PPX Exchange

Vol. 4 Number 5 Copyright 1980 September/October 1980

Important Notice:

There has been an address change at PPX for all correspondence that is sent with checks or money orders enclosed. Please send all membership renewals and program/accessory orders to:

Texas Instruments
P.O. Box 109
Lubbock, TX 79408

Do not change the P.O. Box number on your present order forms or postage paid envelopes. This will void the permit number and the U.S. postal service will return the envelope to you with postage due. New order forms and envelopes are being printed. Until then, please mail your orders to the new address by putting your order form/renewal card into an envelope.

All other correspondence (without checks) should still be sent to Texas Instruments, c/o PPX Department, P.O. Box 53, Lubbock, TX 79408.

WORKING WITH LARGE NUMBERS

By Charles Cole

A problem that has plagued many programmers is that of working with extremely large or small numbers that exceed the limitations of the calculator's range and result in an over/under flow error. Three methods of combating this problem are discussed below. Our example is taken from the field of statistics where working with large numbers in the form of factorials occurs frequently.

Suppose that you want to determine the number of five-card hands that can be dealt from a double deck (104 cards). The number of potential combinations of n items taken r at a time is found as:

$$\binom{n}{r} = \frac{n!}{r!\,(n-r)!}$$

Or

$$\binom{104}{5} = \frac{104!}{5!\,(104-5)!} = \frac{104!}{5! \times 99!}$$

As the range of the calculator is exceeded by 70!, it is obvious that the above cannot be directly evaluated.

continued on page 2

PRINT CODE CHALLENGE

Results

By Jay Claborn - PPX Program Analyst

After three days of keying in routines and timing their runs, I wonder what ever possessed me to recommend that we run this contest in the first place (next time I'll know to require magnetic cards). But be that as it may, the response to Jeff Jones' Challenge in the May/June Exchange was in a word, overwhelming. I thank everyone who participated in the contest.

A total of 163 attempts at creating the best program to produce the print code table were submitted. Because there were several different interpretations of exactly what output was to be created, I altered my judging accordingly:

1) The fastest running routine for each type of output was chosen. All times were recorded on the same TI-59/PC-100A combination, since run times will vary with different TI-59/PC-100 A/C combinations. (Relative speeds will be the same.)
2) The routines with the fewest equivalent program steps were selected. The number of equivalent program steps is equal to the number of program steps plus eight times the number of prestored data registers used.

Before presenting these routines, I would like to make a few observations about these programs. All three of the "fast" programs used prestored data. Registers 90 through 99 were used so the program would fit completely on one card side. Simple experimentation shows that if a number is 7 or more digits long, it is time effective to recall this data out of a prestored register rather than to have the number stored in program step locations. It might also be noted that it requires less equivalent program steps to recall numbers of 11 or more digits out of data registers versus program memory.

Each of the "fast" programs also makes use of the HIR command to directly access the print registers (HIR registers 5-8), so that the print registers do not have to be repeatedly reloaded.

Now for the results, envelopes please (drum roll).

continued on page 3

LARGE NUMBERS

Technique I—Eliminating Terms

One way to handle calculations such as this is to eliminate terms that drive the calculator to an overflow state; but are cancelled out in the normal course of the calculation. Examining the equation, we find that

$$\frac{104!}{5! \times 99!} = \frac{104 \times 103 \times 102 \times 101 \times 100 \times 99!}{5! \times 99!}$$

So, by dividing out 99! before performing the actual calculation, the problem is reduced to

$$\frac{104 \times 103 \times \ldots \times 100}{5!} = \frac{1.1035502 \times 10^{10}}{120} = 91962520.$$

In a program, this problem would be handled in the general form

$$\binom{n}{r} = \frac{n \times (n-1) \times \ldots \times (n-r+1)}{r!}$$

Technique II—Calculation Order

The above approach significantly increases the potential magnitude of the input data. But, is it good enough? What if you needed to determine the number of combinations of 10,000 items taken 30 at a time? Using Technique I, we obtain

$$\frac{10000 \times 9999 \times \ldots \times 9971}{30!} = \frac{\text{OVERFLOW}}{2.6525286 \times 10^{32}}$$

Hence, if the numerator is found first, the calculation overflows. But in this case, we can keep the calculator from overflowing by evaluating the denominator first and multiplying its reciprocal by the factors in the numerator:

$$\frac{10000 \times 9999 \times \ldots 9971}{30!}$$

$$= 1/30! \times 10000 \times 9999 \times \ldots \times 9971$$

$$= 3.7699876 \times 10^{-33} \times 10000 \times 9999 \times \ldots \times 9971$$

$$= 3.6093543 \times 10^{87}.$$

Going further, let's consider the problem of finding the number of possible combinations of 500 items taken 75 at a time. In this case both the numerator and the denominator exceed the calculator's range when using Technique I.

$$\frac{500 \times 499 \times \ldots \times 426}{75!} = \frac{\text{OVERFLOW}}{\text{OVERFLOW}}$$

The method used in the last example won't work either as computing the reciprocal of the denominator produces an underflow error state. But, by examining the equation once more, we discover that

$$\frac{500 \times 499 \times \ldots \times 426}{75!}$$

$$= \frac{500 \times 499 \times \ldots \times 426}{75 \times 74 \times \ldots \times 1}$$

$$= \frac{500}{75} \times \frac{499}{74} \times \ldots \times \frac{426}{1}$$

Now, we may prevent an overflow state by alternating multiplication and division operations in order to keep the calculation within range:

$$500 \div 75 \times 499 \div 74 \times \ldots \times 426 \div 1$$

$$= 3.0781288 \times 10^{90}$$

As you can see, carefully planning the "order" in which a calculation is executed is often the simplest way to prevent an over/underflow state.

Technique III—Data Transformation

Another useful technique is to transform your data before executing the primary calculation and then reversing the transform to obtain the final result. A handy transform for the factorial function is found in program 11 of the Math/Utilities Library. In this program a factorial's natural log is found from an asymptotic series for arguments as large as 4553879000. Taking advantage of this transform we find that

$$\frac{n!}{r!\,(n-r)!} = \exp\left(\ln \frac{n!}{r!\,(n-r)!}\right)$$

$$= \exp\left(\ln n! - (\ln r! + \ln (n-r)!)\right)$$

Continuing our last example,

$$\frac{500!}{75!\ 425!}$$

$$= \exp\left(\ln 500! - (\ln 75! + \ln 425!)\right)$$

$$= \exp\left(2611.330458 - (251.8904022 + 2151.083076)\right)$$

$$= \exp 208.3569803$$

$$= 3.0781288 \times 10^{90}$$

If you don't have access to the Math/Utilities Library you may compute ln n! as:

$$\ln n!$$

$$= \ln (n \times (n-1) \times \ldots \times 1)$$

$$= \ln n + \ln (n-1) + \ldots + \ln 1$$

And don't forget another application of Technique I. This calculation can be sped up by eliminating terms that cancel each other out. Note that

$$\exp\left(\ln n! - (\ln r! + \ln (n-r)!)\right)$$

$$= \exp\left(\ln n! - \ln (n-r)! - \ln r!\right)$$

Now, working with the first two terms inside the parentheses:

$$\ln n! - \ln (n-r)!$$

$$= \ln n + \ln (n-1) + \ldots + \ln (n-r+1)$$

$$+ \ln (n-r) + \ln (n-r-1) + \ldots + \ln 1$$

$$- \ln (n-r) - \ln (n-r-1) - \ldots - \ln 1$$

$$= \ln n + \ln (n-1) + \ldots + \ln (n-r+1)$$

and we've eliminated the addition and subtraction of 2(n-r) terms, or in this example 2 (500 – 75) = 950 operations. The use of this method or Technique II may also be essential to keeping your calculation of ln n! within range.

