PPX EXCHANGE

Vol. 3 Number 4 Copyright 1979 July 1979

PPX POTPOURRI

1. PPX publishes Software Catalog Addendum. With the publication of the D Addendum, there are now over 1600 programs available through PPX-59. Ten new categories have been utilized: Economics (15), Leasing (16), Geology/Resources (46), Pharmacology (56), Life Sciences (General) (59), Nuclear Engineering (67), Sociology (81), Psychology/Psychiatry (82), Water Resources (88), and Natural Resources (Other) (89). There are still seven categories awaiting submissions by PPX members. Any ideas?

2. Help us better serve you—by making certain that you put a correct catalog number for each program you list on your PPX-59 order form.

3. What is the best source you have for answering questions concerning PPX? Believe it or not, 8 out of 10 questions can be answered by reading the frontal matter of your Catalog. It contains information concerning the catalog's usage as well as instructions on ordering programs and accessories through PPX. For example, what does the **letter** at the end of a program number mean? The answer to this question and many others appear in the frontal matter.

One item that is not completely explained in the frontal matter pertains to superceded programs. When a program is received by PPX **that has advantages** not available in a program already residing in the Catalog, the new program supercedes the first. Therefore, no matter which program you order, you will receive the latest program available on that subject. Don't forget, the best source for quick answers to your questions is at your fingertips — **the frontal matter of your PPX-59 Catalog.**

MEMBERSHIP RENEWALS

Is your membership about to expire? To ensure continued receipt of Addendums, newsletters, and ordering privileges, make certain that this is not the case.

Below is printed the renewal table for members whose one year memberships will soon expire. To find your renewal date, check your membership number against the table shown below. Your membership number corresponds with your original membership date.

Membership number	Must be postmarked by
59908258-59908923	July 15
59908924-59910093	August 15
59910094-59910894	September 15
59910895-59911973	October 15

Members with numbers greater than those listed above will be informed of their renewal dates in a future issue of the PPX Exchange.

A renewal subscription card and reminder will be sent to each member in ample time to renew. The subscription card must be returned with a check or money order for $15. Be sure to include your membership number on both your subscription card and check.

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 program 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:

- Statistical programs for: multiple analysis of variance; multiple analysis of covariance; discriminant analysis; multivariate factor analysis; and canonical correlation analysis.
- Traffic engineering programs in the areas of capacity analysis, warrant studies, signal timing and progression analysis.
- LORAN A or C programs that will free the user from lattice charts.

THE MAXIMIZATION OF DATA PACKING FOR ADVANCED PROGRAMMERS

George Vogel

*Editor's Note: PPXers are always coming up with new ideas for getting the most **into and out of** their TI-59s. It is the **into** with which this article is concerned as it describes two innovative methods of packing large amounts of data into the TI-59's registers. **This article is written with the advanced programmer in mind.***

For handling large amounts of data, a technique known as data packing offers the perfect solution. Data packing is the process of storing more than one number into a register. **Two** very **different** methods of data packing are addressed by this article. **The first method packs data in a format that ensures exact output but necessarily limits the range** of the numbers which may be packed. **The second method ensures output with only a specified accuracy** (e.g., +/- 0.1%) but permits the packing of a wide range of numbers. Although these methods differ widely in the way they pack data, they share one goal in common — the maximation of data packing.

The first method — packing numbers within a specified range while ensuring exact output, treats numbers as base x numbers. X is defined as 1 greater than the absolute value of the algebraic difference between the smallest and the largest number (i.e., the range). For example, suppose that our numbers are: 4, 0, 3, 4, then the smallest number is 0 and the largest number is 4. The absolute difference is equal to 4; therefore, the numbers will be treated as base 5 numbers. In base 5 our numbers would be packed as 4034_5. The base 10 number which corresponds to the base x number (for our example, base 5) is the most economical way to store data in the TI-59's 13 digit registers. Our base 5 number 4034_5 is equivalent to 519_{10} in base 10. (This can be checked using

THE MAXIMIZATION OF DATA PACKING *(cont'd.)*

"Base Conversions", PPX-59 #368002A.) As you will notice, we can now pack our 4 digit base 5 representation of our numbers into a 3 digit base 10 representation thus saving 1 digit of storage space. (Even the largest 4 digit base 5 number, $4444_5=624_{10}$, would only require three digit spaces.) The smaller the range of the numbers, the more numbers can be packed per register. If the range between numbers is 1 (e.g., the numbers are either a 1 or a 2) then 43 numbers can be packed per register. On the other hand, if our numbers fall within a range of 10, then 12 numbers can be packed into each data register. Better packing does not appear possible! (Note, as the range increases, the amount of numbers which can be packed per register decreases. This amount is given by the highest power of base x that does not exceed 10^{13}.)

