CALCULATOR PRODUCTS

Tektronix, Inc.
P.O. Box 500, Beaverton, Ore. 97005, U.S.A.

31 MATHEMATICS PROGRAM LIBRARY

VOLUME 1

062-1593-00

INTRODUCTION

This Mathematics Program Library is intended to provide you with tested, workable programs to cover a broad range of mathematical applications of your Tektronix 31 Programmable Calculator. The programs are catagorized into seven sections, ranging from GEOMETRY AND TRIGONOMETRY through NUMERICAL INTEGRATION AND DIFFERENTIAL EQUATIONS.

The majority of these programs are written for the standard TEK 31 Calculator equipped with the 512 step program memory and 64 registers. Two of the matrix programs are designed for the 1024 step Option 04 machine. With larger memory options, more than one program can be loaded and operated, so long as the subroutine LABELs are compatible.

Each program is fully documented with PROGRAM DESCRIPTION, working EXAMPLES (often with reproductions of actual printouts), PROGRAM EXECUTION, and PROGRAM STEPS. In addition, most programs have a NOTES page which you may use to document your program modifications or your own special examples.

All of these programs are written assuming that the optional silent thermal printer is installed in your TEK 31 Calculator. Consequently, the PRINTOUT column on the PROGRAM EXECUTION page always shows the program printout of input data and results. In many cases the EXAMPLES page shows reproductions of actual printouts for the examples given. However, in many programs the printer is not essential, and the results (and often the input data as well) may be manually recalled to the display from the storage registers listed in the PROGRAM EXECUTION instructions. You should note that because of space limitations the listing in the PRINTOUT column may not match exactly the sequence of execution and displays listed in the PRESS and DISPLAY columns.

Many of the programs in this library include options for the "linking" or "chaining" of operations. In these programs the calculated results are automatically stored in place of the original input data so that the program is immediately ready to perform a new operation on the results obtained from the previous operation. For example, in the section COMPLEX

INTRODUCTION

OPERATORS all of the individual programs incorporate this feature so that a whole series of operations on complex numbers may be carried out without the necessity for manual re-entry of data. In these programs the printout will include alphanumeric and mathematical symbols to detail each operation performed in the sequence. To utilize the linkage capability fully, these programs should be stored on magnetic tape cartridges. Some programs require more than one BLOCK for storage. Wherever possible automatic loading routines have been included to simplify the operation of these programs.

Many programs include a variety of options in their operations. For example, program 7-1 includes three different routines for numerical integration: Trapezoidal Rule; Simpson's 1/3 Rule; and Newton's 3/8 Rule. These options are accessed through the subroutine LABEL system via the EXECUTE key. Wherever possible these programs are made interactive with the user through the inclusion of alphanumeric messages in the input/output printer operation. In addition, most programs end with a cleared display so that they are immediately ready for entry of new data.

Careful study of the various programming techniques incorporated in these programs should help you in writing additional programs to solve your own special problems. We suggest that you thoroughly document any programs you write; a month-old set of program steps without a description, working example, and execution instructions may be almost impossible to decipher and use.

We hope these programs will be useful to you in your mathematical analysis. We welcome your comments and suggestions, and we encourage your submission of other programs you find useful.

Calculator Software Development
Tektronix, Inc.
P.O. Box 500
Beaverton, Oregon 97005

PROGRAM LISTING

PROGRAM LISTING


TEKTRONIX
31

TITLE

Mathematics | 1-1

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

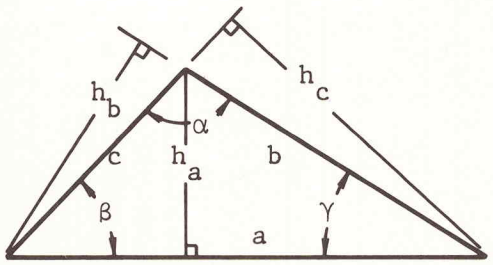
$K_{\emptyset}$: working
K_1 : a
K_2 : b
K_3 : c
K_4 : α
K_5 : β
K_6 : γ
K_7 : h_a
K_8 : h_b
K_9 : h_c

$R_{\emptyset\emptyset\emptyset}$: Area
$R_{\emptyset 1}$: Perimeter

Subroutine labels used:
LBL Ø, 1, 2, 3, 4,
5, 6, 7, 8, 9

PROGRAM DESCRIPTION

This program is designed to solve plane triangles. Four basic subroutines are included to solve for the sides, angles, altitudes to each side, the area, and the perimeter, given various combinations of sides and angles. Angles may be entered in either degrees or radians.



Subroutine LBL 1:

Given three sides a, b, and c.

Subroutine LBL 2:

Given two sides a, b, and their included angle γ.

Subroutine LBL 3:

Given side a, and its including angles β and γ.

Subroutine LBL 4:

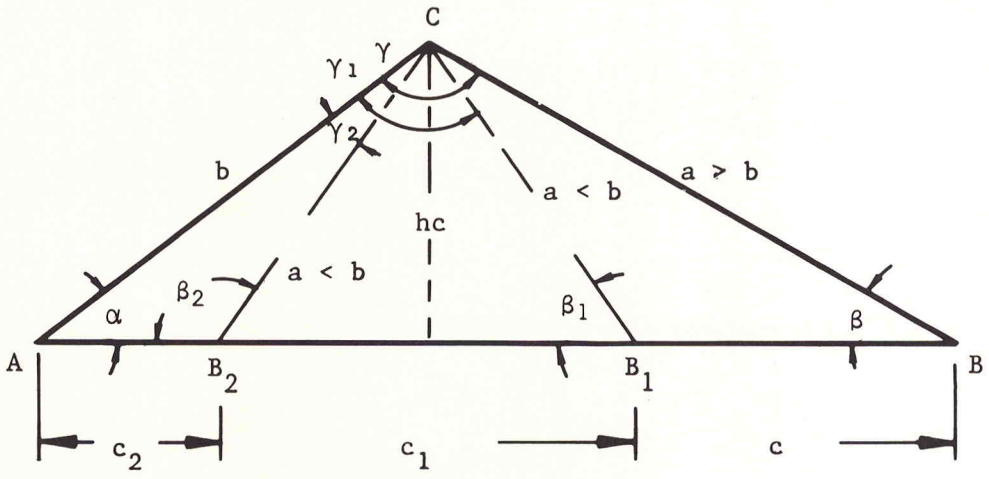
Given two sides a, b, and the opposite angle α.

Each subroutine computes the remaining sides and angles, the three altitudes, and the area and perimeter of the triangle.

Note that in the case of subroutine LBL 4, that given the two sides a and b, and the opposite angle α, the solution of the triangle is unique only if side a $\geq$ side b. If side a is the shorter, then it may be possible to construct two dissimilar triangles that satisfy the input data. In this case the program solves first for the case where $\beta = \beta_1 \leq 90^\circ$ (or $\pi/2$). The program will then automatically compute new input data and branch to subroutine LBL 3 to solve for the second triangle where $\beta = \beta_2 > 90^\circ$. Results for both triangles are automatically printed out.

Note also for this case it is possible for side a to be too short to intersect side c (that is, $a < h_c$), and thus no solution is possible. In this case the program will print the entered data, and 0's for the remaining sides and angles. The program will then truncate with a flashing display.

These anomolies for subroutine LBL 4 can be seen from the sketch below:

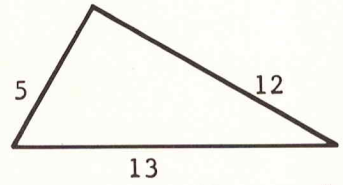


NOTES

EXAMPLES

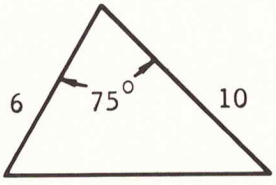
Solve the following triangles:

LBL 1: Given 3 sides:



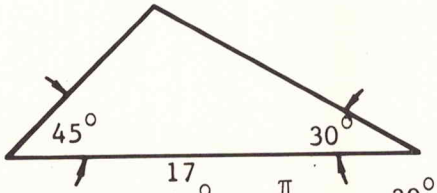
a = 13, b = 12, c = 5

LBL 2: Given 2 sides and included angle:



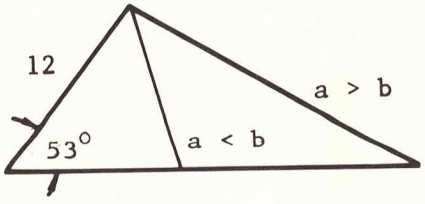
a = 10, b = 6, $\gamma = 75^\circ$ or $\frac{5\pi}{12}$

LBL 3: Given 2 angles and included side:



a = 17, $\beta = 45^\circ$ or $\frac{\pi}{4}$, $\gamma = 30^\circ$ or $\frac{\pi}{6}$

LBL 4: Given 2 sides and opposite angle:



For a > b: a = 20, b = 12, $\alpha = 53^\circ$

For a < b: a = 10, b = 12, $\alpha = 53^\circ$

Try a = 9, b = 12, $\alpha = 53^\circ$
(display will flash since a is too short to intersect side c).

```
PLANE TRIANGLE:

#1

SIDES:
          13.
          12.
           5.

ANG:
          90.
 67.38013505
 22.61986495

ALT:
 4.615384615
           5.
          12.

A&P:
          30.
          30.

END

#2

SIDES:
          10.
           6.
 10.24410633

ANG:
 70.54593807
 34.45406193
          75.

ALT:
 5.795554958
 9.659258263
 5.657452948

A&P:
 28.97777479
 26.24410633

END

#3

SIDES:
          17.
 12.44486373
 8.799847533

ANG:
         105.
          45.
          30.

ALT:
 6.222431864
          8.5
 12.02081528

A&P:
 52.89067085
 38.24471126

END
```

```
#4

SIDES:
          20.
          12.
 24.77609913

ANG:
          53.
 28.63194546
 98.36805454

ALT:
 11.87224354
 19.78707257
  9.58362612

A&P:
 118.7224354
 56.77609913

END

#4

SIDES:
          10.
          12.
 10.07732057

ANG:
          53.
 73.40803156
 53.59196844

ALT:
 9.657727268
 8.048106056
 9.583626121

A&P:
 48.28863634
 32.07732057

SIDES:
          10.
          12.
 4.366239982

ANG:
          53.
 106.5919684
 20.40803156

ALT:
 4.184441154
 3.487034295
 9.583626121

A&P:
 20.92220577
 26.36623998

END

#4
          9.#
         12.#
          0.#

         53.#
          0.#
          0.#

END
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1	Set D/R to desired operating mode.			
2		START	0	PLANE TRIANGLES:
3	Select subroutine to match input data:			#1, #2, #3, or #4
3a	For 3 sides:	EXC 1	0	SIDES:
	a	CONT	0	a : K_1
	b	CONT	0	b : K_2
	c	CONT	0	c : K_3
				ANG:
3b	For 2 sides, 1 angle	EXC 2	0	α : K_4
	a	CONT	0	β : K_5
	b	CONT	0	γ : K_6
	γ	CONT	0	ALT:
				h_a : K_7
3c	For 1 side, 2 angles	EXC 3	0	h_b : K_8
	a	CONT	0	h_c : K_9
	β	CONT	0	A & P:
	γ	CONT	0	A : $R_{ØØ1}$
				P : $R_{Ø1}$
3d	For 2 sides, opposite angle	EXC 4	0	END
	a	CONT	0	
	b	CONT	0	
	α	CONT	0	
	In the case where two triangles are realizable from the given a, b, and α, the calculator will STOP with π in the display. To solve for the second triangle, press CONT.			
4	If a printer is not used, results may be recalled manually from the indicated registers.			
5	For a new triangle return to Step 1, 2, or 3 as desired.			

PROGRAM STEPS

		0	1	2	3	4	5	6	7	8	9	COMMENTS
0	00	PF	PF	P	L	A	N	E	SPC	T	R	
	10	I	A	N	G	L	E	:	CD	PF	RSET	
	20	LBL	1	#	1	CD	STOP	=	K	1	CD	LBL 1
	30	STOP	=	K	2	CD	STOP	=	K	3	x^2	
	40	+	K	2	x^2	-	K	1	x^2	=	÷	
	50	2	÷	K	2	÷	K	3	=	arc	cos	
	60	=	K	4	K	1	x^2	+	K	3	x^2	
	70	-	K	2	x^2	=	÷	2	÷	K	1	
	80	÷	K	3	=	arc	cos	=	K	5	EXC	
	90	9	EXC	5	EXC	Ø	LBL	2	#	2	CD	LBL 2
1	00	STOP	=	K	1	x^2	=	K	Ø	CD	STOP	
	10	=	K	2	x^2	Σ_0	CD	STOP	=	K	6	
	20	cos	×	K	1	×	K	2	×	2	+/-	
	30	+	K	Ø	=	$\sqrt{x}$	=	K	3	x^2	Σ_0	
	40	K	1	x^2	×	2	+/-	)	Σ_0	K	Ø	
	50	÷	2	÷	K	2	÷	K	3	=	arc	
	60	cos	=	K	4	EXC	8	K	4	-	K	
	70	6	=	K	5	EXC	5	EXC	Ø	LBL	3	LBL 3
	80	#	3	CD	STOP	=	K	1	CD	STOP	=	
	90	K	5	CD	STOP	=	K	6	LBL	7	EXC	LBL 7
2	00	8	K	5	-	K	6	=	K	4	sin	
	10	1/x	×	K	1	×	K	5	sin	=	K	
	20	2	÷	K	5	sin	×	K	6	sin	=	
	30	K	3	EXC	5	EXC	Ø	LBL	4	#	4	LBL 4
	40	CD	STOP	=	K	1	x^2	=	K	Ø	CD	
		0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
50	STOP	=	K	2	x^2	Σ_0	CD	STOP	=	K	
60	4	K	1	-	K	2	×	K	4	sin	
70	=	K	9	IF<Ø	EXC	6	CONT	K	1	-	
80	K	9	=	÷	K	1	=	arc	sin	=	
90	K	5	EXC	9	K	6	cos	×	K	1	
3 00	×	K	2	×	2	+/-	+	K	Ø	=	
10	$\sqrt{x}$	=	K	3	EXC	5	K	1	-	K	
20	2	)	IF<Ø	EXC	8	K	5	=	K	5	
30	EXC	9	PF	π	STOP	EXC	7	CONT	EXC	Ø	
40	LBL	5	PF	(S)	(I)	(D)	(E)	(S)	(:)	K	LBL 5
50	1	PRNT	K	2	PRNT	K	3	PRNT	PF	(A)	
60	(N)	(G)	(:)	K	4	PRNT	K	5	PRNT	K	
70	6	PRNT	PF	sin	×	K	2	=	K	7	
80	(A)	(L)	(T)	(:)	PRNT	K	1	×	K	6	
90	sin	=	K	8	PRNT	K	1	×	K	5	
4 00	sin	=	K	9	PRNT	PF	×	K	3	÷	
10	2	=	Rxxx	Ø	Ø	Ø	(A)	(&)	(P)	(:)	
20	PRNT	K	1	+	K	2	+	K	3	=	
30	Rxx	Ø	1	PRNT	RADR	GODP	LBL	6	CD	1/x	LBL 6
40	K	1	PRNT	K	2	PRNT	CD	=	K	3	
50	PRNT	PF	K	4	PRNT	CD	=	K	5	PRNT	
60	=	K	6	PRNT	EXC	Ø	LBL	8	CD	1	LBL 8
70	+/-	arc	cos	-	RADR	GODP	LBL	9	CD	1	LBL 9
80	+/-	arc	cos	-	K	4	-	K	5	=	
90	K	6	RADR	GODP	LBL	Ø	PF	(E)	(N)	(D)	LBL Ø
	0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

COMMENTS

	0	1	2	3	4	5	6	7	8	9
5	CD	PF	PF	RSET						
00										
10										
20										
30										
40										
50										
60										
70										
80										
90										
00										
10										
20										
30										
40										
50										
60										
70										
80										
90										
00										
10										
20										
30										
40										
	0	1	2	3	4	5	6	7	8	9

Plane Closed Polygon Solution

TITLE

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

$K_{\emptyset}$: Area
K_1 : Perimeter
K_2 : Σ Interior Angles
K_3 : x_0
K_4 : y_0
K_5 : counter
K_6 : working
K_7 : working
K_8 : working
K_9 : 180° or π rad

R001 : $x_0, x_1, \ldots, x_{n-1}$
R02 : $y_0, y_1, \ldots, y_{n-1}$
R03 : $x_1, x_2, \ldots, x_n$
R04 : $y_1, y_2, \ldots, y_n$
R05 : working
R06 : working

Subroutine Labels
LBL Ø, 1, 2, 3

PROGRAM DESCRIPTION

Given the coordinates for the vertices of any plane, simple closed polygon, this program calculates the length of each side, the interior angle at each vertex, and the enclosed area of the polygon, its perimeter (summation of sides), and the sum of its interior angles.

The polygon may be either convex or concave, that is, it may have interior angles θ over the full range $0 < \theta < 360^{\circ}$.

In order to calculate the correct values for the interior angles $\theta_1, \theta_2 \ldots \theta_0$, the coordinates for the vertices must be entered in clockwise (CW) progression.

The program saves the coordinates of the initial point (x_0, y_0) and compares them to each point entered. When an entered point (x_n, y_n) coincides with (x_0, y_0), the program automatically truncates, printing the area, perimeter, and summation of the interior angles.

The sum of the interior angles for a plane closed polygon is given by

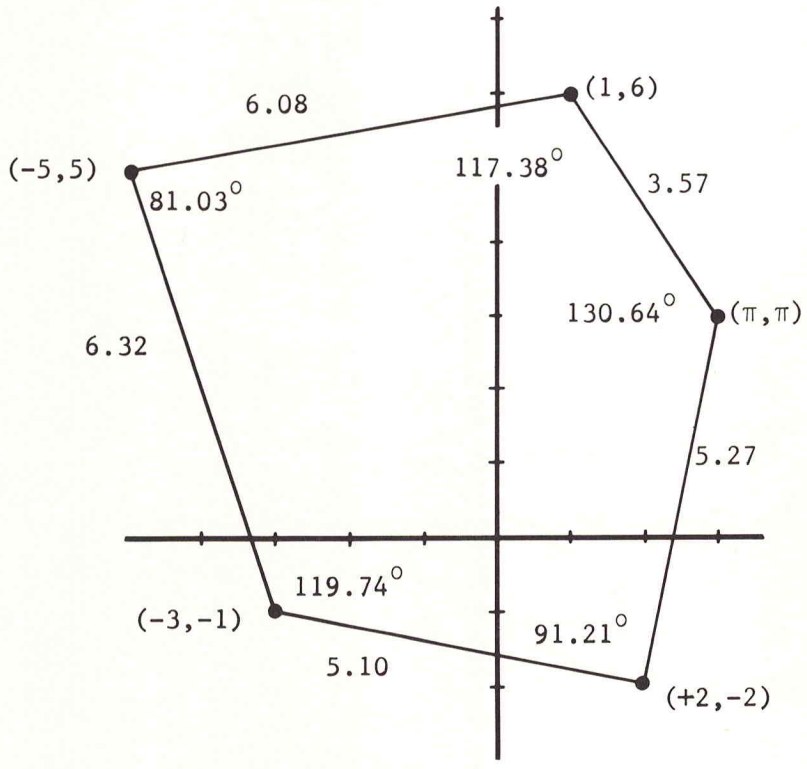
$$\Sigma\theta = (n - 2)\ 180^{\circ}$$

where n is the number of sides.

This program calculates Σθ by accumulating the sum of the interior angles as they are calculated. If the sum does not equal $(n - 2)\ 180^{\circ}$ with allowance for roundoff, the program is not operating properly. If the sum equals $(n + 2)\ 180^{\circ}$, the vertices have been entered in CCW progression.

1. Given the polygon with coordinates as sketched below, enter (x_i, y_i) values in CW order, beginning at point (2, -2). Repeat using any other point as (x_0, y_0), or repeat in CCW order to see the effect on the calculation of the interior angles.