So, if your calculations overflow, don't despair. The examples presented here pertain to a specific type of calculation; but you can probably make one of these techniques work for your problem too.

RESULTS

Running at a swift 7.3 seconds was **Lem Matteson's** 94 step (equivalent steps, mind you) condensed print table routine.

```
  01234567
0  0123456
1 789ABCDE
2 -FGHIJKL
3 MNOPQRST
4 .UVWXYZ+
5 ×*Γπe(),
6 ↑%!/='×X̄
7 ²?÷◊Ⅱ▵ΠΣ
```

To run:

1) Enter program below. (Remember to key in HIR as STO 82 BST BST DEL SST.)
2) Press 10 OP 17 and enter prestored registers.
3) Press 6 OP 17 and record bank one onto a magnetic card.
4) Press RST R/S to start program.

000	01	1	012	69	OP	024	07	7	036	05	05
001	00	0	013	05	05	025	42	STO	037	97	DSZ
002	69	OP	014	43	RCL	026	00	00	038	00	00
003	17	17	015	94	94	027	43	RCL	039	00	00
004	43	RCL	016	69	OP	028	99	99	040	27	27
005	96	96	017	03	03	029	82	HIR	041	06	6
006	69	OP	018	43	RCL	030	37	37	042	69	OP
007	03	03	019	95	95	031	43	RCL	043	17	17
008	43	RCL	020	69	OP	032	98	98	044	00	0
009	97	97	021	04	04	033	82	HIR	045	91	R/S
010	69	OP	022	69	OP	034	38	38			
011	04	04	023	05	05	035	69	OP			

PRESTORED DATA

100000102.	94
304050607.	95
10203.	96
405060708.	97
1.001010101	98
1.000100101	99

NOTE: Registers 98 and 99 contain data out in the guard digits and should be stored as follows:

.101010101 ÷ 100 + 1 = STO 98

.0100010101 ÷ 100 + = STO 99

William Beebe's 83 step program also runs in 7.3 seconds and produces the table below.

```
  0 1 2 3 4 5 6
7 8 9 A B C D E
- F G H I J K L
M N O P Q R S T
. U V W X Y Z +
× * Γ π e ( ) ,
↑ % ! / = ' × X̄
² ? ÷ ◊ Ⅱ ▵ Π Σ
```

An honorable mention goes to **Tom Washburn,** whose program with simular output ran in 7.4 seconds.

To run this program enter steps 000-042, and, to alleviate guard digit hassles, enter the prestored data as program steps 202-239.

NOTE: Step 233 is keyed in 2nd CLR. Press RST R/S to begin execution.

000	69	OP	028	94	94	212	00	0
001	00	00	029	82	HIR	213	01	1
002	01	1	030	36	36	214	00	0
003	00	0	031	82	HIR	215	10	E'
004	69	OP	032	38	38	216	00	0
005	17	17	033	69	OP	217	70	RAD
006	43	RCL	034	05	05	218	00	0
007	90	90	035	97	DSZ	219	60	DEG
008	82	HIR	036	99	99	220	00	0
009	06	06	037	00	00	221	50	I×I
010	43	RCL	038	23	23	222	00	0
011	91	91	039	06	6	223	10	E'
012	82	HIR	040	69	OP	224	00	0
013	07	07	041	17	17	225	00	0
014	43	RCL	042	91	R/S	226	40	IND
015	92	92				227	00	0
016	82	HIR				228	30	TAN
017	08	08				229	00	0
018	69	OP	202	01	1	230	00	0
019	05	05	203	00	0	231	10	E'
020	07	7	204	01	1	232	00	0
021	42	STO	205	00	0	233	20	CLR
022	99	99	206	01	1	234	00	0
023	43	RCL	207	10	E'	235	10	E'
024	93	93	208	00	0	236	00	0
025	82	HIR	209	00	0	237	00	0
026	37	37	210	00	0	238	00	0
027	43	RCL	211	01	1	239	10	E'

It should be pointed out, for those of you who are real speed optimization freaks, that the DSZ loop in the above program slows execution. For example Mr. Beebe's run time can be reduced to 6.1 seconds if the DSZ loop is eliminated as shown below.

000	69	OP	036	43	RCL	072	43	RCL
001	00	00	037	94	94	073	94	94
002	01	1	038	82	HIR	074	82	HIR
003	00	0	039	36	36	075	36	36
004	69	OP	040	82	HIR	076	82	HIR
005	17	17	041	38	38	077	38	38
006	43	RCL	042	69	OP	078	69	OP
007	90	90	043	05	05	079	05	05
008	82	HIR	044	43	RCL	080	43	RCL
009	06	06	045	93	93	081	93	93
010	43	RCL	046	82	HIR	082	82	HIR
011	91	91	047	37	37	083	37	37
012	82	HIR	048	43	RCL	084	43	RCL
013	07	07	049	94	94	085	94	94
014	43	RCL	050	82	HIR	086	82	HIR
015	92	92	051	36	36	087	36	36
016	82	HIR	052	82	HIR	088	82	HIR
017	08	08	053	38	38	089	38	38
018	69	OP	054	69	OP	090	69	OP
019	05	05	055	05	05	091	05	05
020	43	RCL	056	43	RCL	092	43	RCL
021	93	93	057	93	93	093	93	93
022	82	HIR	058	82	HIR	094	82	HIR
023	37	37	059	37	37	095	37	37
024	43	RCL	060	43	RCL	096	43	RCL
025	94	94	061	94	94	097	94	94
026	82	HIR	062	82	HIR	098	82	HIR
027	36	36	063	36	36	099	36	36
028	82	HIR	064	82	HIR	100	82	HIR
029	38	38	065	38	38	101	38	38
030	69	OP	066	69	OP	102	69	OP
031	05	05	067	05	05	103	05	05
032	43	RCL	068	43	RCL	104	06	6
033	93	93	069	93	93	105	69	OP
034	82	HIR	070	82	HIR	106	17	17
035	37	37	071	37	37	107	91	R/S

A complete table is generated in 9.2 seconds by **Charles M. Taylor's** 145 step program.

```
     0  1  2  3  4  5  6  7
  0     0  1  2  3  4  5  6
  1  7  8  9  A  B  C  D  E
  2  -  F  G  H  I  J  K  L
  3  M  N  O  P  Q  R  S  T
  4  .  U  V  W  X  Y  Z  +
  5  ×  *  √  π  e  (  )  ,
  6  ↑  %  ↓  /  =  '  x  x̄
  7  ²  ?  ÷  ‼  Ⅲ  ▵  Π  Σ
```

Follow instructions from Mr. Matteson's program to enter and run this program. This is a "Kamikaze" program, i.e. the magnetic card must be reread for each run.