The following program packs **both positive and negative integer numbers** by substracting the smallest number to be packed from each number entered. The smallest number is then added back to each number during unpacking. This "offset" is the data which is translated from base x to base 10 and stored. Key the following program into LRN mode:

000	76	LBL	033	10	E'	066	74	SM*	099	55	÷
001	11	A	034	43	RCL	067	01	01	100	43	RCL
002	47	CMS	035	03	03	068	97	DSZ	101	05	05
003	25	CLR	036	65	×	069	00	00	102	95	=
004	08	8	037	97	DSZ	070	17	B'	103	59	INT
005	42	STO	038	00	00	071	69	OP	104	65	×
006	01	01	039	10	E'	072	21	21	105	32	X:T
007	01	1	040	01	1	073	61	GTO	106	43	RCL
008	03	3	041	95	=	074	16	A'	107	05	05
009	55	÷	042	42	STO	075	76	LBL	108	95	=
010	53	(	043	04	04	076	12	B	109	22	INV
011	91	R/S	044	76	LBL	077	25	CLR	110	44	SUM
012	42	STO	045	16	A'	078	08	8	111	07	07
013	06	06	046	43	RCL	079	42	STO	112	32	X:T
014	75	-	047	02	02	080	01	01	113	85	+
015	91	R/S	048	42	STO	081	76	LBL	114	43	RCL
016	75	-	049	00	00	082	18	C'	115	06	06
017	01	1	050	89	π	083	43	RCL	116	95	=
018	54	)	051	61	GTO	084	04	04	117	99	PRT
019	94	+/-	052	89	π	085	42	STO	118	43	RCL
020	42	STO	053	76	LBL	086	05	05	119	03	03
021	03	03	054	17	B'	087	43	RCL	120	22	INV
022	28	LOG	055	43	RCL	088	02	02	121	49	PRD
023	95	=	056	03	03	089	42	STO	122	05	05
024	59	INT	057	64	PD*	090	00	00	123	97	DSZ
025	66	PAU	058	01	01	091	73	RC*	124	00	00
026	42	STO	059	76	LBL	092	01	01	125	19	D'
027	02	02	060	89	π	093	42	STO	126	69	OP
028	42	STO	061	91	R/S	094	07	07	127	21	21
029	00	00	062	75	-	095	76	LBL	128	01	1
030	69	OP	063	43	RCL	096	19	D'	129	61	GTO
031	30	30	064	06	06	097	43	RCL	130	18	C'
032	76	LBL	065	95	=	098	07	07			

*Note: If you do not have a printer, replace the Prt instruction at location 117 with R/S.

To use this program: (1) Press A to initialize all registers. (2) Enter the smallest number to be packed, press R/S. (3) Enter the largest number to be packed, press R/S; the amount of numbers which can be packed per register (starting with register 08) will be briefly displayed, and 3.1415... (i.e., π) will be displayed. (4) Enter each number (integers only), press R/S. When 3.1415... appears in the display, the next consecutive register is ready to be packed (until then, the number of the base being used will be displayed). If all numbers have been entered prior to 3.1415... being displayed, you must enter dummy numbers within the range to complete the packing of the register. If an error is made during input, finish packing the register using numbers within the range and press Op 31 0 STO Ind 01 and re-enter the correct numbers for that register. (5) To unpack the registers, press B; all numbers are printed; all data are retained, and can be output again or recorded on magnetic cards for later use. You must press R/S to halt printing. If you are not using a printer, you must press R/S to unpack each number. To start unpacking at a different register than 08, press GTO B SST SST CE, enter the desired register number and press R/S.

To demonstrate use of this program, let's pack numbers which range from 79408 to 79415 inclusive. Since our range is equal to 7 (i.e., 79415 - 79408=7), base 8 is used. Press A, enter 79408, press R/S, enter 79415, and press R/S. The amount of numbers which can be packed per register will be briefly displayed, 14. We will pack the following 14 numbers into register 08 by entering each number and pressing R/S: 79408, 79409, 79410, 79411, 79412, 79413, 79414, 79415, 79414, 79413, 79412, 79411, 79410, and 79409. The number 3.1415... will be displayed to indicate that register 08 is full. Now press B, printed output will look like:

```
79408.
79409.
79410.
79411.
79412.
79413.
79414.
79415.
79414.
79413.
79412.
79411.
79410.
79409.
```

Press R/S to halt printing after the 14 numbers are unpacked from register 08. (If R/S is not pressed, the smallest number will continue to be printed.)

The second method — packs a **wide range** of numbers with only a **specified accuracy** by packing the logarithms of the numbers. Use of three-place logs ensures slide rule accuracy (about +/- 0.1%), adequate for most scientific data. By adding a fourth digit as the characteristic, one can pack logs from 0.000 to 9.999, giving a dynamic range of practically 10^{10}.

