PPX EXCHANGE

Vol. 3 Number 6 Copyright 1979 November 1979

PPX POTPOURRI

1. **Attention new members —** PPX continues to offer the Sourcebook for Programmable Calculators at a sale price of $9.95 (compared to $12.95) through the end of January. The sourcebook contains programs written on a college level that cover a wide range of disciplines from Biomedical Engineering to Music Theory. To order a copy, enter "Sourcebook" on your PPX-59 order form and include your check or money order (only) for $9.95, plus $1.00 postage and handling, and applicable sales tax. Order now as this offer ends January 31, 1980. This offer is good only in the U.S., Canada, and Mexico.

2. **Return to sender —** after the mailing of the July 1979 catalog we found that many members did not receive their addendum. The reason, most often than not, was they had a change of address which was never forwarded to us. Although newsletters are mailed first class and can be forwarded, addendums to the catalog are mailed 3rd class bulk rate, and are therefore returned to PPX stamped "return to sender." As mail cost continue to rise, PPX is making every effort to refrain from passing these costs on to you. But, we need your help . . . we need your new address as soon as you relocate.

For your convenience, starting with this issue, a Change of Address Notice will be included in each future issue of the PPX Exchange. Please return the completed form to us as soon as a new address is known.

3. **TI-59 Specialty Pakettes —** Pakettes continue to provide software for specific disciplines not covered by a TI-59 Solid State Module. Each Pakette contains 5 to 11 programs within, and is closely related to a specific subject area. Each pakette contains programs written by PPX-59 users who defined their own specific program needs and filled these needs by writing programs. At a $10 retail price these pakettes are a true software value. (If purchased separately these programs would cost up to $33). Pakettes do not contain pre-recorded or blank magnetic cards. The PPX-59 Software Catalog (Pages 6-1 and 6-2) lists the program included in each of the following pakettes. Order by writing the pakette name and catalog number on a PPX-59 order form. Enclose $10 for each pakette plus your state's applicable sales tax, and $1.00 to cover postage and handling.

Pakette	Catalog Number
Securities	9B0001
Statistical Testing	9B0002
Civil Engineering	9B0003
Electronic Engineering	9B0004
Blackbody	9B0005
Oil/Gas/Energy	9B0006
Printer Utility	9B0007
Astrology	9B0008
Programming Aids	9B0009
59 Fun	9B0010
3D Graphics	9B0011
Mathematics	9B0012
Fluid Dynamics	9B0013
Lab Chemistry	9B0014
Marketing/Sales	9B0015
Production Planning	9B0016

4. **TI Accessories Reminder —** PPX offers only those accessories listed on page 4-1 of your PPX catalogue. For other consumer/accessories products please write:

Texas Instruments Service Facility
P. O. Box 53
Lubbock, TX. 79408

Fool-Proofing Multiple Program Systems

Donald R. Lambert

Editor's Note: A programming system is a program whose documentation exceeds the general format of PPX-59. The reason that the program documentation is larger than the general format is that the number of program steps usually exceed the 960 steps available on the TI-59 calculator. Programming Systems that have more than 960 steps can be run on a TI-59 by segmenting and running each segment separately. Registers are used to retain the data that is needed between each segment.

Programming Systems often get to be quite large. To keep the size of the system within limits, sometimes a group of similar programs are combined into one system and another group of programs are combined into another (e.g. instead of having one 14-program system, two 7-program systems are written). Such multiple programming systems can pose handling problems to the user. This article addresses two of the most common problems faced when using multiple programming systems and describes how to "fool-proof" against these problems.

Let's try a simple experiment. With the Master Library module in the TI-59, perform the following keystroke sequence: 2nd Pgm, 15, 2nd, OP 09, 2nd, Pgm, 7, 2nd, OP 09. This sequence downloaded program ML-7 on top of ML-15, creating the same effect as when the user of a system accidentally loads cards from the wrong program of the system. In this case ML-7 is still usable, but ML-15 is rendered worse than useless, in that even though it seems to work, it is not doing what we expect it to do!