000	76	LBL	022	05	05	044	36	36
001	11	A	023	43	RCL	045	82	HIR
002	01	1	024	99	99	046	38	38
003	00	0	025	82	HIR	047	43	RCL
004	69	OP	026	05	05	048	91	91
005	17	17	027	43	RCL	049	82	HIR
006	00	0	028	95	95	050	37	37
007	82	HIR	029	82	HIR	051	43	RCL
008	05	05	030	06	06	052	99	99
009	43	RCL	031	43	RCL	053	82	HIR
010	98	98	032	94	94	054	35	35
011	82	HIR	033	82	HIR	055	69	OP
012	06	06	034	07	07	056	05	05
013	43	RCL	035	43	RCL	057	97	DSZ
014	97	97	036	93	93	058	90	90
015	82	HIR	037	82	HIR	059	00	00
016	07	07	038	08	08	060	41	41
017	43	RCL	039	69	OP	061	06	6
018	96	96	040	05	05	062	69	OP
019	82	HIR	041	43	RCL	063	17	17
020	08	08	042	92	92	064	91	R/S
021	69	OP	043	82	HIR			

PRESTORED DATA

NOTE: Load registers as follows:

7	STO	90
1.000010001	STO	91
1000100010	STO	92
500060007	STO	93
3000400	STO	94
10002	STO	95
600070010	STO	96
4000500	STO	97
100020003	STO	98
100	STO	99

1 EE 12 +/- PRD 92 PRD 93 PRD 94 PRD 95
PRD 96 PRD 97 PRD 98 PRD 99 INV EE
1 SUM 92 SUM 93 SUM 94 SUM 95
SUM 96 SUM 97 SUM 98 SUM 99

The routine with the fewest steps was submitted by **Clarence Drumeller**. This little jewel has only 43 steps and generates the table below. Simply press RST R/S to run.

```
   0     1  2     3  4     5  6
7  8     9  A     B  C     D  E
-  F     G  H     I  J     K  L
M  N     O  P     Q  R     S  T
.  U     V  W     X  Y     Z  +
×  *     √  π     e  (     )  ,
↑  %     ↓  /     =  '     x  x̄
²  ?     ÷  ‼     Ⅲ  ▵     Π  Σ
```

000	08	8	015	43	RCL	030	43	RCL
001	42	STO	016	02	02	031	02	02
002	00	00	017	85	+	032	85	+
003	02	2	018	69	OP	033	69	OP
004	00	0	019	02	02	034	05	05
005	00	0	020	43	RCL	035	43	RCL
006	00	0	021	02	02	036	02	02
007	02	2	022	85	+	037	85	+
008	42	STO	023	69	OP	038	97	DSZ
009	02	02	024	03	03	039	00	00
010	25	CLR	025	43	RCL	040	00	00
011	01	1	026	02	02	041	13	13
012	85	+	027	85	+	042	91	R/S
013	69	OP	028	69	OP			
014	01	01	029	04	04			

And if you are a real minimum step fanatic. Mr. Drumeller's program could be shortened by 12 steps (with the sacrifice of longer run time) in the following manner:

000	08	8	011	32	X:T	022	97	DSZ
001	42	STO	012	85	+	023	04	04
002	00	00	013	69	OP	024	00	00
003	01	1	014	23	23	025	12	12
004	32	X:T	015	84	OP*	026	97	DSZ
005	05	5	016	03	03	027	00	00
006	42	STO	017	02	2	028	00	00
007	04	04	018	00	0	029	04	04
008	00	0	019	00	0	030	91	R/S
009	42	STO	020	00	0			
010	03	03	021	02	2			

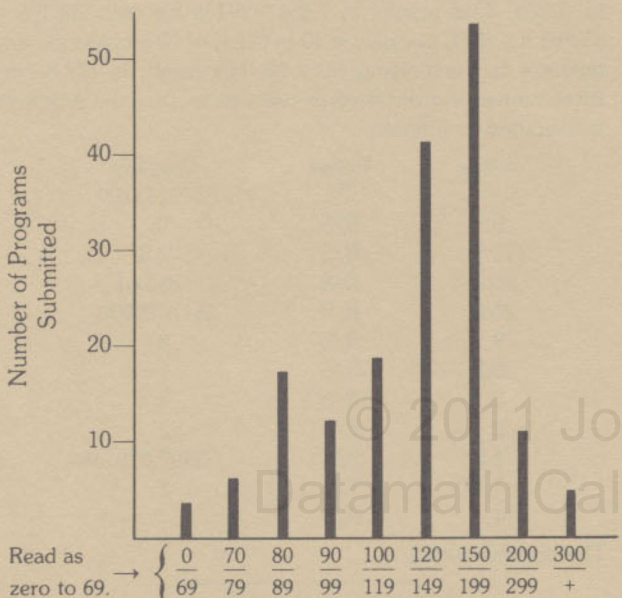
If a complete table in the fewest steps is desired, **C. L. Lees'** program is a winner. Press E to start program.

000	76	LBL	024	00	00	048	11	A
001	11	A	025	39	39	049	71	SBR
002	43	RCL	026	47	CMS	050	00	00
003	01	01	027	08	8	051	06	06
004	69	OP	028	42	STO	052	69	OP
005	21	21	029	00	00	053	03	03
006	52	EE	030	69	OP	054	11	A
007	04	4	031	22	22	055	52	EE
008	85	+	032	43	RCL	056	02	2
009	43	RCL	033	02	02	057	22	INV
010	01	01	034	52	EE	058	52	EE
011	69	OP	035	06	6	059	69	OP
012	21	21	036	71	SBR	060	04	04
013	95	=	037	00	00	061	69	OP
014	22	INV	038	08	08	062	05	05
015	52	EE	039	69	OP	063	02	2
016	92	RTN	040	01	01	064	44	SUM
017	76	LBL	041	11	A	065	01	01
018	15	E	042	52	EE	066	97	DSZ
019	02	2	043	02	2	067	00	00
020	42	STO	044	22	INV	068	00	00
021	01	01	045	52	EE	069	30	30
022	01	1	046	69	OP	070	98	ADV
023	71	SBR	047	02	02	071	92	RTN

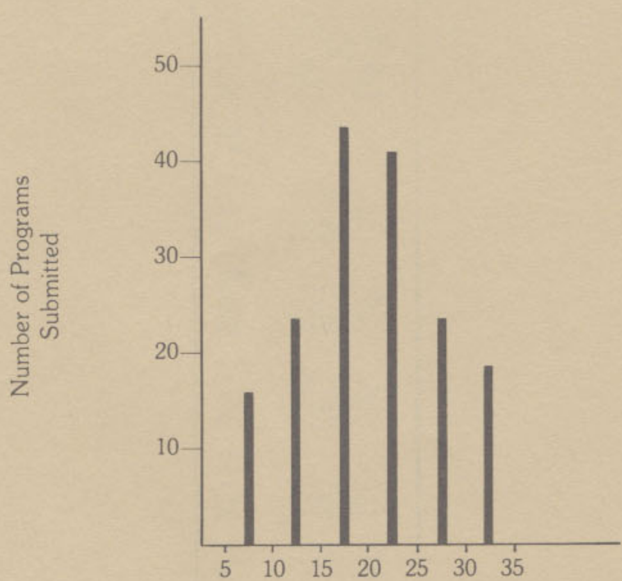
Our final routine, the long form print table submitted by **Gordon S. Cornish,** requires only 15 steps. This program produces each print code and its associated alpha character on a line. To run press RST R/S.

000	25	CLR	005	69	OP	010	20	20
001	42	STO	006	04	04	011	61	GTO
002	00	00	007	69	OP	012	00	00
003	43	RCL	008	06	06	013	03	03
004	00	00	009	69	OP	014	91	R/S

As a matter of interest for those of you who submitted programs I have summarized the run time and step distribution of the entries in charts.



EQUIVALENT PROGRAM STEPS



RUN TIME (Seconds)

Letters to the Editor

Do you have comments, compliments or (shudder) complaints about PPX-59? We have always welcomed letters from our membership, and starting with this issue we will provide space in each newsletter to share your views on PPX with your fellow members. Approximately 3-5 letters dealing with issues of general interest will be featured in each issue. Letters will be edited to fit the space available.

Dear Sir:

I hope some of our members can come up with a method for debugging programs in which the PC-100C printer in the TRACE mode would not only print the functions being performed but also the program line number where the computations are being done. This is especially necessary in long programs with many subroutines.