The program, which follows, will pack three widely ranging numbers (they may be **positive, negative, fractional, and/or whole)** per data register. The following method is utilized to pack and unpack data beginning with register 05:

(1) Numbers between 0 and 1 have negative logs and numbers greater than 1 have positive logs. The number 5 is added to each log, location 049, (see below keycode listing) and later subtracted (location 217), giving an input range from 10^{-5} to 10^{5}.

(2) Negative numbers and 0 have no logs. Therefore, 0 is stored directly as 0.000 and will be output as the smallest positive value in the range (i.e., 10^{-5}). Negative numbers are transformed into positive numbers before calculating the logs. The extreme left position of the current register will contain information as to whether the first, second and/or third number packed is negative. Initially a 1 is in this position, and a 1, 2, or 4 is added if the first, second, or third number is negative, respectively. This position is decoded at output time and the correct sign is given each number.

(3) Numbers outside of the range are stored and output as limiting values (and should therefore be questioned). Those numbers numerically smaller than 10^{-5} are stored as 10^{-5}, and those numbers numerically

© 2011 Joerg Woerner Datamath Calculator Museum

THE MAXIMIZATION OF DATA PACKING *(cont'd.)*

greater than 10^5 are stored as 9.977×10^4. with the correct sign. As many as 255 numbers can be packed in the 85 registers available. Enter the following steps into LRN mode:

000	76	LBL	060	70	RAD	120	42	STO	180	49	PRD
001	11	A	061	29	CP	121	04	04	181	04	04
002	47	CMS	062	77	GE	122	01	1	182	43	RCL
003	58	FIX	063	89	π	123	03	3	183	04	04
004	03	03	064	25	CLR	124	22	INV	184	59	INT
005	05	5	065	76	LBL	125	28	LOG	185	22	INV
006	42	STO	066	89	π	126	52	EE	186	44	SUM
007	01	01	067	74	SM*	127	22	INV	187	04	04
008	76	LBL	068	01	01	128	52	EE	188	32	X:T
009	38	SIN	069	22	INV	129	22	INV	189	92	RTN
010	01	1	070	87	IFF	130	49	PRD	190	76	LBL
011	42	STO	071	01	01	131	04	04	191	17	B'
012	02	02	072	60	DEG	132	10	E'	192	86	STF
013	42	STO	073	43	RCL	133	02	2	193	01	01
014	03	03	074	02	02	134	67	EQ	194	61	GTO
015	03	3	075	44	SUM	135	17	B'	195	59	INT
016	42	STO	076	03	03	136	04	4	196	76	LBL
017	00	00	077	22	INV	137	67	EQ	197	19	D'
018	76	LBL	078	86	STF	138	17	B'	198	04	4
019	39	COS	079	01	01	139	06	6	199	22	INV
020	91	R/S	080	76	LBL	140	67	EQ	200	28	LOG
021	29	CP	081	60	DEG	141	17	B'	201	49	PRD
022	77	GE	082	43	RCL	142	08	8	202	04	04
023	30	TAN	083	02	02	143	67	EQ	203	53	(
024	50	I×I	084	44	SUM	144	17	B'	204	53	(
025	86	STF	085	02	02	145	76	LBL	205	43	RCL
026	01	01	086	97	DSZ	146	59	INT	206	04	04
027	76	LBL	087	00	00	147	19	D'	207	59	INT
028	30	TAN	088	39	COS	148	03	3	208	22	INV
029	32	X:T	089	53	(	149	67	EQ	209	44	SUM
030	04	4	090	43	RCL	150	16	A'	210	04	04
031	22	INV	091	03	03	151	04	4	211	55	÷
032	28	LOG	092	65	×	152	67	EQ	212	03	3
033	64	PD*	093	09	9	153	16	A'	213	22	INV
034	01	01	094	22	INV	154	07	7	214	28	LOG
035	32	X:T	095	28	LOG	155	67	EQ	215	54	)
036	29	CP	096	52	EE	156	16	A'	216	75	-
037	67	EQ	097	22	INV	157	08	8	217	05	5
038	89	π	098	52	EE	158	67	EQ	218	52	EE
039	53	(	099	54	)	159	16	A'	219	54	)
040	32	X:T	100	74	SM*	160	76	LBL	220	22	INV
041	01	1	101	01	01	161	18	C'	221	28	LOG
042	00	0	102	52	EE	162	19	D'	222	22	INV
043	32	X:T	103	69	OP	163	04	4	223	87	IFF
044	22	INV	104	21	21	164	77	GE	224	01	01
045	52	EE	105	61	GTO	165	80	GRD	225	96	WRT
046	28	LOG	106	38	SIN	166	86	STF	226	94	+/-
047	52	EE	107	76	LBL	167	01	01	227	22	INV
048	85	+	108	12	B	168	76	LBL	228	86	STF
049	05	5	109	25	CLR	169	80	GRD	229	01	01
050	54	)	110	05	5	170	19	D'	230	76	LBL
051	22	INV	111	42	STO	171	69	OP	231	96	WRT
052	77	GE	112	01	01	172	21	21	232	99	PRT
053	70	RAD	113	22	INV	173	61	GTO	233	92	RTN
054	09	9	114	86	STF	174	57	ENG	234	76	LBL
055	93	.	115	01	01	175	76	LBL	235	16	A'
056	09	9	116	76	LBL	176	10	E'	236	86	STF
057	09	9	117	57	ENG	177	04	4	237	01	01
058	09	9	118	73	RC*	178	22	INV	238	61	GTO
059	76	LBL	119	01	01	179	28	LOG	239	18	C'