In using multiple program systems, there is a very real chance that the user might (a) accidentally load cards from more than one program into the calculator or (b) that an attempt would be made to either load a data card before previous data had been recorded, or to load a data card produced by a system not in the calculator (each system has its own data cards).

• The first problem, that of mixed program cards, can be fool-proofed as shown in the following example (in which banks one through three are used to store the program steps).

1. Partition to 3 OP 17 (to obtain banks one through three).

2. Reserve a group of labels that are otherwise unused in any of the programs. These labels will be used as tools in fool-proofing the system. Merged labels can be chosen for this purpose as they are inconvenient for normal use. We will use RC* and ST* as our "fool-proofers." (LBL ST* can be entered by pressing Lbl, SST, BST, STO, IND. RC* can be entered similarly). Each program in the system is to have either different labels or labels used in a different order. (N labels are enough to handle N^2-N programs, where N is at least 2).

3. Select a user defined label to trigger the program checking sequence and place this label in the last bank used to store program steps (consider label A for this example). Next, store the following step sequence in bank 3:

```
480  76 LBL
481  11  A
482  43 RCL
483  99  99
484  29 CP
485  01   1
486  71 SBR
487  72 ST*
488  22 INV
489  67  EQ
490  00  00
491  00  00
492  25 CLR
```

The number 1 after the CP step is defined here as the "program number". Each **program** in the system will be assigned a different program number (1, 2, etc.). RCL 99 (in the above sequence) will cause a flashing display and location 000 will contain a RTN instruction.

4. Store the following steps in memory banks one and two:

Bank One

```
000  92 RTN
001  81 RST
002  76 LBL
003  73 RC-
004  01   1
005  92 RTN
```

Bank Two

```
240  76 LBL
241  72 ST*
242  32 X:T
243  61 GTO
244  73 RC*
```

If the calculator has been correctly loaded, the program will branch from bank three to bank two to bank one, where it will fail to transfer to step 000, and will thus continue to run. If the program number is incorrect between banks one and three, (due to mixed program cards) the program will branch to location 000 in bank one which will cause the calculator to display bank one's program number. RCL 99 will cause the display to flash because the partition doesn't allow 99 registers to be displayed. For maximum speed, all labels should be as near the start of their bank as possible.

• The second problem in fool-proofing multiple program systems concerns improper data card handling. Just as the program cards were purposely given program-to-program incompatibility, each system's data cards must be made incompatible with those of every other system. This can be done as follows:

1. Place an integer and fraction in the form of SS. PP (where SS is the integer system identifier and PP is the fraction program identifier) into register 00. Each program can be easily written to tack its own unique decimal fraction onto the number stored in register 00 just before data is recorded. (hint: use the integer key and t-register when writing these steps). This will solve both the problem of foreign data cards, and the attempt to prematurely load another data card.

2. Write the program so that it will do two things: (a) Load all data under program control; and (b) test register 00 (against the t-register) for the presence of any decimal fraction, (which indicates that the calculator contains recorded data). If there is no fraction the calculator should stop. If a fraction is detected the calculator should ready itself to read a data card. The decimal fraction on the newly loaded data card is then tested against the program's decimal fraction, and the program either removes the decimal fraction and runs, or stops, as appropriate.

By keeping these simple fool-proofing methods in mind, multiple program systems can be written which include user protection against mixed program cards and improper handling of data cards.

PROFESSIONAL TRAINING ON THE TI-59

What is the most efficient way for a professional to learn to use the TI-59? The answer to this question for professionals in large and small corporations has been to use Texas Instruments Professional Productivity Program (TIPPP) seminars. TIPPP has been training professionals for over two years through both user and instructor courses on the TI-59.