Mohibullah Durrami
New York, NY

Dear Mr. Durrami:

A program of this type would truly be wonderful when debugging; however, it is not possible because the trace mode is not accessible to program control (except, of course, to turn it on and off via flag nine).

Dear Editor:

This is in regard to the contribution from Hal Schneider under "From the Analyst's Desk" in the May/June 1980 PPX Exchange.

I have used the guard digits for a long time now to store an extra alphanumeric character. As long ago as April 1979, I wrote the Exchange editor of their utility. Many of my programs use this technique, and I have even had a utility program (PPX #908127E) accepted that prints the alpha of the two guard digits.

Gregory L. Stark
Hawthorne, CA

Dear Sirs:

Since I am still getting acquainted with my 59 and programming, have you considered the possibility of listing the names of the more experienced 59 programmers in various areas of the country who might be interested in tutoring or otherwise assisting the new tyros to the ranks?

Bill Barker
Houston, TX

Dear Mr. Barker:

We cannot release membership names and addresses without express permission. How about it, PPX members? Any members who are willing to help out new programmers may send their names and addresses to the "PPX Exchange"; we will maintain a file and forward the information on to new members who request assistance.

CUSTOM DIET PLANNER

By Barry Tepperman

The program below will allow the user to determine a caloric schedule for dieting. It won't tell what foods to eat, but will calculate the limit on the number of calories that a person can intake a day and still lose a given amount of weight per week. The program is effective for individuals in the age range of 20-80 years. **A PC-100A/C is required.**

USER INSTRUCTIONS:

1. Partition to 639.39 by pressing 4 Op 17.
2. Enter program and store the following contents into data memory. Do this by entering the constant then pressing STO XX, where XX is the register number. Record the program and data on 4 card sides.

Constant	Register	Constant	Register
-0.084615	04	364130	34
39.8	05	143035	35
85.0	17	362437	36
150.0	18	4133	37
240.0	19	43132726	38
350.0	20	17441735	39

3. Initialize program by pressing B'.
4. Enter sex by entering 0 if male or 1 if female, press R/S.
5. Enter height in centimeters (cm) and press R/S. If you have your height in inches, first multiply by 2.54. This will convert inches to centimeters.
6. Enter weight in kilograms (kg), press R/S. To convert weight in pounds to kg, divide your weight (lbs) by 2.205.
7. Enter age in format YY.MM, where YY is the year and MM is the month. The body surface area (BSA) in square meters and basal metabolic rate (BMR) will be computed and printed.
8. Enter activity schedule by first entering the hours spent doing the activity, the pressing R/S.

   Activities:
   a. Sleeping
   b. Sitting
   c. Standing
   d. Walking
   e. Exercising

The sum of the hours spent doing the activities must be equal to 24. If not, the program prints out 0 and returns to the beginning of user instruction step 8.

9. Enter total desired weight loss in kg, press R/S.
10. Enter the number of weeks allotted for weight (kg) to be lost, press R/S. The program will print a weekly schedule and, at the end of the weight-loss period, the program will print out the caloric schedule so that the final weight can be maintained.

Example:

A man aged 26 years, 2 months, wishes to lose seven pounds during four weeks. He is 5 feet, 10 inches tall and weighs 188 pounds. His daily schedule calls for the following:

a. 8 hours of sleep
b. 9.5 hours of sitting
c. 3 hours of standing
d. Walks for 2 hours
e. Exercises for 1.5 hours

Solution: (1) Convert pounds to kilograms by dividing by 2.205. This results in 7 lbs = 3.175 kg and 188 lbs = 85.261 kg. (2) Convert 5 ft 10 in (total of 70 inches) to centimeters by multiplying by 2.54. His height is 177.8 cm. After converting the English units to metric, the program is executed as follows:

Enter	Press	Display
	B'	3327133100.
0.	R/S	0.
177.8	R/S	177.8
85.261	R/S	85.261
26.2	R/S	3617620000.
8.	R/S	8.
9.5	R/S	9.5
3.	R/S	3.
2.	R/S	2.
1.5	R/S	3507.678146
3.175	R/S	3.
4.	R/S	

The printed output:

```
CUSTOM SI DIET PLAN

       177.8      CM
      85.261      KG
  26.16666667    AGE
  2.032992501    BSA
  76.41186807    BMR

DAILY CALORIC USE:
  640.8011226   EXER
  585.8753121   WALK
  549.2581051     UP
  985.6131553    SIT
   746.130451    BMR
  3507.678146    SUM

       3.175    LOSS
          4.     WKS

          1.    WEEK
    84.46725      KG
   2632.71879    CAL
  2.016818052    BSA
  75.80065494    BMR

          2.    WEEK
     83.6735      KG
  2561.551164    CAL
  2.008664624    BSA
  75.49094571    BMR

          3.    WEEK
    82.87975      KG
  2526.011752    CAL
  2.000466172    BSA
  75.17957096    BMR

          4.    WEEK
      82.086      KG
  2490.512991    CAL
  1.992222008    BSA
  74.86650505    BMR

  610.9729544   EXER
   558.603844   WALK
  523.6911037     UP
  939.7345917    SIT
  697.0119585    BMR
  3330.014452    SUM
```