*Note: To use without a printer, replace Prt (location 232) with R/S.

To use this program: (1) Press A to initialize all registers; 3.000 will signal that register 05 is ready for packing; (2) Enter each number with R/S, if necessary completing the last register used with zeros until 3.000 00 is displayed again; (3) Press B to unpack the registers; all data is retained as in the previous program. Press R/S to halt printing. If you are not using the printer, press R/S to unpack each number. Output is in fix-3 scientific notation, and may be changed by replacing EE at location 218 with Nop. To begin unpacking at a different register than 05, press GTO B SST SST CE, enter the desired register number, and press R/S.

Let's demonstrate this program by packing the following numbers: -22222, .01234, -5, 25, -.013, 10000. (Note the wide range of numbers used.) Press A to initialize, enter each number individually and press R/S. Now press B, your printout will look like:

```
-2.223 04
 1.233-02
-5.000 00
 2.500 01
-1.300-02
 1.000 04
```

Press R/S to halt printing after the six numbers input have been unpacked. You'll note that the numbers printed after these initial six are the equivalent of 0 (i.e., 10^{-5}).

We have now discussed two very different packing methods which offer different benefits to you. It is for you to decide which method most optimally suits your data.

FROM THE ANALYST'S DESK

- For those members who have the program "Moment Distribution for a Variable Number of Spans" (PPX-59 #628020), please reverse the last two steps of instruction 11 on page 6 to read:

  Enter the magnitude on the right edge, press R/S.

  Enter the magnitude on the left edge, press R/S.

  Without these changes, the error that exists is only evident when the loads are uniformly varying.

- If you have trouble using alphanumerics while in engineering (Eng) or scientific (EE) mode, PPX member, Elmer B. Clausen of Buffalo, New York, has a solution. He has discovered an expanded alphanumeric code table:

This table can be used to avoid unneeded display manipulations while using **alphanumerics when the calculator is in Eng or EE modes.** The problem of printing an alphanumeric message built from the Personal Programming table (page VI-7), while in the Eng or EE mode, is that when the message ends in zero the trailing zero is dropped. This causes the message to change accordingly. For example, print JKLM using Op 06.

Out of Eng or EE modes:

The code for JKLM is 25262730 (from Personal Programming Page VI-7). Enter the code and press Op 04, and Op 06. 25262730 and JKLM are printed.

In the Eng Mode:

Press Eng, enter the code again, press Op 04, and Op 06. The printed output is:

25.26273 06 1Γ↕?

The characters to be printed have totally changed due to the dropping of the "0" in code 25262730. This could have been avoided by using the expanded code table. From this table, the code for JKLM would be 25262728. Notice that the code does not end in zero now. Enter 25262728, press Eng. Op 04, and Op 06. The printed output is:

25.262728 06 JKLM

The only restriction when using the expanded code table while in Eng or EE mode is — the symbols -, ., and ↑ cannot be used at the end of the alphanumeric message.

© 2011 Joerg Woerner
Datafile for Calculator Music

QUARTIC, CUBIC, AND QUADRATIC SOLUTIONS

This program solves an equation of the form $a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0 = 0$, where a_4, a_3, a_2, a_1, and a_0 are known real numbers ($a_4 \neq 0$), and x is an unknown complex number. The method employed is a direct, non-iterative solution, implementing the algorithm given in Abramowitz and Stegun, Handbook of Mathematical Functions. The program may be used to solve for the three roots of a cubic equation by the device of multiplying each term in the cubic by the unknown, 'x', and then solving the resulting quartic equation. The roots of the resulting quartic will include the roots of the original cubic, plus a fourth root, x = 0. Similar extension will provide the direct solution of a quadratic equation.

PPX wishes to thank the author of "Quartic, Cubic, and Quadratic Solutions", Dix Fulton, for his excellent program.

USER INSTRUCTIONS:

1. Enter program.
2. Clear all data memories by pressing CMs.
3. Enter coefficients a_4 through a_0 and press A through E respectively.

4a. **With printer:** Roots are calculated and printed by pressing E'.