The user's course objective is to train up to 50 students per class on the calculator so that they may put it to immediate use. This class includes two 8 hour days of hands-on study covering all features and functions of the TI-59. It is also designed to answer questions about applications and to cover a wide variety of software already written in professional areas (PPX programs and pakettes are used as examples for applications of the TI-59).

The second approach is to train up to 10 students per class to become programmable calculator instructors. This "Instructor's Course" takes a full week of continuous instruction, participation and programming. The ability to not only write but to document a complex program is among the objectives here. This is done by using PPX submission forms as a standard for documentation. These lessons enable the students to start up TI-59 program libraries within their own companies.

In addition to formal classes, TIPPP also can provide a training package comprised of the materials used by TI instructors. These materials enable a user to cover, at his own pace, the same information taught in the users course. The training package includes three video tapes, one hundred and sixty 35mm slides, an instructor's guide and TI-59 workbooks (available for purchase in quantities needed to conduct a training program). This package offers an excellent tool for use within a company's learning center.

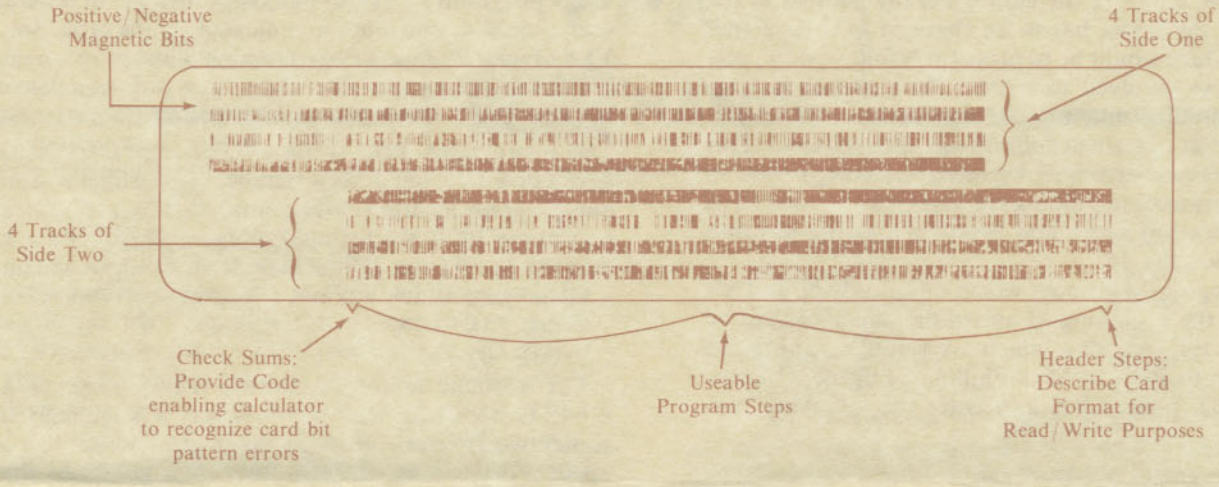
The following suggested retail prices pertain to training programs and materials:

User Course	$1200
Instructor Course	$2400
Self-pace training materials	$ 650

For more information on TIPPP please write or call:

TIPPP, Texas Instruments Inc.
ATTN: John Cable
P. O. Box 10508, MS 5873
Lubbock, Texas 79408
(806) 741-3442

TI-59 MAGNETIC CARD
(actual reproduction)



A LITTLE 'BIT' ABOUT YOUR MAGNETIC CARD

As you know, the TI-59 is different from most programmable calculators in that it provides a means of permanent program and data storage. Did you ever wonder what your TI-59 sees when reading a card? Using a four-track magnetic read head it sees (senses) magnetic bits or blips which were earlier placed on the card in a specific pattern. By controlling whether these bits are positive or negative, data can be represented. This method of data representation is based on the "binary" numbering system.