000	76	LBL
001	11	A
002	91	R/S
003	29	CP
004	32	X:T
005	67	EQ
006	89	π
007	86	STF
008	01	01
009	76	LBL
010	89	π
011	92	RTN
012	76	LBL
013	12	B
014	91	R/S
015	42	STO
016	01	01
017	01	1
018	05	5
019	03	3
020	00	0
021	69	OP
022	04	04
023	43	RCL
024	01	01
025	69	OP
026	06	06
027	91	R/S
028	42	STO
029	00	00
030	02	2
031	06	6
032	02	2
033	02	2
034	69	OP
035	04	04
036	43	RCL
037	00	00
038	69	OP
039	06	06
040	91	R/S
041	42	STO
042	02	02
043	59	INT
044	85	+
045	53	(
046	43	RCL
047	02	02
048	22	INV
049	59	INT
050	55	÷
051	01	1
052	93	.
053	02	2
054	95	=
055	42	STO
056	02	02
057	01	1
058	03	3
059	02	2
060	02	2
061	01	1
062	07	7
063	69	OP
064	04	04
065	43	RCL
066	02	02
067	69	OP
068	06	06
069	92	RTN
070	76	LBL
071	13	C
072	91	R/S
073	32	X:T
074	04	4
075	42	STO
076	07	07
077	01	1
078	00	0
079	42	STO
080	08	08
081	00	0
082	42	STO
083	09	09
084	76	LBL
085	48	EXC
086	32	X:T
087	72	ST*
088	08	08
089	44	SUM
090	09	09
091	91	R/S
092	32	X:T
093	69	OP
094	28	28
095	97	DSZ
096	07	07
097	48	EXC
098	32	X:T
099	72	ST*
100	08	08
101	44	SUM
102	09	09
103	02	2
104	04	4
105	32	X:T
106	43	RCL
107	09	09
108	67	EQ
109	16	A'
110	99	PRT
111	61	GTO
112	13	C
113	76	LBL
114	16	A'
115	71	SBR
116	65	×
117	71	SBR
118	85	+
119	92	RTN
120	76	LBL
121	14	D
122	91	R/S
123	42	STO
124	22	22
125	91	R/S
126	42	STO
127	23	23
128	35	1/X
129	65	×
130	43	RCL
131	22	22
132	95	=
133	42	STO
134	24	24
135	22	INV
136	44	SUM
137	00	00
138	65	×
139	01	1
140	01	1
141	00	0
142	02	2
143	93	.
144	03	3
145	01	1
146	01	1
147	95	=
148	42	STO
149	25	25
150	92	RTN
151	76	LBL
152	15	E
153	02	2
154	06	6
155	02	2
156	02	2
157	69	OP
158	04	04
159	43	RCL
160	00	00
161	69	OP
162	06	06
163	43	RCL
164	09	09
165	75	-
166	43	RCL
167	25	25
168	95	=
169	42	STO
170	26	26
171	01	1
172	05	5
173	01	1
174	03	3
175	02	2
176	07	7
177	69	OP
178	04	04
179	43	RCL
180	26	26
181	69	OP
182	06	06
183	29	CP
184	32	X:T
185	77	GE
186	14	D
187	43	RCL
188	24	24
189	22	INV
190	44	SUM
191	00	00
192	71	SBR
193	30	TAN
194	53	(
195	01	1
196	55	÷
197	05	5
198	02	2
199	54	)
200	44	SUM
201	02	02
202	71	SBR
203	42	STO
204	71	SBR
205	65	×
206	71	SBR
207	85	+
208	92	RTN
209	76	LBL
210	50	IXI
211	22	INV
212	87	IFF
213	01	01
214	49	PRD
215	93	.
216	00	0
217	08	8
218	00	0
219	07	7
220	06	6
221	09	9
222	94	+/-
223	42	STO
224	04	04
225	03	3
226	07	7
227	93	.
228	05	5
229	07	7
230	06	6
231	09	9
232	02	2
233	03	3
234	42	STO
235	05	05
236	22	INV
237	86	STF
238	01	01
239	76	LBL
240	49	PRD
241	92	RTN
242	76	LBL
243	30	TAN
244	43	RCL
245	00	00
246	45	YX
247	93	.
248	04	4
249	02	2
250	05	5
251	65	×
252	43	RCL
253	01	01
254	45	YX
255	93	.
256	07	7
257	02	2
258	05	5
259	65	×
260	93	.
261	00	0
262	00	0
263	07	7
264	01	1
265	08	8
266	04	4
267	95	=
268	42	STO
269	03	03
270	01	1
271	04	4
272	03	3
273	06	6
274	01	1
275	03	3
276	69	OP
277	04	04
278	43	RCL
279	03	03
280	69	OP
281	06	06
282	92	RTN
283	76	LBL
284	85	+
285	05	5
286	42	STO
287	07	07
288	03	3
289	02	2
290	42	STO
291	08	08
292	73	RC*
293	08	08
294	42	STO
295	09	09
296	69	OP
297	38	38
298	69	OP
299	37	37
300	76	LBL
301	44	SUM
302	73	RC*
303	08	08
304	44	SUM
305	09	09
306	69	OP
307	38	38
308	97	DSZ
309	07	07
310	44	SUM
311	43	RCL
312	09	09
313	92	RTN
314	76	LBL
315	65	×
316	05	5
317	42	STO
318	07	07
319	01	1
320	04	4
321	42	STO
322	08	08
323	43	RCL
324	00	00
325	55	÷
326	06	6
327	09	9
328	93	.
329	08	8
330	05	5
331	03	3
332	02	2
333	02	2
334	05	5
335	95	=
336	42	STO
337	15	15
338	76	LBL
339	34	ΓX
340	43	RCL
341	08	08
342	85	+
343	06	6
344	95	=
345	42	STO
346	21	21
347	85	+
348	01	1
349	02	2
350	95	=
351	42	STO
352	06	06
353	73	RC*
354	08	08
355	65	×
356	53	(
357	43	RCL
358	16	16
359	85	+
360	53	(
361	73	RC*
362	21	21
363	75	-
364	43	RCL
365	16	16
366	54	)
367	54	)
368	65	×
369	43	RCL
370	15	15
371	95	=
372	72	ST*
373	06	06
374	69	OP
375	38	38
376	97	DSZ
377	07	07
378	34	ΓX
379	92	RTN
380	76	LBL
381	42	STO
382	53	(
383	43	RCL
384	04	04
385	65	×
386	43	RCL
387	02	02
388	85	+
389	43	RCL
390	05	05
391	54	)
392	65	×
393	43	RCL
394	03	03
395	95	=
396	42	STO
397	16	16
398	01	1
399	04	4
400	03	3
401	00	0
402	03	3
403	05	5
404	69	OP
405	04	04
406	43	RCL
407	16	16
408	69	OP
409	06	06
410	92	RTN
411	76	LBL
412	17	B'
413	01	1
414	05	5
415	04	4
416	01	1
417	03	3
418	06	6
419	03	3
420	07	7
421	03	3
422	02	2
423	69	OP
424	01	01
425	03	3
426	00	0
427	00	0
428	00	0
429	03	3
430	06	6
431	02	2
432	04	4
433	00	0
434	00	0
435	69	OP
436	02	02
437	01	1
438	06	6
439	02	2
440	04	4
441	01	1
442	07	7
443	03	3
444	07	7
445	00	0
446	00	0
447	69	OP
448	03	03
449	03	3
450	03	3
451	02	2
452	07	7
453	01	1
454	03	3
455	03	3
456	01	1
457	00	0
458	00	0
459	69	OP
460	04	04
461	69	OP
462	05	05
463	98	ADV
464	71	SBR
465	11	A
466	71	SBR
467	12	B
468	71	SBR
469	30	TAN
470	71	SBR
471	50	IXI
472	71	SBR
473	42	STO
474	98	ADV
475	01	1
476	06	6
477	01	1
478	03	3
479	02	2
480	04	4
481	02	2
482	07	7
483	04	4
484	05	5
485	69	OP
486	01	01
487	01	1
488	05	5
489	01	1
490	03	3
491	02	2
492	07	7
493	03	3
494	02	2
495	69	OP
496	02	02
497	03	3
498	05	5
499	02	2
500	04	4
501	01	1
502	05	5
503	00	0
504	00	0
505	04	4
506	01	1
507	69	OP
508	03	03
509	03	3
510	06	6
511	01	1
512	07	7
513	06	6
514	02	2
515	00	0
516	00	0
517	00	0
518	00	0
519	69	OP
520	04	04
521	69	OP
522	05	05
523	71	SBR
524	13	C
525	71	SBR
526	86	STF
527	98	ADV
528	71	SBR
529	14	D
530	02	2
531	07	7
532	03	3
533	02	2
534	03	3
535	06	6
536	03	3
537	06	6
538	69	OP
539	04	04
540	43	RCL
541	22	22
542	69	OP
543	06	06
544	04	4
545	03	3
546	02	2
547	06	6
548	03	3
549	06	6
550	69	OP
551	04	04
552	43	RCL
553	23	23
554	69	OP
555	06	06
556	98	ADV
557	01	1
558	42	STO
559	33	33
560	76	LBL
561	90	LST
562	04	4
563	03	3
564	01	1
565	07	7
566	01	1
567	07	7
568	02	2
569	06	6
570	69	OP
571	04	04
572	43	RCL
573	33	33
574	69	OP
575	06	06
576	71	SBR
577	15	E
578	98	ADV
579	01	1
580	44	SUM
581	33	33
582	97	DSZ
583	23	23
584	90	LST
585	71	SBR
586	16	A'
587	71	SBR
588	86	STF
589	92	RTN
590	76	LBL
591	86	STF
592	03	3
593	02	2
594	42	STO
595	06	06
596	05	5
597	42	STO
598	07	07
599	03	3
600	09	9
601	42	STO
602	08	08
603	76	LBL
604	96	WRT
605	73	RC*
606	08	08
607	69	OP
608	04	04
609	73	RC*
610	06	06
611	69	OP
612	06	06
613	69	OP
614	38	38
615	69	OP
616	36	36
617	97	DSZ
618	07	07
619	96	WRT
620	43	RCL
621	34	34
622	69	OP
623	04	04
624	43	RCL
625	09	09
626	69	OP
627	06	06
628	92	RTN

PROGRAMMING CORNER

Seven new programs are now available as a result of the program requests featured in the May/June issue. (Not to mention that seven PPX members have new Solid State Software™ modules for their efforts.) These programs are "Callaway Handicap" (PPX #988045) by Edmund K. Gravely, "Callaway Handicapping Method" (PPX #988046) by Dan S. Hirschfeld, "Relief Valve Sizing" (PPX #668129) by Jeff Jones, "Safety-Pressure Relief Valve Sizing" (PPX #668130) by Robert A. Hinton, "Pressure Relief Valve Sizing" (PPX #668131) by Rob R. MacGregor, "Non Parametric Tests" (PPX #228044) by Guy H. Nelson, and "Multiple Range Tests" (PPX #228045) by Thomas Masurat.