4b. **Without printer:** Initiate calculation of the roots by pressing E'. To display the real parts of roots x_1 through x_4 press A' through D' respectively. After pressing a user defined label, press x ↔ t for the imaginary part of the root.

EXAMPLE:

Find the two roots of the equation:

$$x^2 + 2x + 3 = 0$$

Solution: Because this is a quadratic equation it must be changed to a quartic by multiplying each term by x^2:

$$x^4 + 2x^3 + 3x^2 = 0$$

Enter	Press	Display	Comments
	CMs	0.	Clear memories
1	A	1.	a_4
2	B	2.	a_3
3	C	3.	a_2
	E'	0.	
	A'	-1.	Real x_1
	x ↔ t	-1.414213562	Imag. x_1
	B'	-1.	Real x_2
	x ↔ t	1.414213562	Imag. x_2
	C'	0.	Real x_3
	x ↔ t	0.	Imag. x_3
	D'	0.	Real x_4
	x ↔ t	0.	Imag. x_4

Note: The solution of some cubic and quadratic equations may result in numbers being displayed for all four roots. This is due to roundoff errors, caused by the fact that some intermediate values can be stored in the calculator only as approximations. When this happens, choose the most reasonable answers by discarding the answers which are approximately equal to zero.

Datamath Calculator Museum

TI-59 LISTING*

000	76	LBL	048	32	X:T	096	04	4	144	65	×	192	03	3	240	08	08	288	00	00	336	95	=	384	00	0	432	02	02
001	16	A'	049	94	+/-	097	35	1/X	145	42	STO	193	95	=	241	94	+/-	289	54	)	337	50	I×I	385	67	EQ	433	32	X:T
002	22	INV	050	32	X:T	098	49	PRD	146	07	07	194	32	X:T	242	65	×	290	50	I×I	338	32	X:T	386	49	PRD	434	01	1
003	76	LBL	051	94	+/-	099	00	00	147	43	RCL	195	55	÷	243	02	2	291	34	fX	339	43	RCL	387	22	INV	435	03	3
004	17	B'	052	76	LBL	100	49	PRD	148	06	06	196	03	3	244	85	+	292	22	INV	340	06	06	388	58	FIX	436	00	0
005	86	STF	053	44	SUM	101	01	01	149	75	-	197	95	=	245	43	RCL	293	44	SUM	341	65	×	389	16	A'	437	03	3
006	01	01	054	75	-	102	49	PRD	150	43	RCL	198	37	P/R	246	08	08	294	05	05	342	43	RCL	390	17	B'	438	69	DP
007	06	6	055	73	RC*	103	02	02	151	05	05	199	65	×	247	85	+	295	95	=	343	05	05	391	18	C'	439	04	04
008	61	GTO	056	09	09	104	49	PRD	152	95	=	200	03	3	248	32	X:T	296	42	STO	344	85	+	392	19	D'	440	32	X:T
009	42	STO	057	55	÷	105	03	03	153	55	÷	201	34	fX	249	95	=	297	07	07	345	43	RCL	393	98	ADV	441	69	DP
010	76	LBL	058	03	3	106	43	RCL	154	02	2	202	55	÷	250	69	DP	298	43	RCL	346	08	08	394	98	ADV	442	06	06
011	18	C'	059	05	5	107	01	01	155	75	-	203	02	2	251	10	10	299	03	03	347	65	×	395	98	ADV	443	91	R/S
012	22	INV	060	01	1	108	42	STO	156	43	RCL	204	95	=	252	65	×	300	55	÷	348	43	RCL	396	76	LBL	444	76	LBL
013	76	LBL	061	07	7	109	06	06	157	07	07	205	42	STO	253	32	X:T	301	02	2	349	07	07	397	49	PRD	445	14	D
014	19	D'	062	01	1	110	33	X²	158	65	×	206	06	06	254	50	I×I	302	95	=	350	75	-	398	91	R/S	446	42	STO
015	86	STF	063	03	3	111	85	+	159	33	X²	207	94	+/-	255	22	INV	303	42	STO	351	43	RCL	399	76	LBL	447	01	01
016	01	01	064	02	2	112	43	RCL	160	95	=	208	42	STO	256	45	Y^	304	08	08	352	01	01	400	11	A	448	32	X:T
017	08	8	065	07	7	113	03	03	161	42	STO	209	05	05	257	03	3	305	85	+	353	95	=	401	42	STO	449	01	1
018	76	LBL	066	69	DP	114	49	PRD	162	08	08	210	32	X:T	258	85	+	306	53	(	354	50	I×I	402	04	04	450	03	3