Although we use the decimal (or base 10 system) on a daily basis, we should recognize that other based systems do exist. There is for example base 8, octal; base 16, hexadecimal; and base 2, or binary, which is used by the TI-59 card reader/writer.

The number 17 (17 in base 10) can be expressed as:

$1 \times 10^1 + 7 \times 10^0$.

But, the number 17 in base 2 (or binary) is 10001, which can be expressed as:

$1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$

Since there are only two numerals in this system (0 and 1), every digit can be used to reflect the status of a switch (on or off). Each digit of a binary number is called a bit (binary digit). The presence of a positive or negative bit on a magnetic card represents data (see the figure above). The calculator is able to translate data back and forth from key code to binary. Thus by reading or writing a specific pattern of bits on a card, your TI-59 is able to save both the program and data.

PPX-59 PROGRAMMING CORNER

This column is devoted to PPX-59 programming suggestions. If you have a program(s) that you would like to see made available through PPX-59, send your suggestions to PPX. In this way, members who enjoy programming are made aware of your programming needs. PPX-59 is not staffed to do custom programming; therefore, member suggested programs will become available only if another member of PPX-59 comes to the rescue.

Our members would like to see:

- A program that would generate a hemoglobin dissociation curve given the saturation and partial pressure of oxygen (for both venous and arterial blood).
- Civil Engineering program that would determine the influence lines of beams with a variable number of spans, variable moment of inertia, and loaded with uniform and concentrated loads or concentrated moving loads.
- Practical machine shop programs for generation of geometric shapes (e.g., circle) on milling machines, lathes and shapers (where numeric control machines are not available).
- A program that would perform the design calculations for screws with special threads according to federal standard H28-section 3.

Our mistake! — In September, our PPX-59 Programming Corner requested that a program be written to give lagged cross correlations between x(t) and y(t) for lags of +/- 6 or more. PPX member, Gerald Fiester, has informed us that his program, "Lagged Regression for Time Series," (PPX #208032D), would perform exactly as the one requested.

MEMBER # ☐☐☐☐☐☐☐☐☐

NOTICE OF CHANGE OF ADDRESS

In order to ensure uninterrupted service, please submit change of address at least six weeks prior to change. Please mail to:

Texas Instruments, Inc.
PPX Department
P. O. Box 53
Lubbock, TX 79408

NAME: _____

OLD ADDRESS: _____

_____

NEW ADDRESS: _____

_____

EFFECTIVE DATE: _____

PHONE: _____

MEMORY FLASHCARD

A challenging memory game that allows the player to test his ability to recall in sequence up to 10 rapidly flashing numbers (up to 10 digits in length). Program uses the master library.

PPX wishes to thank David E. Powers, for his excellent program.

User Instructions:

1. Enter Program
2. Enter seed (0-199017), press E'.
3. Enter number of flashes (1-10), press A
4. Enter maximum number of digits (1-10), press B
5. Press C to begin flash sequence.
6. Enter guess for sequence, pressing R/S after each number.
7. Press E to display score for round.*
8. Press A' to display mean score, or press 1 to guess again (repeat step 6), or press C to review sequence.
9. Press B' to determine cumulative score.
10. Press C' to determine mean score.

*See note under example problem.

Example:

Enter	Press	Display	Comments
2541	E'	0	Enter seed
3	A	3	Select number of flashes
3	B	3	Select max. digits per flash
	C	85	Sequence flashed
		456	
		516	
		0	
85	R/S	85	1st number in sequence
456	R/S	456	2nd number in sequence
516	R/S	516	3rd number in sequence
			All correct
	E	9	Score *

Note: The score is based upon the total number of flashes multiplied times the number of digits. One point is deducted for each incorrect digit guessed. Based on this ground rule, when the display flashed '85', it was recognized as a '085' for scoring purposes (i.e. 3 digits).