If there are programs you would like to see made available through PPX, send the request.

We will be accepting submissions to fill the following request until December 31, 1980:

- A flat pattern development program (i.e., Cone, Square to Round, Elbow, and Intersecting patterns) for metal fabrication industry.
- A program to calculate the half-spaces or extra spaces needed between words in each line of type to provide for uniform line length throughout a printed page. The required inputs would be column width and lengths for each line.
- A program to analyze time-series data (i.e., univariate ARIMA models) using the Box-Jenkins approach.
- A program in which the TI-59 is the opponent in the game "Othello".

Solving Systems of Linear Equations

By Douglas P. Anderson

Are you tired of finding determinants, especially for those systems of eight equation in eight variables? What about those systems that are not independent, or when the number of equations is not equal to the number of variables?

Here is a simpler method I learned at Iowa State University in a linear algebra course taught by Colby Kegley. It is called row reduction of matrices. First, I will define a row reduced matrix and how to compute it. Then, I will show how row reduced matrices are used to solve systems of linear equations.

For a matrix to be in row reduced form, it must meet these criteria: (1) Any zero row must appear at the bottom. (2) In any non-zero rows, the first non-zero entry is a one, called a leading one. (3) Any column containing a leading one has zeroes for the rest of its entries. (4) As you scan down the non-zero rows, the leading ones move to the right. The following are not in row reduced form as they violate one of the four criteria respectively.

$$\begin{bmatrix} 0 & 0 \\ 1 & 2 \end{bmatrix} \quad \begin{bmatrix} 2 & 3 \\ 0 & 6 \end{bmatrix} \quad \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

It is interesting to note that a matrix has one and only one row reduced form.

Any matrix can be transformed to its row reduced form by performing a series of row operations on it. There are 3 row operations. (1) Multiplying a row by a constant. (2) Adding a multiple of one row to another row. (3) Switching two rows. For example, consider the matrix $\begin{bmatrix} 0 & 3 \\ 2 & 1 \end{bmatrix}$. First, switch rows 1 and 2 to obtain $\begin{bmatrix} 2 & 1 \\ 0 & 3 \end{bmatrix}$. Now, take row 1 times ½ and row 2 times 1/3 to get $\begin{bmatrix} 1 & \frac{1}{2} \\ 0 & 1 \end{bmatrix}$. Finally, take row 2 times −½ and add it to row 1, $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. This is the row reduced form of the original matrix.

Now, to get down to business, I will apply this technique to solve the simultaneous equations $x + 2y = 1$ and $3x + y = 0$. First, form an augmented matrix of the coefficients and constants, $\left[\begin{array}{cc|c} 1 & 2 & 1 \\ 3 & 1 & 0 \end{array}\right]$. Now, find the row reduced form of this matrix which is $\left[\begin{array}{cc|c} 1 & 0 & -1/5 \\ 0 & 1 & 3/5 \end{array}\right]$. The solution of these equations, then, is $x = -1/5$, $y = 3/5$.

The system $2x + y = 1$, $x + 2y = 0$, $3x + 3y = 2$ is solved similarly.

Augmented Matrix $\left[\begin{array}{cc|c} 2 & 1 & 1 \\ 1 & 2 & 0 \\ 3 & 3 & 2 \end{array}\right]$ Row Reduced form $\left[\begin{array}{cc|c} 1 & 2 & 0 \\ 0 & 1 & -1/3 \\ 0 & 0 & 1 \end{array}\right]$

But, the bottom row says 0 = 1! This means that the system is inconsistent, that is, no solution exists.

As a final example, consider the system

$$\begin{aligned} x - 2y \quad\quad\; + w &= 0 \\ x - 2y + z + 4w &= 1 \\ 2x - 4y + z + 5w &= 1 \end{aligned}$$

As before:

Augmented matrix $\left[\begin{array}{cccc|c} 1 & -2 & 0 & 1 & 0 \\ 1 & -2 & 1 & 4 & 1 \\ 2 & -4 & 1 & 5 & 1 \end{array}\right]$ Row Reduced form $\left[\begin{array}{cccc|c} 1 & -2 & 0 & 1 & 0 \\ 0 & 0 & 1 & 3 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right]$

This system is dependent and its solution is $x - 2y + w = 0$, $z + 3w = 1$ or $x = 2y - w$, $z = 1 - 3w$.

The method of row reduction is truly simple and will solve any system or tell you that it is not solvable. For large matrices, it is much faster than determinants. It also gives you the power to solve systems where the number of equations is not equal to the number of variables.

To make things even simpler for myself, I developed a program to find the row reduced form of any matrix with up to as many elements as 61 minus the number of columns.

USER INSTRUCTIONS:

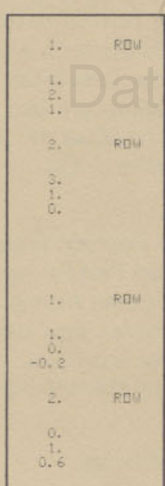
1. Repartion to 319.79
2. Load program
3. Enter number of rows in matrix, press A.
4. Enter number of columns in matrix, press B.
5. Enter row number of data to be entered, press C.
6. Enter row elements in order, one at a time, press D after each entry.
7. Repeat 5 and 6 for each row.
8. Compute row form of original matrix by pressing E.

PROGRAM NOTES:

1. This program requires the PC-100 A/C print cradle, but it is easy to modify this program for use without it.
2. **The Math/Utilities module is needed for this program.**
3. Calculator round off error may prevent a row reduced matrix element from being exactly zero, but user discretion will recognize such errors by considering significant digits.
4. Registers 00-07,70-79 are used, plus 08 on up are used to store matrix elements.

For example, consider the system used before, x + 2y = 1, 3x + y = 0

Enter	Press	Display/ Printed	Comments
2	A	3	Program requires one extra row for manipulations.
3	B	3.03	
1	C	0	First row data.
1	D	1	
2	D	2	
1	D	1	
2	C	0	Second row data.
3	D	3	
1	D	1	
0	D	0	
	E		Requires about 20 seconds.