019	42	STO	067	04	04	115	06	06	163	33	X²	211	22	INV	259	43	RCL	307	33	X²	355	77	GE	403	32	X:T	451	00	0
020	42	STO	068	02	2	116	33	X²	164	85	+	212	44	SUM	260	08	08	308	85	+	356	48	EXC	404	01	1	452	02	2
021	09	09	069	95	=	117	65	×	165	53	(	213	05	05	261	69	DP	309	43	RCL	357	43	RCL	405	03	3	453	69	DP
022	73	RC*	070	98	ADV	118	43	RCL	166	43	RCL	214	22	INV	262	10	10	310	06	06	358	05	05	406	00	0	454	04	04
023	09	09	071	69	DP	119	00	00	167	06	06	215	44	SUM	263	65	×	311	75	-	359	48	EXC	407	05	5	455	32	X:T
024	55	÷	072	06	06	120	85	+	168	55	÷	216	06	06	264	43	RCL	312	43	RCL	360	07	07	408	69	DP	456	69	DP
025	02	2	073	32	X:T	121	43	RCL	169	03	3	217	65	×	265	08	08	313	02	02	361	42	STO	409	04	04	457	06	06
026	95	=	074	42	STO	122	02	02	170	75	-	218	02	2	266	50	I×I	314	54	)	362	05	05	410	32	X:T	458	91	R/S
027	33	X²	075	09	09	123	94	+/-	171	43	RCL	219	95	=	267	22	INV	315	34	fX	363	76	LBL	411	69	DP	459	76	LBL
028	75	-	076	02	2	124	42	STO	172	07	07	220	32	X:T	268	45	Y^	316	22	INV	364	48	EXC	412	06	06	460	15	E
029	69	DP	077	04	4	125	07	07	173	33	X²	221	43	RCL	269	03	3	317	44	SUM	365	43	RCL	413	91	R/S	461	42	STO
030	39	39	078	03	3	126	65	×	174	54	)	222	05	05	270	76	LBL	318	08	08	366	04	04	414	76	LBL	462	00	00
031	73	RC*	079	00	0	127	53	(	175	65	×	223	22	INV	271	30	TAN	319	95	=	367	49	PRD	415	12	B	463	32	X:T
032	09	09	080	01	1	128	04	4	176	33	X²	224	77	GE	272	75	-	320	42	STO	368	00	00	416	42	STO	464	01	1
033	69	DP	081	03	3	129	65	×	177	95	=	225	38	SIN	273	43	RCL	321	06	06	369	49	PRD	417	03	03	465	03	3
034	29	29	082	02	2	130	43	RCL	178	29	CP	226	32	X:T	274	07	07	322	43	RCL	370	01	01	418	32	X:T	466	00	0
035	95	=	083	02	2	131	00	00	179	77	GE	227	76	LBL	275	95	=	323	06	06	371	49	PRD	419	01	1	467	01	1
036	29	CP	084	69	DP	132	54	)	180	39	COS	228	38	SIN	276	42	STO	324	65	×	372	02	02	420	03	3	468	69	DP
037	77	GE	085	04	04	133	22	INV	181	94	+/-	229	43	RCL	277	06	06	325	43	RCL	373	49	PRD	421	00	0	469	04	04
038	43	RCL	086	43	RCL	134	44	SUM	182	34	fX	230	06	06	278	55	÷	326	07	07	374	03	03	422	04	4	470	32	X:T
039	50	I×I	087	09	09	135	06	06	183	32	X:T	231	77	GE	279	02	2	327	85	+	375	00	0	423	69	DP	471	69	DP
040	34	fX	088	69	DP	136	95	=	184	43	RCL	232	95	=	280	95	=	328	43	RCL	376	82	HIR	424	04	04	472	06	06
041	32	X:T	089	06	06	137	94	+/-	185	08	08	233	32	X:T	281	42	STO	329	08	08	377	08	08	425	32	X:T	473	91	R/S
042	76	LBL	090	66	PAU	138	42	STO	186	32	X:T	234	61	GTO	282	05	05	330	65	×	378	01	1	426	69	DP	474	61	GTO
043	43	RCL	091	32	X:T	139	05	05	187	22	INV	235	30	TAN	283	85	+	331	43	RCL	379	69	DP	427	06	06	475	10	E'
044	34	fX	092	92	RTN	140	43	RCL	188	37	P/R	236	76	LBL	284	53	(	332	05	05	380	04	04	428	91	R/S			
045	87	IFF	093	76	LBL	141	07	07	189	32	X:T	237	39	COS	285	33	X²	333	75	-	381	82	HIR	429	76	LBL			
046	01	01	094	10	E'	142	55	÷	190	22	INV	238	34	fX	286	75	-	334	43	RCL	382	18	18	430	13	C			
047	44	SUM	095	43	RCL	143	03	3	191	45	Y^	239	44	SUM	287	43	RCL	335	01	01	383	32	X:T	431	42	STO			

Note: Key in steps 376 and 377 by pressing STO 82 STO 08, backstep (BST) and delete (Del) the two STO instructions. Then single step (SST) to location 378. Steps 381 and 382 can be keyed in a similar manner using STO 82 STO 18.