000	76	LBL	024	76	LBL	048	87	EQ	072	44	SUM	096	53	(	120	11	B	144	12	12	168	59	INT	192	31	31	216	36	PGM
001	13	C	025	14	D	049	00	00	073	14	D	097	43	RCL	121	59	INT	145	32	X⇄T	169	72	ST*	193	53	(	217	15	15
002	43	RCL	026	92	RTN	050	56	56	074	43	RCL	098	12	12	122	42	STO	146	01	1	170	14	14	194	43	RCL	218	11	A
003	25	25	027	42	STO	051	86	STF	075	27	27	099	65	×	123	00	00	147	01	1	171	01	1	195	30	30	219	05	5
004	42	STO	028	27	27	052	01	01	076	61	GTO	100	53	(	124	42	STO	148	32	X⇄T	172	44	SUM	196	55	÷	220	36	PGM
005	00	00	029	43	RCL	053	01	1	077	00	00	101	43	RCL	125	25	25	149	77	GE	173	14	14	197	43	RCL	221	15	15
006	01	1	030	25	25	054	44	SUM	078	45	45	102	36	36	126	32	X⇄T	150	02	02	174	37	DSZ	198	31	31	222	12	B
007	05	5	031	42	STO	055	25	25	079	43	RCL	103	75	-	127	01	1	151	25	25	175	00	00	199	54	)	223	25	CLR
008	43	STO	032	00	00	056	37	DSZ	080	28	28	104	01	1	128	01	1	152	36	PRM	176	01	01	200	42	STO	224	92	RTN
009	14	14	033	01	1	057	00	00	081	32	X⇄T	105	54	)	129	32	X⇄T	153	15	F	177	52	52	201	32	32	225	53	(
010	73	RC*	034	05	5	058	00	00	082	84	+/-	106	54	)	130	77	GE	154	16	C'	178	43	RCL	202	92	RTN	226	01	1
011	14	14	035	42	STO	059	64	64	083	34	FX	107	75	-	131	02	02	155	50	INT	179	12	12	203	76	LBL	227	65	×
012	66	PAU	036	14	14	060	87	IFF	084	92	RTN	108	53	(	132	25	25	156	32	X⇄T	180	92	RTN	204	17	B'	228	00	0
013	01	1	037	25	CLR	061	01	01	085	76	LBL	109	43	RCL	133	01	1	157	73	INT	181	76	LBL	205	43	RCL	229	54	)
014	44	SUM	038	42	STO	062	00	00	086	15	E	110	00	0	134	42	STO	158	42	STO	182	14	A'	206	30	30	230	92	RTN
015	14	14	039	28	28	063	74	R/S	087	53	(	111	65	×	135	42	STO	159	13	13	183	25	CLR	207	92	RTN	231	76	LBL
016	97	DSZ	040	22	INV	064	25	CLR	088	53	(	112	43	RCL	136	14	14	160	43	RCL	184	42	STO	208	76	LBL	232	18	C'
017	00	00	041	86	STF	065	29	CP	089	43	RCL	113	12	12	137	43	RCL	161	12	12	185	26	26	209	10	E'	233	43	RCL
018	00	00	042	01	01	066	43	RCL	090	25	25	114	54	)	138	00	00	162	22	INV	186	43	RCL	210	47	CMS	234	32	32
019	10	10	043	43	RCL	067	27	27	091	65	×	115	54	)	139	92	RTN	163	28	LOG	187	29	29	211	29	CP	235	92	RTN
020	01	1	044	27	27	068	92	RTN	092	43	RCL	116	42	STO	140	76	LBL	164	49	PRD	188	44	SUM	212	36	PGM			
021	44	SUM	045	32	X⇄T	069	42	STO	093	12	12	117	29	29	141	12	B	165	13	13	189	30	30	213	15	15			
022	26	26	046	73	RC*	070	27	27	094	54	)	118	92	RTN	142	59	INT	166	43	RCL	190	01	1	214	15	E			
023	25	CLR	047	14	14	071	01	1	095	75	-	119	76	LBL	143	42	STO	167	13	13	191	44	SUM	215	05	5			