```
CLOSED POLYGON:
ENTER & PRINT
COORDINATES IN
CW ORDER; CALC &
PRINT SIDES,
INTERIOR ANGLES,
AREA, PERIMETER,
& SUM INT ANGLES

X0=
              2.
Y0=
-             2.


 5.099019514

-             3.
-             1.

 119.7448813

  6.32455532

-             5.
              5.

  81.0273734

  6.08276253

              1.
              6.

 117.3791771

 3.571681908

 3.141592654
 3.141592654

 130.6401373

 5.266802522

              2.
-             2.

 91.20843094
```




```
AREA=
 45.63716694

PERIMETER=
 26.34482179

SUM INT ANGLES=
          540.

END
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1	Set D/R to desired operating mode.			
2		START	0	TITLE
3	x_0	CONT	0	x_0
	y_0	CONT	0	y_0
				*side 0, 1
4	x_1	CONT	0	x_1
	y_1	CONT	0	y_1
				*angle 1
				side 1, 2
5	x_2	CONT	0	x_2
	y_2	CONT	0	y_2
				angle 2
				side 2, 3
6	x_3	CONT	0	x_3
	y_3	CONT	0	y_3
	.			angle 3
	.			side 2, 3
	.			.
				.
				.
				angle n-1
				side n-1, 0
7	$x_n = x_0$	CONT		$x_n = x_0$
	$y_n = y_0$	CONT		$y_n = y_0$
				angle 0
	To solve a new polygon,			area : $K_\emptyset$
	press	EXC Ø or START		perimeter : K_1
				Σ int angles: K_2
				END

*When a printer is not used, replace PRNT steps with STOP as detailed on the PROGRAM STEPS pages to display each calculated side and angle in turn.

PROGRAM STEPS

		0	1	2	3	4	5	6	7	8	9
0	00	PF	PF	C	L	O	S	E	D	SPC	P
	10	O	L	Y	G	O	N	:	SPC	E	N
	20	T	E	R	SPC	&	SPC	P	R	I	N
	30	T	PF	C	O	O	R	D	I	N	A
	40	T	E	S	SPC	I	N	PF	C	W	SPC
	50	O	R	D	E	R	;	SPC	C	A	L
	60	C	SPC	&	P	R	I	N	T	SPC	S
	70	I	D	E	S	,	PF	I	N	T	E
	80	R	I	O	R	SPC	A	N	G	L	E
	90	S	,	A	R	E	A	,	SPC	P	E
1	00	R	I	M	E	T	E	R	,	&	SPC
	10	S	U	M	SPC	I	N	T	SPC	A	N
	20	G	L	E	S	LBL	Ø	PF	X	Ø	=
	30	CD	=	K	Ø	=	K	1	=	K	2
	40	STOP	=	Rxxx	Ø	Ø	1	=	K	3	PRNT
	50	Y	Ø	=	CD	STOP	=	Rxx	Ø	2	=
	60	K	4	PRNT	PF	PF	CD	STOP	=	Rxx	Ø
	70	3	CD	STOP	=	Rxx	Ø	4	CD	1	+/-
	80	=	K	5	arc	cos	=	K	9	LBL	3
	90	Rxx	Ø	1	-	Rxx	Ø	3	=	K	7
2	00	Rxx	Ø	2	-	Rxx	Ø	4	=	K	8
	10	$\sqrt{\Sigma x^2}$	K	7	=	Σ_1	PRNT*	PF	1/x	×	K
	20	7	)	Ø	arc	cos	=	Rxx	Ø	5	K
	30	8	IF<Ø	K	9	-	Rxx	Ø	5	)	×
	40	2	+	Rxx	Ø	5	=	Rxx	Ø	5	CONT
		0	1	2	3	4	5	6	7	8	9

COMMENTS

LBL Ø
Initialization
and data entry

LBL 3
Calc. side

*If a printer is not used, replace PRNT with STOP to display the calculated side.

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9
50	K	5	+	1	=	K	5	IF=Ø	Rxx	Ø
60	5	=	K	6	CONT	Rxx	Ø	1	+	Rxx
70	Ø	3	)	÷	2	×	K	8	)	Σ_0
80	Rxx	Ø	3	−	K	3	)	IF=Ø	Rxx	Ø
90	4	−	K	4	)	IF=Ø	EXC	1	CONT	Rxx
3 00	Ø	3	=	Rxx	Ø	1	PRNT	Rxx	Ø	4
10	=	Rxx	Ø	2	PRNT	PF	CD	STOP	=	Rxx
20	Ø	3	−	Rxx	Ø	1	=	K	7	CD
30	STOP	=	Rxx	Ø	4	−	Rxx	Ø	2	=
40	K	8	$\sqrt{\Sigma x^2}$	K	7	=	1/x	×	K	7
50	)	Ø	arc	cos	=	Rxx	Ø	6	K	8
60	IF<Ø	K	9	−	Rxx	Ø	6	)	×	2
70	+	Rxx	Ø	6	=	Rxx	Ø	6	CONT	EXC
80	2	EXC	3	LBL	1	K	6	+	K	9
90	=	Rxx	Ø	6	−	2	×	K	9	)
4 00	IF≥Ø	=	Rxx	Ø	6	CONT	K	3	PRNT	K
10	4	PRNT	PF	EXC	2	PF	PF	K	Ø	x^a
20	1	=	K	Ø	(A)	(R)	(E)	(A)	(=)	PRNT
30	PF	K	1	(P)	(E)	(R)	(I)	(M)	(E)	(T)
40	(E)	(R)	(=)	PRNT	PF	K	2	(S)	(U)	(M)
50	(SPC)	(I)	(N)	(T)	(SPC)	(A)	(N)	(G)	(L)	(E)
60	(S)	(=)	PRNT	PF	(E)	(N)	(D)	CD	PF	PF
70	RSET	LBL	2	Rxx	Ø	6	−	Rxx	Ø	5
80	)	IF<Ø	+	K	9	×	2	CONT	=	PRNT*
90	PF	+	K	2	=	K	2	RADR	GODP	
	0	1	2	3	4	5	6	7	8	9

*If a printer is not used, replace PRNT with STOP to display calculated interior angle.

COMMENTS

LBL 1
Conclusion
Print
Area : $K_Ø$
Perimeter : K_1
Σ Int Angles : K_2

LBL 2
Calc. and print interior angle.

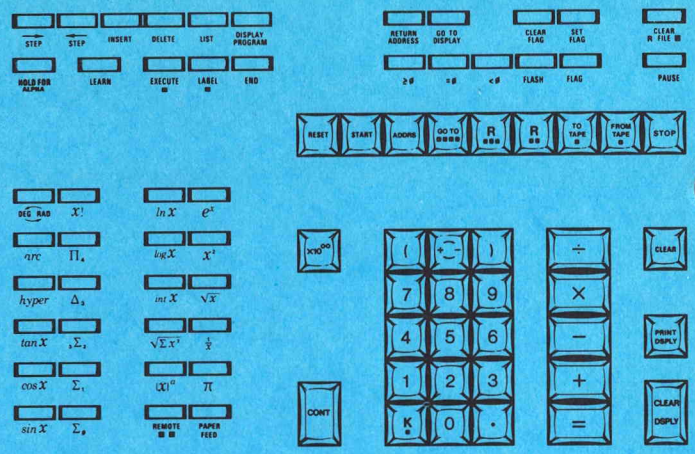
NOTES

SECTION 2 GENERAL FUNCTIONS

OCTAL CODE	PRINT OUT	KEY	SYMBOL
001	LBL_	LABEL ■	LBL
002	FTP_	FROM TAPE ■	FTP
003	TTP_	TO TAPE ■	TTP
004	EXC_	EXECUTE ■	EXC
005	CFI_	CLEAR R FILE ■	CFI
007	GOTO	GO TO ■■■■	GT
010	R___	R ■■■	Rxxx
011	R__	R ■■	Rxx
040	CLDP	CLEAR DSPLY	CD
041	IFFL	FLASH	IFFL
042	S FG	SET FLAG	SFG
043	STOP	STOP	STOP
044	PRNT	PRINT DSPLY	PRNT
045	CLR	CLEAR	CLR
046	PAUS	PAUSE	PAUS
047	CLFG	CLEAR FLAG	CLFG
050	(	(	(
051	)	)	)
052	*	×	×
053	+	+	+
054	RSET	RESET	RSET
055	-	−	-
056	.	.	.
057	/	÷	÷
060	0	0	ø
061	1	1	1
062	2	2	2
063	3	3	3
064	4	4	4
065	5	5	5
066	6	6	6
067	7	7	7
070	8	8	8
071	9	9	9
072	ADR	ADDRS	ADR
073	STRT	START	STRT
074	IF<0	<ø	IF<ø
075	=	=	=
076	IF>=	≥ø	IF≥ø
077	IF=0	=ø	IF=ø