TEXAS INSTRUMENTS PRESENTS

Three new libraries have been produced by Texas Instruments and are now available through PPX-59. To order, simply write the library's name and catalog number on your order form and include your check or money order, plus $1.00 postage and handling, and your state's applicable sales tax.

ELECTRICAL ENGINEERING LIBRARY $35.00

(Order By Cat. No. 9A0038)

Phase-locked loops. Finds natural frequency, damping factor, loop-noise bandwidth and filter component values for either passive or active phase-locked loops.

S→Y Parameter Conversions. Transforms a set of S(Y) parameters to a set of Y(S) parameters.

Complex Arithmetic. Given two complex numbers X and Y the following operations are performed: X + Y, X - Y, X × Y, X ÷ Y, Y^X, $\sqrt[X]{Y}$ and $\log_X$ X.

Complex Functions. Evaluates the following functions for a complex number X: polar representation (r,θ) of X, rectangular representation (a + b i) of X, X^2, $\sqrt{X}$, 1/X, e^x and ln X.

Complex Trigonometric Functions. Given a complex number X the following functions and inverses are evaluated: sin X, cos X, tan X.

Ratio Conversions. Computes remaining three quantities with one of the following givens: decibels, nepers, power, voltage.

Signal Detection. Computes the remaining quantity when given any two of the following: signal to noise ratio, probability of detection, probability of false alarm.

Roots of a Polynomial. Uses a Lin-Bairstow method to find all roots, real and complex.

Chained Multiplication of Polynomials. Performs a chain multiplication of polynomial functions in one variable.

Reactance Chart. Simulates a standard reactance chart.

Series-Parallel Impedance Conversions. Converts a parallel resistance and reactance combination to series and vice versa.

Lowpass, Highpass and Bandpass Filters. Computes component values in the design of second order multiple feedback active lowpass, highpass and bandpass filters. Also computes parameters for Butterworth and Tchebysheff filters.

Convolution. Given the impulse response for a linear system, uses the convolution integral to find the system's output for an input waveform.

Root Locus Calculations. Given the open-loop poles and zeroes of a linear feedback system, computes the following root locus parameters: asymptote intersection point, asymptote angles, departure angles from complex poles and arrival angles at complex zeroes.

Discrete Fourier Transform. Transforms a time series of up to 32 points to the frequency domain and vice versa.

Smith Chart Calculations. Performs various transmission line calculations equivalent to graphical constructions on the Smith Chart.

Network Analysis. Computes frequency response of a general linear network made up of resistors, capacitors and inductors given starting and ending frequency and numbers of intervals desired.

AGRICULTURE LIBRARY $50.00

(Order By Cat. No. 9A0040)

Metric Conversion. Converts weight and temperature measurements.

Batch Mix. Calculates quantities of ingredients for x sizes of total mix.

Beef Cow Ration Analyzer. Determines the nutrient requirement of gestating and lactating beef cows and compares them to the nutrients contained in a given ration.

Feedlot Ration Analyzer. Projects from a given ration the average daily gain, cost per pound of gain, amount and type of protein supplement required, and the amount of calcium and phosphorus supplied.

MP and UFP Determination. Calculates metabolizable protein and urea fermentation potential values for feedstuffs and supplements.

Dairy Ration Balancer. Calculates grain feeding requirements necessary to supplement different levels of milk production on various forage feeding programs for dairy cattle.

Swine and Poultry Ration Formulation. Provides amounts of ingredients required for specified protein or lysine levels. Also calculates percent protein, lysine, calcium, phosphorus and amount of metabolizable energy per pound.

Swine and Poultry Ration Analysis. Calculates the average composition of up to seven nutrients for any number of ingredients that are used in a ration or formula.

Relative Value of Swine Feed Ingredients. Uses the prices of corn, soybean meal and dicalcium phosphate to determine the competitive value of other feedstuffs as sources of energy, lysine and phosphorus.

Gestation Management. Uses breeding date to calculate expected birth date, and upcoming management practice schedules.

Beef Weaning and Yearling Weight Adjustment. Adjusts weaning weights to a 205-day basis and yearling weights to 365 days. Any birth weights and age of dam adjustment factors can be utilized.

Cow-Calf, Ewe or Feeder Pig Production Work Sheet. Performs an economic analysis of the beef cow-calf, ewe and feeder pig production enterprises.

Cattle, Pig or Lamb Feeding Work Sheet. Performs an economic analysis of the cattle feeding, lamb feeding and the feeder pig finishing enterprises.

Land Purchase and Farm Loan Analysis. Performs a land purchase, financial and economic analysis. Also evaluates three types of loans commonly used by farmers.

RPN SIMULATOR $35.00

(Order By Cat. No. 9A0039)

Requires TI Programmable 59. Enlarge your collection of TI Programmable 59 software by quickly converting programs written in Reverse Polish Notation. This library, with the aid of the PC-100A/C, lets you convert most RPN programs in three easy steps.