THE ADDITION OF FRACTIONS

Adding and subtracting fractions can cause anybody trouble (most often parents checking their children's homework). For those who are having this problem and want something done about it, PPX comes to your aid with — "The Addition of Fractions." This program allows the user to add or subtract any two fractions and receive the answer in the lowest terms. All operations performed with this program are additions. Therefore, to do subtraction, enter +/- for the appropriate numerator. The negative sign always has to be entered with the numerator. If a PC-100A/C is attached, the outputs will be printed and labeled.

PPX wishes to thank the author of "The Addition of Fractions," Ron Jorgensen, for his excellent program.

User Instructions:

1. Enter Program
2. Clear memories and t-register by pressing CLR, CMs, x ⇄ t.
3. Enter the first fraction using the following two steps:
   Enter numerator, press A.
   Enter denominator, press B.
   Note: Display flashes when a zero denominator is entered.
4. Enter the second fraction by first entering the numerator and then the denominator and pressing C and D, respectively.
5. The answer's numerator will be displayed after step 4 (above) is completed. Obtain the final denominator by pressing E.

Example: 2/3 - 4/5 = ? Answer: -2/15

Enter	Press	Display	Comments
	CLR	0	
	CMs		Clear memories and t-register.
	x ⇄ t		The previous value of the t-register will be displayed.
2	A	2.	Numerator #1
3	B	3.	Denominator #1
4+/-	C	-4.	Numerator #2
5	D	5.	Denominator #2
		-2.	Final Numerator
	E	15.	Final Denominator

TI-59 LISTING FOR "THE ADDITION OF FRACTIONS"