OCTAL CODE	PRINT OUT	KEY	SYMBOL
100	K_	K ■	K
101	DG/R	DEG RAD	D/R
102	ARC	arc	arc
103	HYP	$hyper$	hyp
104	TAN	$\tan x$	tan
105	COS	$\cos x$	cos
106	SIN	$\sin x$	sin
107	X!	$x!$	x!
110	PI4	Π_4	Π_4
111	DLT3	Δ_3	Δ_3
112	3SM2	${}_3\Sigma_2$	${}_3\Sigma_2$
113	SUM1	Σ_1	Σ_1
114	SUM0	$\Sigma_ø$	Σ_0
115	LN	$\ln x$	ln
116	LOG	$\log x$	log
117	INT	$int\, x$	int
120	RSSQ	$\sqrt{\Sigma x^2}$	$\sqrt{\Sigma x^2}$
121	X↑A	$\|x\|^a$	x^a
122	RM__	REMOTE ■■	RMT
123	E↑X	e^x	e^x
124	X↑2	x^2	x^2
125	SQRT	$\sqrt{x}$	$\sqrt{x}$
126	1/X	$\frac{1}{x}$	1/x
127	PI	π	π
130	PAPR	PAPER FEED	PF
131	*10↑	$\times 10^{oo}$	10^{oo}
132	CONT	CONT	CONT
133	R AD	RETURN ADDRESS	RADR
134	+/-	+/-	+/-
135	GODP	GO TO DISPLAY	GODP
136	IFFG	FLAG	IFFG
137	ENDR	END	ENDR

OCTAL CODE	PRINT OUT	KEY	SYMBOL
201		LABEL ■	LBL
202		FROM TAPE ■	FTP
203		TO TAPE ■	TTP
204		EXECUTE ■	EXC
205		CLEAR R FILE ■	CFI
207		BELL	BELL
210		SPACE ←	SPC←
211		TAB ← STEP	TAB
220		STEP	STP←
221		INSERT	NSRT
222		DELETE	DLT
223		→ STEP	STP→
224		LIST	LIST
225		DISPLAY PROGRAM	DPRG
226		LEARN	LRN
240		SPACE	SPC
241	!	!	!
242	"	"	"
243	#	#	#
244	$	$	$
245	%	%	%
246	&	&	&
247	'	'	'
250	(	(	(
251	)	)	)
252	*	*	*
253	+	+	+
254	,	,	,
255	-	-	-
256	.	.	.
257	/	/	/
260	0	0	∅
261	1	1	1
262	2	2	2
263	3	3	3
264	4	4	4
265	5	5	5
266	6	6	6
267	7	7	7
270	8	8	8
271	9	9	9
272	:	:	:
273	;	;	;
274	<	<	<
275	=	=	=
276	>	>	>
277	?	?	?



OCTAL CODE	PRINT OUT	KEY	SYMBOL
300	@	@	@
301	A	A	A
302	B	B	B
303	C	C	C
304	D	D	D
305	E	E	E
306	F	F	F
307	G	G	G
310	H	H	H
311	I	I	I
312	J	J	J
313	K	K	K
314	L	L	L
315	M	M	M
316	N	N	N
317	O	O	O
320	P	P	P
321	Q	Q	Q
322	R	R	R
323	S	S	S
324	T	T	T
325	U	U	U
326	V	V	V
327	W	W	W
330	X	X	X
331	Y	Y	Y
332	Z	Z	Z
333	[	[	[
334	\	\	\
335	]	]	]
336	↑	↑	↑
337	_	_	_

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

	$\Gamma(x)$ or x!
K_0 :	x
K_1 :	ln
K_2 :	log
K_3 :	$\Gamma(x)$ or x!
K_4 :	mantissa
K_5 :	exponent
K_6 :	$R_n(x)$ for ln $\Gamma(x)$ or ln x!
K_9 :	x or x + 1

Subroutine labels used:
1, 2, IFFG, IF≥Ø,
x!, 10^{oo},
(A) , (?)

PROGRAM DESCRIPTION

For $1 < x \leq 1 \times 10^9$ this program calculates

ln $\Gamma(x)$		ln x!
log $\Gamma(x)$		log x!
$\Gamma(x)$	or	x!
mantissa $\Gamma(x)$		mantissa x!
exponent $\Gamma(x)$		exponent x!
$R_n(x)$, ln $\Gamma(x)$		$R_n(x)$, ln x!

for the asymptotic approximation

$$\ln \Gamma(x) \simeq (x - \tfrac{1}{2})\ln x + \tfrac{1}{2}\ln 2\pi - x$$

$$+ \frac{1}{12x} - \frac{1}{360x^3} + \frac{1}{1260x^5} - \frac{1}{1680x^7} + \frac{1}{1188x^9}$$

and the identity

$$x! = x\Gamma(x) = \Gamma(x + 1) \quad \text{or} \quad \ln x! = \ln \Gamma(x + 1)$$

For x > 70 or x > 69, the directly calculated values for $\Gamma(x)$ or x!, respectively, will exceed the calculator's dynamic range. In this case the printout will not include $\Gamma(x)$ or x!. The proper value for $\Gamma(x)$ or x! is then found from

$$\Gamma(x) \text{ or } x! \equiv (\text{mantissa}) \times 10^{(\text{exponent})} \equiv (K_4) \times 10^{(K_5)}.$$

For integer values of x, $\Gamma(x)$ or x! is accurate to the number of significant digits displayed. For small non-integer values of x the error in ln $\Gamma(x)$ or ln x! will be less than or equal to the first neglected term in asymptotic series; in this case the error is given by

$$\frac{-691}{360360(y)^{11}} < R_n(x) \leq 0 \ , \quad y = \begin{array}{l} x \text{ for } \ln \Gamma(x) \\ x + 1 \text{ for } \ln x! \end{array}$$

where $R_n(x)$ = [True value for ln $\Gamma(x)$ or ln x!]

- [calculated value for ln $\Gamma(x)$ or ln x!]

For $x > \simeq 3.5$, the error term $R_n(x)$ is not significant.

While the program will operate satisfactorily for positive $x < 1$, the error in the calculated value for $\Gamma(x)$ increases very rapidly as $x \to 0$, as shown by the error term.

References

1. A. Abramowitz and I. A. Stegun (editors), *Handbook of Mathematical Functions, AMS 55*, Dept. of Commerce, Washington, D.C., 1964.

NOTES

EXAMPLES

Find Γ(13)

13!

Γ(1.995)

Γ(89.5)