As before x = –0.2 = –1/5
y = 0.6 = 3/5

000	76	LBL	080	77	77	160	85	+	240	95	=
001	11	A	081	42	STO	161	43	RCL	241	36	PGM
002	85	+	082	79	79	162	78	78	242	07	07
003	01	1	083	01	1	163	55	÷	243	11	A
004	95	=	084	44	SUM	164	01	1	244	43	RCL
005	42	STO	085	79	79	165	00	0	245	79	79
006	75	75	086	43	RCL	166	00	0	246	36	PGM
007	91	R/S	087	75	75	167	95	=	247	07	07
008	76	LBL	088	32	X:T	168	94	+/-	248	12	B
009	12	B	089	43	RCL	169	36	PGM	249	36	PGM
010	42	STO	090	79	79	170	07	07	250	07	07
011	76	76	091	77	GE	171	18	C'	251	71	SBR
012	55	÷	092	00	00	172	36	PGM	252	75	-
013	01	1	093	58	58	173	07	07	253	43	RCL
014	00	0	094	29	CP	174	16	A'	254	75	75
015	00	0	095	43	RCL	175	43	RCL	255	75	-
016	85	+	096	79	79	176	77	77	256	01	1
017	43	RCL	097	85	+	177	36	PGM	257	95	=
018	75	75	098	43	RCL	178	07	07	258	32	X:T
019	95	=	099	78	78	179	11	A	259	43	RCL
020	36	PGM	100	55	÷	180	36	PGM	260	79	79
021	07	07	101	01	1	181	07	07	261	22	INV
022	17	B'	102	00	0	182	71	SBR	262	77	GE
023	86	STF	103	00	0	183	55	÷	263	01	01
024	00	00	104	95	=	184	00	0	264	87	87
025	91	R/S	105	94	+/-	185	42	STO	265	43	RCL
026	76	LBL	106	36	PGM	186	79	79	266	77	77
027	13	C	107	07	07	187	01	1	267	77	GE
028	36	PGM	108	18	C'	188	44	SUM	268	02	02
029	07	07	109	67	EQ	189	79	79	269	79	79
030	18	C'	110	00	00	190	43	RCL	270	43	RCL
031	91	R/S	111	83	83	191	77	77	271	76	76
032	76	LBL	112	01	1	192	32	X:T	272	32	X:T
033	14	D	113	36	PGM	193	43	RCL	273	43	RCL
034	36	PGM	114	07	07	194	79	79	274	78	78
035	07	07	115	16	A'	195	67	EQ	275	22	INV
036	13	C	116	43	RCL	196	02	02	276	77	GE
037	91	R/S	117	79	79	197	53	53	277	00	00
038	76	LBL	118	36	PGM	198	85	+	278	55	55
039	15	E	119	07	07	199	43	RCL	279	43	RCL
040	43	RCL	120	11	A	200	78	78	280	72	72
041	70	70	121	43	RCL	201	55	÷	281	69	OP
042	69	OP	122	75	75	202	01	1	282	01	01
043	01	01	123	36	PGM	203	00	0	283	43	RCL
044	43	RCL	124	07	07	204	00	0	284	73	73
045	71	71	125	12	B	205	95	=	285	69	OP
046	71	SBR	126	36	PGM	206	29	CP	286	02	02
047	02	02	127	07	07	207	67	EQ	287	43	RCL
048	85	85	128	71	SBR	208	02	02	288	74	74
049	29	CP	129	65	×	209	53	53	289	69	OP
050	00	0	130	43	RCL	210	94	+/-	290	03	03
051	42	STO	131	77	77	211	36	PGM	291	00	0
052	77	77	132	36	PGM	212	07	07	292	69	OP
053	42	STO	133	07	07	213	18	C'	293	04	04
054	78	78	134	11	A	214	36	PGM	294	69	OP
055	01	1	135	43	RCL	215	07	07	295	05	05
056	44	SUM	136	79	79	216	16	A'	296	00	0
057	77	77	137	36	PGM	217	43	RCL	297	42	STO
058	01	1	138	07	07	218	77	77	298	79	79
059	44	SUM	139	12	B	219	36	PGM	299	43	RCL
060	78	78	140	36	PGM	220	07	07	300	75	75
061	43	RCL	141	07	07	221	11	A	301	32	X:T
062	77	77	142	71	SBR	222	43	RCL	302	01	1
063	85	+	143	65	×	223	75	75	303	44	SUM
064	43	RCL	144	43	RCL	224	36	PGM	304	79	79
065	78	78	145	75	75	225	07	07	305	43	RCL
066	55	÷	146	36	PGM	226	12	B	306	79	79
067	01	1	147	07	07	227	36	PGM	307	77	GE
068	00	0	148	11	A	228	07	07	308	03	03
069	00	0	149	43	RCL	229	71	SBR	309	17	17
070	95	=	150	77	77	230	65	×	310	94	+/-
071	94	+/-	151	36	PGM	231	43	RCL	311	36	PGM
072	36	PGM	152	07	07	232	79	79	312	07	07
073	07	07	153	12	B	233	85	+	313	12	B
074	18	C'	154	36	PGM	234	43	RCL	314	61	GTO
075	22	INV	155	07	07	235	75	75	315	02	02
076	67	EQ	156	71	SBR	236	55	÷	316	99	99
077	01	01	157	65	×	237	01	1	317	98	ADV
078	58	58	158	43	RCL	238	00	0	318	98	ADV
079	43	RCL	159	77	77	239	00	0	319	92	RTN

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. Please limit your submissions to four double-spaced typed pages, and forward them to:

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

Précis

Editor's Note: *This column presents the new PPX programs which are now available. The abstracts here are from programs that the Analysts thought would be of special interest to members. 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 "Programming Corner" column.*

048006G Advertising R/F: C2 Finder
Given population, average, issue audience, and the audience to N issues, this program will find the proper two-issue cummulative audience needed for beta-based reach/frequency formula input.
Jeremy D. Sprague, New York, NY
355 Steps, PC-100A

078013G Whole Life-Term Rider Mixture
This program calculates the amounts of whole life insurance and term rider which will simultaneously add up to the total amount of insurance needed and will cost an amount the proposed insured has agreed he can budget.
Edward Gold, East Meadow, NY
54 Steps

148013G Checking Account Management
Provides a more complete and compact record of account activity than does a bank statement. Once a month, user enters deposits, withdrawals, amount and number of each check written, and numbers of checks returned. A printed record of all transactions is completed by a list of checks outstanding and checkbook and account balances. The resulting condition of the account is recorded on magnetic cards for later use.
George Vogel, Newton Highlands, MA
384 Steps, PC-100A

168002G Short Term Leasing
Computes the monthly rental payments of an object for 1, 2, or 3 years. Program takes into account the selling price, repair costs, the prime loan rate, and the number of years article is leased.
Rober A. Owen, Miliptas, CA
67 Steps, Mod 1

188035G Simulated Stock Market Speculation
The simulation uses the random number generator to simulate the results of stock market speculation with random price movements when using the following trading rules: (1) Buy 100 shares when the price rises above a stated value; (2) Sell 100 shares when the price falls below a stated value. The simulation flashes the change in price, and the value of both the stock account and the cash account as each trading cycle is completed. When either the stock portfolio or the cash account falls below a minimum necessary to trade 100 shares, the program concludes with a summary profit and loss report.
Ed B. Flowers, Forest Hills, NY
350 Steps, Mod 2

208060G Least Squares Polynomial Fitting
This program will fit a general polynomial expression to a set of (x, y) points, x = 0, 1, 2, ... N, for polynomials up to the 11th degree. Orthogonal polynomials are constructed for the argument set x.
Marvin Methven, Glendale, CA
436 Steps

218043G Randomized Blocks ANOVA
This program computes the sum of squares (SS), mean squares (MS), F ratios, and degrees of freedom necessary to compile a complete analysis of variance table. The F ratios may also be accessed directly (with their resepctive degrees of freedom) without displaying SS or MS. Similarly, MS can be accessed without displaying SS. Accepts up to 48 treatment groups and an unlimited number of subjects (observations).
David L. Turock, Montclair, NJ
383 Steps

268039G Random Number/Sample Generator
This program generates Random Numbers from the Normal, Uniform Exponential, Arcsin, Rayleigh, Logistic, LaPlace, Cauchy, or Log Normal distribution. Numbers may be generated individually; or in a sample of any desired size in which case the sample size, mean, variance, and high and low observations are also returned. Scaling factors may also be input to adjust the distribution.
Tom Chambers, San Francisco, CA
480 Steps, Mod 1