000	76	LBL	076	69	OP	152	39	COS	228	01	1
001	11	A	077	04	04	153	77	GE	229	00	0
002	42	STO	078	43	RCL	154	38	SIN	230	00	0
003	10	10	079	02	02	155	85	+	231	69	OP
004	25	CLR	080	69	OP	156	43	RCL	232	04	04
005	03	3	081	06	06	157	00	00	233	43	RCL
006	01	1	082	98	ADV	158	95	=	234	15	15
007	04	4	083	98	ADV	159	48	EXC	235	69	OP
008	01	1	084	76	LBL	160	00	00	236	06	06
009	03	3	085	60	DEG	161	61	GTO	237	98	ADV
010	00	0	086	43	RCL	162	38	SIN	238	98	ADV
011	00	0	087	01	01	163	76	LBL	239	91	R/S
012	02	2	088	42	STO	164	39	COS	240	76	LBL
013	69	OP	089	00	00	165	43	RCL	241	35	1/X
014	04	04	090	43	RCL	166	00	00	242	94	+/-
015	43	RCL	091	02	02	167	18	C'	243	42	STO
016	10	10	092	61	GTO	168	76	LBL	244	12	12
017	69	OP	093	38	SIN	169	70	RAD	245	42	STO
018	06	06	094	76	LBL	170	75	-	246	13	13
019	91	R/S	095	18	C'	171	43	RCL	247	43	RCL
020	76	LBL	096	43	RCL	172	13	13	248	04	04
021	12	B	097	01	01	173	95	=	249	61	GTO
022	42	STO	098	55	÷	174	67	EQ	250	34	FX
023	01	01	099	43	RCL	175	80	GRD	251	76	LBL
024	25	CLR	100	00	00	176	77	GE	252	34	FX
025	01	1	101	65	×	177	70	RAD	253	75	-
026	06	6	102	43	RCL	178	85	+	254	43	RCL
027	01	1	103	02	02	179	43	RCL	255	13	13
028	07	7	104	95	=	180	13	13	256	95	=
029	03	3	105	42	STO	181	95	=	257	67	EQ
030	01	1	106	04	04	182	48	EXC	258	45	YX
031	00	0	107	61	GTO	183	13	13	259	77	GE
032	02	2	108	30	TAN	184	61	GTO	260	34	FX
033	69	OP	109	76	LBL	185	70	RAD	261	85	+
034	04	04	110	30	TAN	186	76	LBL	262	43	RCL
035	43	RCL	111	43	RCL	187	80	GRD	263	13	13
036	01	01	112	04	04	188	43	RCL	264	95	=
037	69	OP	113	55	÷	189	12	12	265	48	EXC
038	06	06	114	43	RCL	190	55	÷	266	13	13
039	98	ADV	115	01	01	191	43	RCL	267	61	GTO
040	91	R/S	116	65	×	192	13	13	268	34	FX
041	76	LBL	117	43	RCL	193	95	=	269	76	LBL
042	13	C	118	10	10	194	42	STO	270	45	YX
043	42	STO	119	95	=	195	14	14	271	43	RCL
044	11	11	120	42	STO	196	25	CLR	272	12	12
045	25	CLR	121	12	12	197	03	3	273	55	÷
046	03	3	122	25	CLR	198	01	1	274	43	RCL
047	01	1	123	43	RCL	199	04	4	275	13	13
048	04	4	124	04	04	200	01	1	276	95	=
049	01	1	125	55	÷	201	03	3	277	94	+/-
050	03	3	126	43	RCL	202	00	0	278	42	STO
051	00	0	127	02	02	203	00	0	279	16	16
052	00	0	128	65	×	204	00	0	280	25	CLR
053	03	3	129	43	RCL	205	69	OP	281	03	3
054	69	OP	130	11	11	206	04	04	282	01	1
055	04	04	131	95	=	207	43	RCL	283	04	4
056	43	RCL	132	44	SUM	208	14	14	284	01	1
057	11	11	133	12	12	209	69	OP	285	03	3
058	69	OP	134	43	RCL	210	06	06	286	00	0
059	06	06	135	12	12	211	91	R/S	287	00	0
060	91	R/S	136	22	INV	212	76	LBL	288	00	0
061	76	LBL	137	77	GE	213	15	E	289	69	OP
062	14	D	138	35	1/X	214	43	RCL	290	04	04
063	67	EQ	139	42	STO	215	04	04	291	43	RCL
064	10	E'	140	13	13	216	55	÷	292	16	16
065	42	STO	141	43	RCL	217	43	RCL	293	69	OP
066	02	02	142	04	04	218	13	13	294	06	06
067	25	CLR	143	61	GTO	219	95	=	295	91	R/S
068	01	1	144	70	RAD	220	42	STO	296	76	LBL
069	06	6	145	76	LBL	221	15	15	297	10	E'
070	01	1	146	38	SIN	222	25	CLR	298	35	1/X
071	07	7	147	75	-	223	01	1	299	00	0
072	03	3	148	43	RCL	224	06	6	300	91	R/S
073	01	1	149	00	00	225	01	1			
074	00	0	150	95	=	226	07	7			
075	03	3	151	67	EQ	227	03	3			

PROGRAMMING POINT OF VIEW-REVISTED

A short "TI-59 evaluator test," written by Maurice Swinnen and his programming group, appeared in the March 1979 issue of PPX Exchange. The test was based on the premise that individuals solve programming problems differently depending upon their profession and programming experience. To discover the readers' programming point of view the reader was asked to solve the following problem, and then turn to a page inside the March newsletter to determine his 'programming' point of view.

> The display contains either a 1 or a 2. Write a program that leaves a 1 in the display if the previous content was a 2, and vice-versa.