89.5!

```
GAMMA FUNCTIONS
 & FACTORIALS

ENTER X

FOR G(X), EXC 1
FOR X!, EXC 2

X =
          13.

G(X) =
   479001600.

LN G(X) =
   19.9872145

LOG G(X) =
 8.680336964

MAN, EXP:
     4.790016
           8.

***

X =
          13.

X! =
 6227020800.

LN X! =
 22.55216385

LOG X! =
 9.794280316

MAN, EXP:
    6.2270208
           9.

***

X =
        1.995

G(X) =
 .9978969209

LN G(X) =
-2.105293665E-03

LOG G(X) =
-9.143174215E-04

MAN, EXP:
     .9978969
           0.

RN(X) =
-9.624309239E-07

***
```

```
GAMMA FUNCTIONS
 & FACTORIALS

ENTER X

FOR G(X), EXC 1
FOR X!, EXC 2

X =
          89.5

LN G(X) =
 311.4071073

LOG G(X) =
 135.2423883

MAN, EXP:
    1.7473838
          135.

***

X =
          89.5

LN X! =
 315.9013459

LOG X! =
 137.1942114

MAN, EXP:
    1.5639085
          137.

***
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1		START	0	TITLE INSTRUCTIONS
2	x value			
3	For $\Gamma(x)$ For x!	EXC 1 EXC 2	0 0	x: K_0 *$\Gamma(x)$: K_1 ln $\Gamma(x)$: K_2 log $\Gamma(x)$: K_3 mantissa: K_4 exponent: K_5 #$R_n(x)$: K_6 or x: K_0 *x!: K_1 ln x!: K_2 log x!: K_3 mantissa: K_4 exponent: K_5 #$R_n(x)$: K_6
4	For a new x return to Step 2.			
	If the printer is not used, recall results from the indicated registers.			

*$\Gamma(x)$ or x! does not print when range has been exceeded.

#Error term prints only for $x < 3.5$.

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS LBLs
0	CLR	PF	G	A	M	M	A	SPC	F	U	
10	N	C	T	I	O	N	S	SPC	SPC	&	
20	SPC	F	A	C	T	O	R	I	A	L	
30	S	PF	PF	E	N	T	E	R	SPC	X	
40	PF	PF	F	O	R	SPC	G	(	X	)	
50	,	SPC	E	X	C	SPC	1	PF	F	O	
60	R	SPC	X	!	,	SPC	E	X	C	SPC	
70	2	PF	PF	STOP	LBL	1	=	x^a	1	=	1
80	K	Ø	X	SPC	=	PRNT	PF	CLFG	EXC	A	
90	LBL	2	=	x^a	1	=	K	Ø	X	SPC	2
1 00	=	PRNT	PF	SFG	+	1	LBL	A	=	K	A
10	9	x^2	1/x	×	1	4	Ø	÷	3	3	
20	-	3	=	÷	2	÷	K	9	x^2	+	
30	2	=	÷	7	÷	K	9	x^2	-	1	
40	=	÷	6	÷	K	9	x^2	+	5	=	
50	÷	6	Ø	÷	K	9	-	K	9	+	
60	K	9	ln	×	(	K	9	-	·	5	
70	)	+	(	2	×	π	)	ln	÷	2	
80	=	K	1	÷	1	Ø	ln	=	K	2	
90	int	=	K	5	K	1	e^x	=	K	3	
2 00	K	9	-	K	9	int	)	IF=Ø	K	3	
10	+	·	5	)	+/-	+/-	int	=	K	3	
20	CONT	CLR	1	Ø	x^a	(	K	2	-	K	
30	5	)	×	1	10^{oo}	7	+	·	5	=	
40	int	÷	1	10^{oo}	7	=	D/R	D/R	=	K	
	0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
											LBLs
50	4	CLR	+/-	6	9	1	÷	3	6	Ø	
60	3	6	Ø	÷	K	9	x^a	1	1	=	
70	K	6	K	3	1/x	K	3	IFFG	EXC	x!	
80	CONT	IFFL	CLR	EXC	IFFL	CONT	G	(	X	)	
90	SPC	=	PRNT	PF	LBL	IFFL	K	1	L	N	IFFL
3 00	SPC	G	(	X	)	SPC	=	PRNT	PF	K	
10	2	L	O	G	SPC	G	(	X	)	SPC	
20	=	PRNT	PF	EXC	10^{oo}	EXC	?	LBL	x!	IFFL	x!
30	CLR	EXC	IF≥Ø	CONT	X	!	SPC	=	PRNT	PF	
40	LBL	IF≥Ø	K	1	L	N	SPC	X	!	SPC	IF≥Ø
50	=	PRNT	PF	K	2	L	O	G	SPC	X	
60	!	SPC	=	PRNT	PF	EXC	10^{oo}	LBL	?	K	?
70	Ø	-	3	.	5	)	IF<Ø	R	N	(	
80	X	)	SPC	=	K	6	PRNT	PF	CONT	*	
90	*	*	CD	PF	RSET	LBL	10^{oo}	M	A	N	10^{oo}
4 00	,	SPC	E	X	P	:	K	4	PRNT	K	
10	5	PRNT	PF	RADR	GODP						
20											
30											
40											
50											
60											
70											
80											
90											
	0	1	2	3	4	5	6	7	8	9	

NOTES

Number Base Conversion: Base 10 to N

TITLE

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

$K_{\emptyset}$: new base N
K_1 : x, then R_{10}
K_2 : exponent for $K_{\emptyset}$
K_3 : base N digits
K_4 - K_8 : working
K_9 : number of decimal places desired, M

Subroutine labels used
1, 2
(A) , (B) , (C)

PROGRAM DESCRIPTION

This program converts any positive base 10 number x of 10 or fewer digits to its equivalent in any other base N where

$$N > 1$$

The resulting number is of the form

$$*MSD_n, \ldots, LSD_o \cdot MSDD_1, \ldots, LSDD_m; R_{10}$$

where R_{10} is the remainder of the base 10 number.

The number M of decimal digits DD to the right of the decimal point may be selected by the user; when that number of decimal digits has been calculated, the program will truncate, displaying R_{10}.

Whenever the remainder R_{10} is 0, the program will automatically truncate, displaying 0.

For integer values of x and N, the converted number will also be an integer, so the program will automatically truncate without printing any decimal digits. For integer x and N, the remainder R will typically be 0 as expected; however, in some cases round-off errors may cause a small remainder term.

Note that the new base N will typically be an integer, although the program will operate correctly with non-integer values.

* MSD is the Most Significant Digit and LSD the Least Significant Digit to the left of the decimal point. MSDD is the Most Significant Decimal Digit and LSDD the Least Significant Decimal Digit to the right of the decimal point.

EXAMPLES

1. Convert 459_{10} to base 8, with m = 4:

   $459_{10} = 713_8$; $R_{10} = 0$

2. Convert 459.99 to base 8, with m = 4:

   $459.99_{10} = 713.7727_8$

   $R_{10} = 9.76566 \times 10^{-6}$

3. Convert 27_{10} to base 2, with m = 4:

   $27_{10} = 11011_2$; $R_{10} = 0$

4. Convert 25.562_{10} to base 2, with m = 6:

   $25.562_{10} = 11001.100011_2$

   $R_{10} = 1.5125 \times 10^{-2}$

5. Convert $7.893 \times 10^3{}_{10}$ to base 16, m = 4:

   $7.893 \times 10^3{}_{10} = 1\ 14\ 13\ 5_{16}$

   $R_{10} = 0$

6. Convert 92.38_{10} to base e, m = 4:

   $92.38_{10} = 11210.0110_e$

   $R_{10} = 1.479666726 \times 10^{-2}$

```
BASE CONVERSION:

BASE 10 > BASE N
FOR N > 1, N=?
           8.

# OF DEC DIGITS
DESIRED, M=?
           4.

EXC 1: ENTR BASE
10 #; CONT

***

BASE 10 # = ?
         459.

           7.
           1.
           3.
            *

REMAINDER =
           0.

***

BASE 10 # = ?
        459.99

           7.
           1.
           3.
            *
           7.
           7.
           2.
           7.

REMAINDER =
 9.765660000E-06

***

N=?
           2.

M=?
           4.

***

BASE 10 # = ?
          27.

           1.
           1.
           0.
           1.
           1.
            *

REMAINDER =
           0.

***

N=?
           2.

M=?
           6.

***

BASE 10 # = ?
        25.562

           1.
           1.
           0.
           0.
           1.
            *
           1.
           0.
           0.
           0.
           1.
           1.

REMAINDER =
 1.512500000E-02

***

N=?
          16.

M=?
           4.

***

BASE 10 # = ?
 7.893000000E+03

           1.
          14.
          13.
           5.
            *

REMAINDER =
           0.

***

N=?
 2.718281828

M=?
           4.

***

BASE 10 # = ?
        92.38

           1.
           1.
           2.
           1.
           0.
            *
           0.
           1.
           1.
           0.

REMAINDER =
 1.479666726E-02

***
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1		START	0	Title N = ?
2	N, new base	CONT	0	N : $K_{\emptyset}$ M = ?
3	M, number of decimal digits desired	CONT	0	M : K_9 Instructions
4		EXC 1		
	Base 10 number	CONT	Remainder R_{10}	MSD : K_3 . . LSD * MSDD . . LSDD
5	To convert a new base 10 number to base N without changing M, return to Step 4. To change N or M without reprinting the Title and Instructions: EXC 2 and return to Step 2. If the printer is not used, replace PRNT at step 190 with STOP to view each digit as it is calculated. Press CONT to restart after each digit is calculated.			R_{10}

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
0	PF	CLR	B	A	S	E	SPC	C	O	N	
10	V	E	R	S	I	O	N	:	PF	B	
20	A	S	E	SPC	1	Ø	SPC	>	SPC	B	
30	A	S	E	SPC	N	F	O	R	SPC	N	
40	SPC	>	SPC	1	,	SPC	N	=	?	CLR	
50	STOP	=	K	Ø	PRNT	PF	int	-	K	Ø	
60	=	K	7	#	SPC	O	F	SPC	D	E	
70	C	SPC	D	I	G	I	T	S	PF	D	
80	E	S	I	R	E	D	,	SPC	M	=	
90	?	CLR	STOP	=	K	9	PRNT	PF	E	X	
1 00	C	SPC	1	;	SPC	E	N	T	R	SPC	
10	B	A	S	E	1	Ø	SPC	#	;	SPC	
20	C	O	N	T	PF	PF	*	*	*	CD	
30	PF	RSET	LBL	1	B	A	S	E	SPC	1	LBL 1
40	Ø	SPC	#	SPC	=	SPC	?	CLR	STOP	=	
50	K	1	PRNT	PF	ln	÷	K	Ø	ln	)	
60	int	=	K	2	K	1	int	-	K	1	
70	=	K	5	LBL	A	K	1	÷	K	Ø	LBL A
80	x^a	K	2	=	int	D/R	D/R	=	K	3	
90	PRNT*	K	2	IF=Ø	SPC	SPC	SPC	SPC	SPC	SPC	
2 00	SPC	SPC	SPC	SPC	SPC	*	CONT	CLR	1	=	
10	K	4	K	2	x^a	1	=	K	6	K	
20	2	=	K	8	-	1	=	K	2	LBL	
30	B	K	Ø	Π_4	K	6	-	1	=	K	LBL B
40	6	-	1	)	IF≥Ø	EXC	B	CONT	K	8	
	0	1	2	3	4	5	6	7	8	9	

* If the printer is not used, replace PRNT with STOP to view each digit in turn as it is calculated.

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
50	e^x	int	x^a	=	K	6	×	K	4	+	
60	K	4	1/x	×	(	1	-	K	6	=	
70	-	1	)	×	K	8	x^a	+	1	=	
80	×	K	3	+/-	)	Σ_1	K	8	-	1	
90	)	IF≥Ø	EXC	(A)	CONT	K	5	+	K	7	
00	)	IF≥Ø	EXC	(C)	CONT	K	8	+	K	9	
10	)	IF=Ø	EXC	(C)	CONT	K	1	IF=Ø	EXC	(C)	
20	CONT	EXC	(A)	LBL	(C)	PF	(R)	(E)	(M)	(A)	LBL (C)
30	(I)	(N)	(D)	(E)	(R)	(SPC)	(=)	K	1	10^{oo}	
40	=	K	1	PRNT	PF	(*)	(*)	(*)	CD	PF	
50	RSET	LBL	2	(N)	(=)	(?)	CLR	STOP	=	K	LBL 2
60	Ø	PRNT	PF	int	-	K	Ø	=	K	7	
70	(M)	(=)	(?)	CLR	STOP	=	K	9	PRNT	PF	
80	(*)	(*)	(*)	CD	PF	RSET					
90											
00											
10											
20											
30											
40											
50											
60											
70											
80											
90											
	0	1	2	3	4	5	6	7	8	9	

3

NOTES

Number Base Conversion: Base N to 10, N ⩾ 2

TITLE

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

$K_{\emptyset}$: base 10 integer
K_1 : base 10 decimal fraction
K_2 : base N digit, then combined base 10 number
K_3 : working
K_4 : working
K_5 : working
K_6 : base N

Subroutine labels used:
1, 2, 3, 4
(A) , (B)

PROGRAM DESCRIPTION

This program will convert any positive base N number of arbitrary length to its base 10 equivalent, within the display accuracy of the calculator, where the base N is an integer such that

$$N \geq 2$$

Given the base N number

$$*MSD_n, \ldots, LSD_o \cdot MSDD_1, \ldots, LSDD_m ,$$

the program calculates the sums and base 10 equivalents for each separate digit to the left of the decimal point beginning with LSD_o and proceeding to MSD_n. The program then separately calculates the base 10 equivalent for each separate digit of the fractional portion beginning with DD_1 and working to the right to DD_m.