328011G Cooley-Tukey Fourier Transform
Performs an 8, 16 or 32 point Fast Fourier Transform using an algorithm based on a Fortran Algorithm by Cooley and Tukey. Input may be complex time data. Output is the complex forward or inverse transform.
Robert E. Harrison, Rancho Palos Verdes, CA
318 Steps, Mod 1

398194G Properties of Circles
Given any one of radius, diameter, circumference, area, side of inscribed square, side of equiperipheral square, sphere volume, or sphere surface area, program computes the rest. Additionally, given any one of chord, arc, central angle, rise of segment, or area of sector, program computes the rest and area of minor segment.
Ken Stoops, Anchorage, AK
743 Steps, PC-100A

398195G General Complex Equation Solver
Solves any equation for real or complex roots. The program is a complex implementation of Newton's procedure which differentiates, iterates, and solves the user-entered equation automatically.
Dix Fulton, Seattle, WA
118 Steps, PC-100A

Précis

398196G Addition of Fractions
Adds (or subtracts) any number of fractions and gives the resulting fraction in lowest terms. Also calculates the greatest common divisor of two numbers.
George Vogel, Newton Highlands, MA
105 Steps

418096G Ideal Gas Law
This program uses the ideal gas equation (PV=nRT) and a precise value of R to accurately calculate the unknown component of the equation. It also uses a precise conversion from Celius to Kelvin and vise versa. Given the number of grams of a substance at certain conditions the molecular (atomic) weight is calculated. Given the molecular weight at any conditions, program calculates the density.
Ronald J. Williams, Silver Spring, MD
406 Steps

468011G True Dip and Azimuth Angle
Program generates true dip and Azimuth Angle. Inputs required are two apparent dip angles and their included Azimuth Angle. Strike can be easily calculated from Azimuth Angle Solution.
Thomas J. Oliver, Dewey, AZ
113 Steps, Mod 3

508057G Clinical Decisions: Treatment Threshold
Decision analysis can be used to calculate a therapeutic threshold if the utility of treating or not retreating in the presence or absence of a particular disease is known. A case of possible appendicitis is used to illustrate the method.
Hershel Goren, Cleveland, OH
116 Steps, PC-100A

568008G Pharmacokinetics of Phenytoin
This program calculates best-fit values of Vmax and Km from input data that consists of at least two pairs of steady-state plasma concentration (Cpss) versus dose (Ro) data. Once these parameters have been determined, predictions of required Ro to produce a desired Cpss, or the expected Cpss from a selected Ro are readily made by this program.
John P. Toscano, Minneapolis, MN
317 Steps, Mod 1 or 2

628156G Concrete Mix Design (Absolute Volume)
Computes batch weights for a one cubic yard batch of concrete based on the absolute volume occupied by aggregates. Provisions are included for using either Type I or Type II cement and up to three aggregates.
Ronald E. Sherard, Hays, KS
216 Steps

638028G Radix-50 Packed Word Decoder
Decodes the Digital Equipment Corporation 16-bit Radix-50 packed word into the three characters which are encoded.
William Kantrowitz, Brookline, MA
125 Steps, PC-100A

708004G Required Coverage Angles
Calculates the required vertical and horizontal angles of coverage at the front row and at the last row required of a point source cluster of speakers given six room measurements. Allows change in any input parameters for examination of impact on speaker cluster.
Frank G. Nagy, Akron, OH
384 Steps

728002G Rational Analysis of Clays
Converts the oxide analysis of clays to typical ceramic minerals with the other body components. This enables a closer analysis of the properties of the resultant mixture, including the fired properties.
Charles F. Hanks, Alliance, OH
206 Steps, PC-100A

868017G Well Log Analysis
Computes porosity (for specified lithology) from log bulk density, apparent limestone density porosity, apparent limestone neutron porosity, and/or sonic travel time. Also computes: porosity and apparent lithology from FDC-CNL crossplot; formation water resistivity from known measured value at any temperature, from formation water salinity, or from the log self potential curve; water saturation from log deep resistivity curve and any previously computed porosity; and Schlumberger's "Productivity Ration Index" (PRI).
M. J. Crose, Midland, TX
709 Steps

908174G TI-57 Program Listing
Lists a TI-57 program, with alphanumeric keycode descriptors. Handles all merged codes.
David Kantrowitz, Brookline, MA
106 Steps, PC-100A

908175G Memory Malfunction Diagnostic
This diagnostic program tests memory by storing a test value in memory registers R01 through R99, summing the negative of the test value into each memory register, and checking that the recalled result from each register is zero.
Palmer O. Hanson, Largo, FL
160 Steps, PC-100A

918212G Thaipan
This game is based on trading, taking place in Hong Kong, around the turn of the century. Your name is Thaipan, a famous trader, with a big debt. You trade four items: general cargo, arms, silk, and opium. You go from store to store, accumulating money and debt, while trying pay off master money lender Wu and escape death.
Andrew Klausman, Lincolndale, NY
640 Steps, PC-100A, Mod 1

918213G Pick A Number
In this game both the user and the program pick a positive integer. The player with the lower number scores 1 point, unless the difference between the two numbers is one; in which case the player with the higher number scores. The program plays with its optimum strategy and is challenging to beat. The first player with 10 points wins the game.
Michael Carlton, Covina, CA
203 Steps, Mod 1

from the Analyst's Desk

• If you repartition your calculator during a stage of executing your program (either externally via keyboard or internally via program execution), Paul Sperry of Boulder, Colorado, sends a warning: If you're not careful, you can lose the program pointer during repartitioning. For example, if the program stops at step 460 and you subsequently repartition to 399.69 (7 OP 17), the program pointer gets "stuck" inside what becomes data memory, the R/S key will not operate and the LRN key will not work.

To study this phenomenon, without going into LRN mode, press GTO 460 7 OP 17. Pressing R/S results in a flashing 0 display. Press CLR LRN. The LRN key will not function.

The solution to this problem is to first get the pointer back by pressing RST. Then study your program to find and correct the error.

MEMBERSHIP RENEWALS

Is 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.

A renewal card and reminder will be sent to each member in ample time to renew. Return the card to PPX with your check or money order for $18. Be sure to include your membership number on both your card and your check.

Membership Number	Renewal Due:
100001-101982	December 15
909707-910537	October 15
910538-911277	November 15
911278-912141	December 15
920479-921076	October 15
921077-921780	November 15
921781-922653	December 15

TI-59 Programming Seminar

A Texas Instruments Programming Seminar may be coming to your area. These seminars will provide beginning and intermediate programming training on the TI-59. Classes consist of two 8-hour days of hands-on training that begin at 8:30 A.M. and last until 4:30 P.M. A luncheon will be served daily. You must provide your own TI-59 and it is highly recommended that you also bring your PC-100 A/C Printer. As a service to our members, we are including the schedule below for dates and locations.

SEMINAR DATES	LOCATION
November 13/14	Detroit, MI
November 17/18	Cleveland, OH
December 4/5	Orlando, FL
December 11/12	Burlingame, CA

To attend, send your name, mailing address, and telephone number to: TI-59 Seminar; P.O. Box 10508, M/S 5873; Lubbock, Texas 79408. Also include your check or money order for the tuition fee of $150 per person (purchase orders are not accepted). Those members that want to attend need to respond as quickly as possible as there is a limit of 50 participants per class.

ADDRESS CHANGES

In order to ensure uninterrupted service, please submit address changes to PPX at lease 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

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

ADDRESS CORRECTION REQUESTED

BULK RATE
U.S. POSTAGE
PAID
Permit No. 1
Lubbock, Texas