- Enter the RPN keycodes and watch your PC-100A/C print the corresponding TI-59 keystrokes in a program listing format.
- Key the program back into your TI-59 and record it on magnetic cards for future use.
- Execute the program on your TI-59. Programs included in this library act as subroutines which simulate RPN instructions.

© 2011 Joerg Woerner
Datamath Calculator Museum

FROM THE ANALYST'S DESK *(cont'd.)*

• Mr. Jared Weinberger of Bologna, Italy, has passed along the following example of an Op 10 (signum function) application. Using Op 10 **enables you to obtain an non-branching (t register is not used)** solution for sequence processing problems, thus saving program steps and memory registers. One example of a sequence processing problem is the bank service charge problem mentioned in the Personal Programming Manual (pages IV-93-98).

The problem states that, as a manager of a prominent local bank, you need a fast and easy method of determining the service charge for the customers who have accounts with your bank.

The service charge is based on the following rates:

$0.10 per check for the first five checks (1-5),
$0.09 per check for the next five (6-10),
$0.08 per check for the next five (11-15),
$0.07 per check for each check over 15.

Mr. Weinberger's program consists of only 33 steps as follows:

000	76	LBL	017	00	00
001	11	A	018	43	RCL
002	58	FIX	019	00	00
003	02	02	020	65	×
004	42	STO	021	53	(
005	00	00	022	69	OP
006	65	×	023	10	10
007	93	.	024	85	+
008	01	1	025	01	1
009	16	A'	026	54	)
010	16	A'	027	55	÷
011	76	LBL	028	02	2
012	16	A'	029	00	0
013	75	-	030	00	0
014	05	5	031	95	=
015	22	INV	032	92	RTN
016	44	SUM			

© 2011 Joerg Woerner
Datamath Calculator Museum

Enter the number of checks and press A to calculate the service charge.

How does Op 10 apply to this solution? First, remember that Op 10 assigns the signum function to the value of x currently in the display register and responds with the following:

display register value x	display response
x > 0	1.
x = 0	0.
x < 0	-1.

The algorithm used in Mr. Weinberger's solution is basically the same as the one used in the Personal Programming Manual's solution. This algorithm is: Multiply the number of checks by $ .10 and then substract $ .01 for the number of checks over 5, 10, and 15. In the Op 10 solution, the subtraction sequence (LBL A') is performed three times regardless of the number of checks. This sequence is (in essence): -5 INV SUM 00 RCL 00 x .01 (register 00 is used to store the number of checks). LBL A is used to perform the sequence three times.

Complications arise when n - 5, n - 10, or n - 15 result in a negative number, where n is equal to the number of checks. In each case, $ .01 should not be subtracted. The following guidelines determine when the subtraction should take place:

(1) If the contents of register 00 are negative, the product of RCL 00 x .01 should be multiplied by 0. The sequence, RCL 00 x (Op 10 + 1) provides the necessary zero for this multiplication. The result is that no subtraction takes place.
(2) If the contents of register 00 are zero, a correction need not be made (the multiplication takes care of itself). Again, no subtraction takes place.
(3) If contents of register 00 are positive, the product of RCL 00 x .01 should be multiplied by 1. However, as steps 021-026 result in multiplication by 2 rather than 1; a correction must be made. To do this, RCL 00 x .01 must be changed to RCL 00 x .005. This is accomplished by dividing RCL 00 by 200 (thus saving one step). In this case, the following subtraction takes place: $ .01 is subtracted x times, where x is equal to the **total** number of checks greater than 5, 10, and 15.

In this short example, we saved six steps and one memory register, and avoided use of the t register (compared to the manual's solution). We hope that this has given you some food for thought!

• PPX member, Mr. Jim Roe, has discovered a way to extend the Memo Pad program, PPX #908016 (or LE-10), to print all characters, not just those on a telephone pad. Six program steps need to be added to the program listing of PPX #908016:

306	76	LBL
307	10	E'
308	71	SBR
309	01	01
310	24	24
311	92	RTN

If you have the Leisure Library, the program must be downloaded into calculator memory, the above steps added, and then recorded on a magnetic card for future use. Downloading is explained in your Personal Programming Manual, page III-4.

After including the six additional steps, any special character can be added to the memo pad message by two user instructions:

(1) Enter the code of the character from the table on page VI-7 of your Personal Programming Manual. Press E'.
(2) For example, let's print out the word hello followed by an exclamation mark.

Enter	Press	Display	Comments
	SBR CLR	20.	
3	B	19.	H
2	B	18.	E
4	C	17.	L
4	C	16.	L
5	C	15.	O
73	E'	14.	!
	A'	20.	Prints Hello!

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

Copyright © 1979, By Texas Instruments Incorporated