Thus the integer and decimal fraction portions of the resulting base 10 number are accumulated separately, increasing the overall accuracy of the program. These results are printed separately and as a combined or mixed base 10 number.

* MSD_n is the Most Significant Digit and LSD_o the Least Significant Digit to the left of the decimal point. $MSDD_1$ is the Most Significant Decimal Digit and $LSDD_m$ is the Least Significant Decimal Digit to the right of the decimal point.

EXAMPLES

1. Convert the base 2 number

   100110011100.11100011101

   to base 10

   Results, Combined: 2460.88916

   displayed or K_2

   Integer: 2460. from $K_\emptyset$

   Fraction: .8891601563 from K_1

```
BASE CONVERSION:

BASE N > BASE 10

N=?
          2.

FOR INTEGER PART
EXC 1 & ENTER
DIGITS IN ORDER,
LSD TO MSD; THEN
FOR DECIMAL PART
EXC 2 & ENTER
DIGITS IN ORDER,
MSDD TO LSDD.

EXC 3 TO PRINT
RESULTS.

***
           0.
           0.
           1.
           1.
           1.
           0.
           0.
           1.
           1.
           0.
           0.
           1.
            *
           1.
           1.
           1.
           0.
           0.
           0.
           1.
           1.
           1.
           0.
           1.

BASE 10 RESULT

INT PART =
        2460.
DEC PART =
 .8891601563

TOTAL =
  2460.88916

***
```

2. Convert the base 16 number

   15 9 6 13 2 0 . 1 7 15 6 3

   to base 10

   Results, Combined: 16346400.09

   displayed or K_2

   Integer: 16346400 from $K_\emptyset$

   Fraction: $9.360027313 \times 10^{-2}$

   from K_1

```
N=?
          16.

***
           0.
           2.
          13.
           6.
           9.
          15.
            *
           1.
           7.
          15.
           6.
           3.

BASE 10 RESULT

INT PART =
    16346400.
DEC PART =
 9.3600273I3E-02

TOTAL =
 16346400.09

***
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1		START	0	Title N = ?
2	N	CONT	0	N : K_6 Instructions
3		EXC 1	0	
	LSD_o . . .	CONT . . .	0	LSD_o . . .
	MSD_n	CONT	0	MSD_n
4		EXC 2		
	$MSDD_1$ . . .	CONT . . .	0	$MSDD_1$ . . .
	$LSDD_m$	CONT	0	$LSDD_m$
5		EXC 3	Base 1Ø Total	Base 10 Result Int Part: $K_Ø$ Dec Part: K_1 Total: K_2

To convert a new base N number return to Step 3.

To change to a new base N' without printing the Title and Instructions: EXC 4, and return to Step 2.

If the printer is not used, recall results from the indicated registers.

PROGRAM STEPS

		0	1	2	3	4	5	6	7	8	9	COMMENTS
0	00	PF	CLR	B	A	S	E	SPC	C	O	N	
	10	V	E	R	S	I	O	N	:	PF	B	
	20	A	S	E	SPC	N	SPC	>	SPC	B	A	
	30	S	E	SPC	1	Ø	PF	N	=	?	CD	
	40	STOP	=	K	6	PRNT	PF	F	O	R	SPC	
	50	I	N	T	E	G	E	R	SPC	P	A	
	60	R	T	E	X	C	SPC	1	SPC	&	SPC	
	70	E	N	T	E	R	PF	D	I	G	I	
	80	T	S	SPC	I	N	SPC	O	R	D	E	
	90	R	,	L	S	D	SPC	T	O	SPC	M	
1	00	S	D	;	SPC	T	H	E	N	F	O	
	10	R	SPC	D	E	C	I	M	A	L	SPC	
	20	P	A	R	T	E	X	C	SPC	2	SPC	
	30	&	SPC	E	N	T	E	R	PF	D	I	
	40	G	I	T	S	SPC	I	N	SPC	O	R	
	50	D	E	R	,	M	S	D	D	SPC	T	
	60	O	SPC	L	S	D	D	·	PF	PF	E	
	70	X	C	SPC	3	SPC	T	O	SPC	P	R	
	80	I	N	T	PF	R	E	S	U	L	T	
	90	S	·	PF	PF	*	*	*	CD	PF	RSET	
2	00	LBL	1	CLR	=	K	3	=	K	Ø	=	LBL 1
	10	K	1	LBL	A	CLR	STOP	=	K	2	PRNT	LBL A
	20	×	K	6	x^a	K	3	+	·	5	=	
	30	int	Σ_0	K	3	+	1	=	K	3	EXC	
	40	A	LBL	2	CLR	=	K	1	1	=	K	LBL 2
		0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
50	4	SPC	SPC	SPC	SPC	SPC	SPC	SPC	SPC	SPC	
60	SPC	SPC	*	LBL	B	CLR	STOP	=	K	2	LBL B
70	PRNT	K	6	Π_4	K	4	1/x	×	K	2	
80	)	Σ_1	EXC	B	LBL	3	PF	B	A	S	LBL 3
90	E	SPC	1	Ø	SPC	R	E	S	U	L	
3 00	T	PF	PF	K	Ø	I	N	T	SPC	P	
10	A	R	T	SPC	=	PRNT	+	K	1	D	
20	E	C	SPC	P	A	R	T	SPC	=	PRNT	
30	PF	=	K	2	T	O	T	A	L	SPC	
40	=	PRNT	PF	*	*	*	PF	PF	RSET	LBL	
50	4	N	=	?	CD	STOP	=	K	6	PRNT	LBL 4
60	PF	*	*	*	CD	PF	RSET				
70											
80											
90											
00											
10											
20											
30											
40											
50											
60											
70											
80											
90											
	0	1	2	3	4	5	6	7	8	9	

NOTES

Number Base Conversion: Base N to 10, $2 \leqslant N \leqslant 9$

TITLE

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

PROGRAM DESCRIPTION

This program will convert any ten digit base N integer, decimal fraction, or mixed number to its base 10 equivalent, within the accuracy of the calculator. The base N must be an integer such that

$$2 \leq N \leq 9 .$$

REGISTERS USED

$K_{\emptyset}$: Base 10 number
K_1 : exponent, or position counter
K_2 : base N digits
K_3 : working
K_4 : working
K_5 : working
K_6 : base N
K_7 : base N number

Subroutine labels used:
1, 2
(A) , (B) , (C)

EXAMPLES

1. Convert the base 2 number

   110110.11 to base 10:

   Result: 54.75

2. Convert the base 8 number

   713.55 to base 10:

   Result: 459.703125

3. Convert the base 5 number

   -41330.14444 to base 10

   Result: -2715.39968

4. Convert these base 8 numbers to base 10:

Base 8	Base 10
107	71
-113	-75
127	87
130	88
071	57
-045	-37

```
BASE CONVERSION:

BASE N > BASE 10

FOR 2 <= N <= 9:
N=?
          2.

EXC 1; ENTER 10
DIGITS MAX; CONT

***

BASE N # = ?
   110110.11

BASE 10 RESULT
       54.75

***

N=?
          8.

***

BASE N # = ?
      713.55

BASE 10 RESULT
  459.703125

***

N=?
          5.

***

BASE N # = ?
-41330.14444

BASE 10 RESULT
- 2715.39968

***
```

```
N=?
          8.

***

BASE N # = ?
          107.

BASE 10 RESULT
          71.

***

BASE N # = ?
-         113.

BASE 10 RESULT
-          75.

***

BASE N # = ?
          127.

BASE 10 RESULT
          87.

***

BASE N # = ?
          130.

BASE 10 RESULT
          88.

***

BASE N # = ?
          71.

BASE 10 RESULT
          57.

***

BASE N # = ?
-          45.

BASE 10 RESULT
-          37.

***
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1		START	0	Title N = ?
2	N	CONT	0	N : K_6 Instructions
3		EXC 1		
	Base N number	CONT	Base 10 equivalent	Base N number: K_7 Base 10 equivalent: $K_\emptyset$
4	To convert a new base N number return to Step 3. To change to a new base N' without printing the Title and Instructions: EXC 2, and return to Step 2. If a printer is not used, recall results from the indicated registers.			

