supplies V and the error of V. Can handle up to 9 variables.
Bradley John Roth, Overland Park, KS
164 Steps

668174H Maximum Stress - Rectangular Flat Panel
This program calculates the maximum axial stress for a rectangular simply supported plate, membrane, or thin plate loaded by normal pressure. The decision of which of the above three conditions apply is accounted for and the appropriate equations utilized. Ten inputs are required, four of which are easily read from enclosed curves.
Alfred L. Priestley, Burbank, CA
635 Steps, Mod 1

668176H Calculating the F-Chart for Solar Collectors
The program calculates the F-Chart for Solar collectors for Jan. thru Dec. when given all the required data.
S.J. Charney, Freedom, PA
320 Steps, PC-100A

668175H Solar Heating Analysis: The F-Chart Method
This program uses the widely accepted f-chart method to evaluate the long term performance of a flat plate solar collector system. The required input parameters include the standard collector properties: heat removal factor, heat loss coefficient, absorbtion-transmission product, average insolation, collector area, air flow rate, storage and flow capacitances, and monthly heating load. The program is divided into two parts. The first part calculates the K correction factors to modify the standard f-chart equations. These factors are then used in the second part to determine the month-by-month and fraction of total energy supplied by the solar collector system.
Matt Russell, Glendora, CA
689 Steps, Mod 10

668179H F Chart: I
For liquid systems, program calculates the solar supplied fraction of the heating load. Method used is presented in the book Solar Heating Design by the f-Chart Method. The required inputs are the collector parameters, load and meteorological data. Program will calculate the solar fraction for three user defined collector areas.
Curtis Jackson, Lk Havasu Cy, AZ
912 Steps, PC-100A, Mod 10

698030H Fire Sprinkler System Design
The user enters basic data and system dimensions. The program calculates total design friction loss and average allowable friction loss per foot of pipe. Input and output parameters are in useful table format if the PC-100 is used. The method lends itself to all system topologies: grid, tree, loop, etc. Extensive "what-if" capabilities allow trade offs.
Terence J. Fitzpatrick, Independence, KY
419 Steps

758010H Label Exterminator
This program prints the actual keystrokes necessary to convert a program using conventional labels of one using direct addressing. This conversion usually results in a faster execution time and sometimes yields a shorter overall program. Included is a routine which calculates the net gain or loss of program steps along with error-trapping and user-prompting routines.
Paul Hofstadler, Albuquerque, NM
900 Steps, PC-100A

788057H Astronomical Variables
This program will calculate the nutation in longitude, nutation in obliquity, obliquity of the ecliptic, and the equation of the equinoxes for any date at any time.
S.T. Bradley, Council Bluffs, IA
849 Steps, PC-100A, Mod 1

798053H Percentage of Trap
This program uses filtered readings of reflected printing ink density as a method of determining % of trapping (laydown of one ink over another). The results are calculated from anti logarithms of ratios of density.
Dave G. Vequist, Pittsburg, KS
273 Steps, PC-100A

798056H Estimating Ink Coverage
A method of estimating ink mileage based on square inches of coverage and paper surface. Ink cover is determined by extent of text, bold type, halftones and solid areas. Paper surfaces that affect coverage may be selected from Gloss, Dull, Matt or Wove, Vellum and News. Colors of ink are grouped according to expected coverage.
David G. Vequist, Pittsburg, KS
269 Steps, PC-100A

908220 Chemical Raw Material Formulation Calculations
This program will calculate the volume (if weights are in grams), cost, % weight, % volume, and % cost of each ingredient. It will also calculate total weight, total milliliters, total cost, specific gravity, lbs./gal, total gallons (if weight is in lbs), $/lb, $/gal, and $/ton. The program is based on the equation: density = grams/millilter. It will handle 15 ingredients in a formulation. Inputs are the weight, specific gravity, and cost ($/lb) of each ingredient.
Robert R. Fulton, Friendswood, TX
480 Steps, PC-100A

918286H Rule Away
Similar to Hamarabi (#918060F) except; factors of equations are random to give variety, the concept of plague is introduced, and your overall score is given at the end of the game which allows you to compare players. "If the crown fits, wear it."
David S. Lane, Clearwater, FL
398 Steps, PC-100A

918294H Chess Descriptive Notation
Program prints all moves of a standard chess game in descriptive notation. Up to 99 moves may be entered, more than adequate for most games. Special moves, such as castling or en passant, can be entered with ease. Includes all standard symbols.
Ronald W. Rushing, Albany, GA
389 Steps, PC-100A, Mod 10

918303H Phantom Banner with Sequential Background
Phantom banner generated against a background which is sequenced across, downward and diagonally. Decimal to binary conversion is used to form banner letters.
Serge Borodin, Brooklyn, NY
320 Steps, PC-100A

928050H Travelin' Man
This program presents a verbal problem in algebra analogous to the classic presentation. A problem is given which demands algebraic translation from English to Mathematics, numerical answers, and deductive reasoning. Printed text is the manner in which the problem is given. Multiple choice and numerical-answer problems appear.
Robert Sutliff, Bronx, NY
400 Steps, PC-100A

The PPX Exchange is published bimonthly and is the only newsletter published by Texas Instruments for TI-59 owners. Members are invited to contribute articles and items of general interest to other TI-59 users. Authors of accepted feature articles for the newsletter will receive their choice of either a one year complimentary PPX membership or a Solid State Software™ module. Please double-space and type all submissions, and forward them to:

Texas Instruments, PPX
P.O. Box 53
Lubbock, Texas 79408
Attn: PPX Exchange Editor

TEXAS INSTRUMENTS
INCORPORATED
PPX • P.O. Box 53 • Lubbock, Texas 79408
U.S. CALCULATOR PRODUCTS DIVISION

ADDRESS CORRECTION REQUESTED