Since then Maurice Swinnen received the following additional "solutions" from readers:

From professional computer analyst:

000	76	LBL
001	11	A
002	42	STO
003	00	00
004	73	RC*
005	00	00
006	92	RTN
2.		01
1.		02

An OP 10 nut came up with this one:

000	76	LBL
001	11	A
002	32	X:T
003	02	2
004	22	INV
005	67	EQ
006	00	00
007	10	10
008	69	OP
009	10	10
010	92	RTN

and finally, a statistician took the cake with this one:

000	76	LBL
001	11	A
002	47	CMS
003	29	CP
004	78	Σ+
005	78	Σ+
006	73	RC*
007	02	02
008	92	RTN
009	10	E'
010	92	RTN

FROM THE ANALYST'S DESK

• PPX member John F. Szablya of Pullman, Washington, has passed along a procedure that is helpful when editing transfer addresses.

This procedure may be confusing (and dangerous) to beginning programmers because it is a "shortcut." It uses a method that is not explained in the Personal Programming Manual. It is not our purpose to explain why it works but to provide a method for editing that could save the user some trouble.

By closely adhering to the procedure below, a minimum number of keystrokes will have to be pressed when editing:

1. Find the keystroke whose code would correct the address (refer to the key code table on page V-50 of Your Personal Programming Manual). If the code is not found in the table, you must resort to the normal method of editing.

2. Go to the location in programming memory at which the correction needs to take place.
3. Press the keystroke.

For example, consider the case when the address of the instruction, if flg 2 346, has to be changed to if flg 2 365. Normal procedure would be to press LRN followed by pressing all six keys involved to initiate the change (i.e. if flg 2 365). But note that only one location had to be changed. The location that contained code 46 was changed to 65. This change could have been done by referring to page V-50 in Personal Programming, finding the keystroke of code 65 to be x (multiplication sign), and pressing x at the location which contained the 46.

- Two program sequences which differ only by an INV instruction at the beginning of one of the sequences can be split using a label. This idea, which saves duplicate program steps, was sent to PPX by Don Phillip of Eugene, Oregon.

For example, press LRN and enter the following keystrokes:

```
000  76 LBL
001  11  A
002  22 INV
003  76 LBL
004  12  B
005  23 LNX
006  91 R/S
```

Enter 5, press A, the display will contain the antilogarithm of the natural log (i.e., 148.4131591). If you enter 5 and press B, the display will contain the natural logarithm of 5 (i.e., 1.609437912).

However, the INV condition will not hold during a label search. Therefore, if INV is pressed from the keyboard before pressing a user defined key, the INV will be ignored. Try entering 5 and pressing INV then B. You will note that execution ignores the INV instruction. Likewise, if your routine looks like this:

```
007  76 LBL
008  13  C
009  22 INV
010  14  D
011  91 R/S
012  00  0
013  00  0
014  76 LBL
015  14  D
016  23 LNX
017  92 RTN
```

The search for label D will cause the INV instruction to be disregarded.

- PPX has received several questions about keying in TI-59 programs from PC-100A tape listings. In general, the user should key in what is indicated on the tape. **See page VI-6 of your Personal Programming manual** for a complete list of PC-100A tape instructions and their equivalent key strokes. The following are exceptions to the rule:

The instructions Fix, st flg, if flg, and Dsz should be keyed in with a single digit number (0-9) which is stored in one program location. When printed, a 'silent' zero is present in the PC-100A listing. For example, if Fix 2 is keyed into the TI-59 (with printer) the printed output will include a **'silent'** zero which was not keyed in.

```
000  58  FIX
001  02  02
```

The absolute address following the instructions x = t, x ≥ t, GTO, SBR, and Dsz n (where n is the register) occupies two program locations. As an absolute address can be up to 3 digits long, the hundreds digit goes into the first location and the tens and ones digits share the second location. For example, pressing GTO 058 results in the following printout:

```
000  61  GTO
001  00  00
002  58  58
```

The PPX Exchange is published every other month and is the only newsletter published by Texas Instruments for TI-59 owners. You are invited to submit items you feel are of general interest to other TI-59 users. Inputs should be limited to 3 double-spaced typed pages. Please forward your newsletter inputs and any questions to:

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

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