PROGRAM STEPS

		0	1	2	3	4	5	6	7	8	9	COMMENTS
0	00	PF	CLR	B	A	S	E	SPC	C	O	N	
	10	V	E	R	S	I	O	N	:	PF	B	
	20	A	S	E	SPC	N	SPC	>	SPC	B	A	
	30	S	E	SPC	1	Ø	PF	F	O	R	SPC	
	40	2	SPC	<	=	SPC	N	SPC	<	=	SPC	
	50	9	:	N	=	?	CLR	STOP	=	K	6	
	60	PRNT	PF	E	X	C	SPC	1	;	SPC	E	
	70	N	T	E	R	SPC	1	Ø	PF	D	I	
	80	G	I	T	S	SPC	M	A	X	;	SPC	
	90	C	O	N	T	PF	*	*	*	CLR	PF	
1	00	RSET	LBL	1	B	A	S	E	SPC	N	SPC	LBL 1
	10	#	SPC	=	SPC	?	CLR	STOP	=	K	7	
	20	PRNT	PF	x^a	1	=	K	5	int	=	K	
	30	2	CLR	=	K	Ø	=	K	1	LBL	A	LBL A
	40	K	2	÷	1	Ø	=	K	2	-	K	
	50	2	int	=	×	1	Ø	×	K	6	x^a	
	60	K	1	+	.	5	=	int	Σ_0	CD	1	
	70	Σ_1	K	2	int	=	K	2	x^a	-	1	
	80	)	IF≥Ø	EXC	A	CONT	CLR	=	K	1	K	
	90	5	-	K	5	int	=	K	2	LBL	B	LBL B
2	00	K	2	×	1	Ø	=	K	2	CD	1	
	10	=	K	4	K	1	=	K	3	LBL	C	LBL C
	20	K	6	Π_4	K	3	-	1	=	K	3	
	30	IF≥Ø	EXC	C	CONT	CD	1	Σ_1	K	4	1/x	
	40	×	K	2	int	)	Σ_0	K	2	-	K	

	0	1	2	3	4	5	6	7	8	9	COMMENTS
50	2	int	=	K	2	x^a	-	1	)	IF≥Ø	
60	EXC	(B)	CONT	K	7	10^{oo}	=	10^{oo}	Ø	Ø	
70	e^x	int	x^a	×	2	-	1	=	×	K	
80	Ø	=	K	Ø	(B)	(A)	(S)	(E)	(SPC)	(1)	
90	Ø	(SPC)	(R)	(E)	(S)	(U)	(L)	(T)	PRNT	PF	
3 00	(*)	(*)	(*)	PF	PF	RSET	LBL	2	(N)	(=)	LBL 2
10	(?)	CD	STOP	=	K	6	PRNT	PF	(*)	(*)	
20	(*)	CD	PF	RSET							
30											
40											
50											
60											
70											
80											
90											
00											
10											
20											
30											
40											
50											
60											
70											
80											
90											
	0	1	2	3	4	5	6	7	8	9	

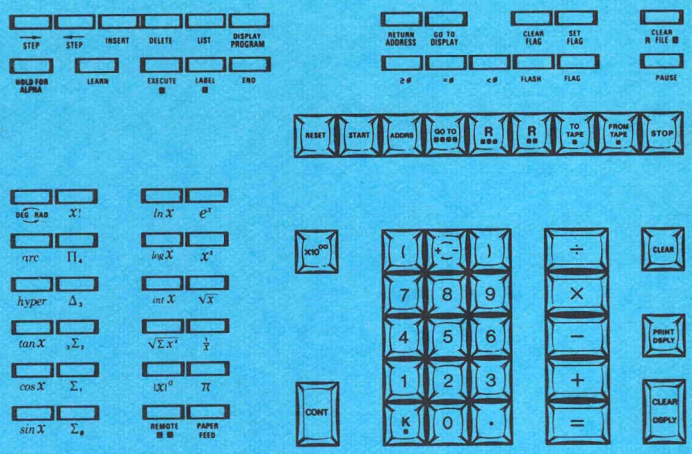
NOTES

SECTION 3 POLYNOMIALS

OCTAL CODE	PRINT OUT	KEY	SYMBOL
001	LBL_	LABEL ■	LBL
002	FTP_	FROM TAPE ■	FTP
003	TTP_	TO TAPE ■	TTP
004	EXC_	EXECUTE ■	EXC
005	CFI_	CLEAR R FILE ■	CFI
007	GOTO	GO TO ■■■■	GT
010	R___	R ■■■	Rxxx
011	R__	R ■■	Rxx
040	CLDP	CLEAR DSPLY	CD
041	IFFL	FLASH	IFFL
042	S FG	SET FLAG	SFG
043	STOP	STOP	STOP
044	PRNT	PRINT DSPLY	PRNT
045	CLR	CLEAR	CLR
046	PAUS	PAUSE	PAUS
047	CLFG	CLEAR FLAG	CLFG
050	(	(	(
051	)	)	)
052	*	×	×
053	+	+	+
054	RSET	RESET	RSET
055	-	−	-
056	.	.	.
057	/	÷	÷
060	0	0	ø
061	1	1	1
062	2	2	2
063	3	3	3
064	4	4	4
065	5	5	5
066	6	6	6
067	7	7	7
070	8	8	8
071	9	9	9
072	ADR	ADDRS	ADR
073	STRT	START	STRT
074	IF<0	<ø	IF<ø
075	=	=	=
076	IF>=	≥ø	IF≥ø
077	IF=0	=ø	IF=ø

OCTAL CODE	PRINT OUT	KEY	SYMBOL
100	K_	K ■	K
101	DG/R	DEG RAD	D/R
102	ARC	*arc*	arc
103	HYP	*hyper*	hyp
104	TAN	$\tan x$	tan
105	COS	$\cos x$	cos
106	SIN	$\sin x$	sin
107	X!	$x!$	x!
110	PI4	Π_4	Π_4
111	DLT3	Δ_3	Δ_3
112	3SM2	${}_3\Sigma_2$	${}_3\Sigma_2$
113	SUM1	Σ_1	Σ_1
114	SUM0	$\Sigma_ø$	Σ_0
115	LN	$\ln x$	ln
116	LOG	$\log x$	log
117	INT	*int* x	int
120	RSSQ	$\sqrt{\Sigma x^2}$	$\sqrt{\Sigma x^2}$
121	X↑A	$\|x\|^a$	x^a
122	RM__	REMOTE ■ ■	RMT
123	E↑X	e^x	e^x
124	X↑2	x^2	x^2
125	SQRT	$\sqrt{x}$	$\sqrt{x}$
126	1/X	$\frac{1}{x}$	1/x
127	PI	π	π
130	PAPR	PAPER FEED	PF
131	*10↑	$\times 10^{oo}$	10^{oo}
132	CONT	CONT	CONT
133	R AD	RETURN ADDRESS	RADR
134	+/-	+/-	+/-
135	GODP	GO TO DISPLAY	GODP
136	IFFG	FLAG	IFFG
137	ENDR	END	ENDR

OCTAL CODE	PRINT OUT	KEY	SYMBOL
201		LABEL ■	LBL
202		FROM TAPE ■	FTP
203		TO TAPE ■	TTP
204		EXECUTE ■	EXC
205		CLEAR R FILE ■	CFI
207		BELL	BELL
210		SPACE ←	SPC←
211		TAB ←	TAB
220		STEP	STP←
221		INSERT	NSRT
222		DELETE	DLT
223		→ STEP	STP→
224		LIST	LIST
225		DISPLAY PROGRAM	DPRG
226		LEARN	LRN
240		SPACE	SPC
241	!	!	!
242	"	"	"
243	#	#	#
244	$	$	$
245	%	%	%
246	&	&	&
247	'	'	'
250	(	(	(
251	)	)	)
252	*	*	*
253	+	+	+
254	,	,	,
255	-	-	-
256	.	.	.
257	/	/	/
260	0	0	Ø
261	1	1	1
262	2	2	2
263	3	3	3
264	4	4	4
265	5	5	5
266	6	6	6
267	7	7	7
270	8	8	8
271	9	9	9
272	:	:	:
273	;	;	;
274	<	<	<
275	=	=	=
276	>	>	>
277	?	?	?



OCTAL CODE	PRINT OUT	KEY	SYMBOL
300	@	@	@
301	A	A	A
302	B	B	B
303	C	C	C
304	D	D	D
305	E	E	E
306	F	F	F
307	G	G	G
310	H	H	H
311	I	I	I
312	J	J	J
313	K	K	K
314	L	L	L
315	M	M	M
316	N	N	N
317	O	O	O
320	P	P	P
321	Q	Q	Q
322	R	R	R
323	S	S	S
324	T	T	T
325	U	U	U
326	V	V	V
327	W	W	W
330	X	X	X
331	Y	Y	Y
332	Z	Z	Z
333	[	[	[
334	\	\	\
335	]	]	]
336	↑	↑	↑
337	_	_	_

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

$K_{\emptyset}$: x_0
K_1 : working
K_2 : working
K_3 : A
K_4 : B
K_5 : C
K_6 : D
K_7 : x_1, or R
K_8 : x_2, or I

Subroutine labels used:
$\emptyset$, 1, 2, 3, 4, $\sqrt{x}$

PROGRAM DESCRIPTION

This program solves the quadratic equation

$$Ax^2 + Bx + C = 0$$

or the cubic equation

$$Ax^3 + Bx^2 + Cx + D = 0$$

where the coefficients A, B, C, and D are real, and $A \neq 0$.

The quadratic equation is solved for roots

$$x_{1,\,2} = -\frac{b}{2a} \pm \sqrt{\left(\frac{b}{2a}\right)^2 - \frac{c}{a}}$$

For the cubic equation, the program solves for one real root x_0 by an iterative algorithm, then reduces the equation to a quadratic, which is then solved for x_1 and x_2.

For the quadratic the character of the roots is indicated by the discriminant

$$d = \left(\frac{b}{2a}\right)^2 - \frac{c}{a}$$

For $d > 0$: x_1 and x_2 are real and unequal.

For $d = 0$: x_1 and x_2 are real and equal.

For $d < 0$: x_1 and x_2 are the complex conjugate pair

$$x_1 = R + jI, \quad x_2 = R - jI$$

This latter condition is indicated by a flashing display at the conclusion of the program.

EXAMPLES

1. $2x^2 + 3x - 7 = 0$

```
QUADRATIC &
 CUBIC EQUATION
  SOLUTIONS:

FOR QUAD EXC 2;
FOR CUBIC EXC 3:


A*X↑2+B*X+C=0

ENTER A,B,C

             2.
             3.
-            7.


REAL ROOTS X1,X2

 1.265564437
-2.765564437

END
```

2. $2x^2 + 3x + 7 = 0$

```
A*X↑2+B*X+C=0

ENTER A,B,C

             2.
             3.
             7.


COMPLEX ROOTS
  R,I#; R,-I#:

-          .75
  1.71391365#

-          .75
- 1.71391365#

END
```

3. $x^2 - 25 = 0$

```
A*X↑2+B*X+C=0

ENTER A,B,C

             1.
             0.
-           25.


REAL ROOTS X1,X2

             5.
-            5.

END
```

4. $3x^2 + 5x = 0$

```
A*X↑2+B*X+C=0

ENTER A,B,C

             3.
             5.
             0.


REAL ROOTS X1,X2

-1.000000000E-12
-1.666666667

END
```

5. $x^3 + 2x^2 + 10x - 20 = 0$

```
A*X↑3+B*X↑2+C*X
  +D=0

ENTER A,B,C,D

             1.
             2.
            10.
-           20.

REAL ROOT X0:

 1.368808108

COMPLEX ROOTS
  R,I#; R,-I#:

-1.684404054
  3.43133135#

-1.684404054
- 3.43133135#

END
```

6. $x^3 - 6x^2 + 11x - 6 = 0$

```
A*X↑3+B*X↑2+C*X
  +D=0

ENTER A,B,C,D

             1.
-            6.
            11.
-            6.

REAL ROOT X0:

             1.

REAL ROOTS X1,X2

             3.
             2.

END
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1		START	0	
2a	For QUADRATIC	EXC 2	0	see examples
b	or for CUBIC	EXC 3	0	for printout details
3	A A	CONT	0	
	B or B	CONT	0	
	C C	CONT	0	
	D	CONT	0 (may flash)	
4	If a printer is not used, recall results from registers as follows			
	$K_{\emptyset}$: x_0 (CUBIC only)			
	K_7 : x_1 real root			
	K_8 : x_2 real root			
	or			
	K_7 : R complex			
	K_8 : I roots			
	If final display is flashing, roots are complex.			
5	To solve a new equation, of either degree, return to Step 2.			

PROGRAM STEPS

		0	1	2	3	4	5	6	7	8	9	COMMENTS
0	00	PF	PF	EXC	Ø	LBL	1	K	Ø	+	K	LBL 1
	10	4	=	×	K	Ø	+	K	5	=	×	Iterative
	20	K	Ø	+	K	6	=	10^{OO}	=	10^{OO}	Ø	cubic
	30	Ø	e^x	int	x^a	×	2	−	1	−	K	algorithm
	40	1	=	x^a	×	2	+/−	+	1	=	K	
	50	3	×	K	1	=	K	1	K	3	×	
	60	K	2	×	2	x^a	(	K	3	−	1	
	70	=	K	2	Σ_0	K	Ø	−	K	2	)	
	80	−	K	Ø	)	x^a	−	1	)	IF≥Ø	EXC	
	90	1	CONT	PF	R	E	A	L	SPC	R	O	Print real
1	00	O	T	SPC	X	Ø	:	K	Ø	PF	PRNT	x_0
	10	PF	+	K	4	=	K	4	×	K	Ø	
	20	+	K	5	=	K	5	CD	1	=	K	
	30	3	EXC	$\sqrt{x}$	EXC	4	LBL	$\sqrt{x}$	K	4	÷	LBL $\sqrt{x}$
	40	2	÷	K	3	=	K	7	x^2	−	K	Quadratic
	50	5	÷	K	3	=	K	8	IF≥Ø	R	E	algorithm
	60	A	L	SPC	R	O	O	T	S	SPC	X	
	70	1	,	X	2	PF	$\sqrt{x}$	=	K	8	−	Print real
	80	K	7	=	K	7	PRNT	−	2	×	K	x_1
	90	8	=	K	8	PRNT	RADR	GODP	CONT	C	O	x_2
2	00	M	P	L	E	X	SPC	R	O	O	T	
	10	S	K	7	SPC	SPC	R	,	I	#	;	
	20	SPC	R	,	−	I	#	:	+/−	=	K	Print complex
	30	7	PF	PRNT	K	8	$\sqrt{x}$	=	K	8	PRNT	R, I
	40	PF	CLR	K	7	PRNT	CD	1/x	K	8	+/−	R, −I
		0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
50	PRNT	RADR	GODP	LBL	Ø	Q	U	A	D	R	LBL Ø
60	A	T	I	C	SPC	&	CLR	SPC	C	U	Introduction
70	B	I	C	SPC	E	Q	U	A	T	I	
80	O	N	SPC	SPC	SPC	S	O	L	U	T	
90	I	O	N	S	:	PF	PF	F	O	R	
3 00	SPC	Q	U	A	D	SPC	E	X	C	SPC	
10	2	;	SPC	F	O	R	SPC	C	U	B	
20	I	C	SPC	E	X	C	SPC	3	:	STOP	
30	RSET	LBL	2	PF	PF	A	*	X	↑	2	LBL 2
40	+	B	*	X	+	C	=	Ø	CLR	PF	Quadratic
50	E	N	T	E	R	SPC	A	,	B	,	Data input
60	C	STOP	PF	=	K	3	PRNT	CD	STOP	=	
70	K	4	PRNT	CD	STOP	=	K	5	PRNT	PF	
80	PF	EXC	$\sqrt{x}$	EXC	4	LBL	3	PF	PF	A	LBL 3
90	*	X	↑	3	+	B	*	X	↑	2	Cubic
4 00	+	C	*	X	SPC	SPC	SPC	+	D	=	Data input
10	Ø	CLR	PF	E	N	T	E	R	SPC	A	
20	,	B	,	C	,	D	STOP	PF	=	K	
30	3	PRNT	CD	STOP	=	PRNT	÷	K	3	=	
40	K	4	CD	STOP	=	PRNT	÷	K	3	=	
50	K	5	CD	STOP	=	PRNT	÷	K	3	=	
60	K	6	x^a	1	÷	K	6	=	IFFL	CLR	
70	1	CONT	=	K	1	CD	-	K	6	=	
80	K	Ø	=	K	2	EXC	1	LBL	4	PF	LBL 4
90	E	N	D	CD	PF	RSET					END
	0	1	2	3	4	5	6	7	8	9	

NOTES

3rd-Order Polynomial Interpolation

TITLE

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

$K_{\emptyset}$: y_0
K_1 : y_1
K_2 : y_2
K_3 : y_3
K_4 : x_0
K_5 : x_3
K_6 : x_i
K_7 : working, then y_i

Subroutines used:
LBL Ø

PROGRAM DESCRIPTION

Given four value pairs for a function

$$y = g(x)$$

at equally spaced values for x such that

$$y_0 = g(x_0)$$

$$y_1 = g(x_1)$$

$$y_2 = g(x_2)$$

$$y_3 = g(x_3)$$

where the interval h between successive values of x is a positive real number

$$h = x_j - x_{j-1} \quad \text{or} \quad h = \frac{x_3 - x_0}{3},$$

then for any x_i, this program evaluates the unique third-order polynomial

$$y_i = P(x_i)$$

that satisfies the four given points.

The values for y_0 through y_3 must be entered in ascending order; the values x_0 through x_3 must be equally spaced.

The program finds a value y_i for any x_i entered; however, it is most accurate in the range $x_0 < x_i < x_3$. For many functions extrapolation beyond this range will give grossly inaccurate values for y_i.

The program prints out input data and results.

EXAMPLES

Given the data pairs,

$x_o = 0$ $\quad y_o = -1$
$x_1 = 1$ $\quad y_1 = 1$
$x_2 = 2$ $\quad y_2 = 2$
$x_3 = 3$ $\quad y_3 = 0$

find y values for these x_i:

$x_i = .125$
$x_i = .750$
$x_i = .933$
$x_i = 1.383$
$x_i = 2.5$
$x_i = 3.$

```
THIRD-ORDER
 POLYNOMIAL
  INTERPOLATION:

ENTER INITIAL
  DATA:

Y0=
-.          1.
Y1=
            1.
Y2=
            2.
Y3=
            0.

X0=
            0.
X3=
            3.

INTERPOLATIONS:

ENTER X VALUES;
  PRESS CONT

X=
          .125
Y=
- .763671875

X=
           .75
Y=
      .515625

X=
          .933
Y=
  .875022421

X=
         1.383
Y=
 1.610094871

X=
          2.5
Y=
          1.5

X=
           3.
Y=
           0.

END
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1		START		TITLE
				INSTRUCTIONS
2	y_0	CONT	0	y_0
	y_1	CONT	0	y_1
	y_2	CONT	0	y_2
	y_3	CONT	0	y_3
	x_0	CONT	0	x_0
	x_3	CONT	0	x_3
3	x_i	CONT	y_i	x_i, y_i
	.	.	.	.
	.	.	.	.
	.	.	.	.
	.	.	.	.
	x_{n-1}	CONT	y_{n-1}	x_{n-1}, y_{n-1}
4	For the last point	SET FLAG		
5	x_n	CONT	y_n	x_n, y_n
				END
	When a printer is not used, recall y_i from K_7, or for automatic display of y_i, follow the instructions on the PROGRAM STEPS pages.			See examples for printout detail.
	For a new x_0 to x_3 interval return to Step 1.			

		0	1	2	3	4	5	6	7	8	9	COMMENTS
0	00	PF	PF	T	H	I	R	D	-	O	R	
	10	D	E	R	CLR	SPC	P	O	L	Y	N	
	20	O	M	I	A	L	PF	SPC	SPC	I	N	
	30	T	E	R	P	O	L	A	T	I	O	
	40	N	:	PF	PF	E	N	T	E	R	SPC	
	50	I	N	I	T	I	A	L	PF	SPC	SPC	
	60	D	A	T	A	:	PF	PF	Y	Ø	=	
	70	STOP	=	K	Ø	PRNT	Y	1	=	CD	STOP	
	80	=	K	1	PRNT	Y	2	=	CD	STOP	=	
	90	K	2	PRNT	Y	3	=	CD	STOP	=	K	
1	00	3	PRNT	PF	X	Ø	=	CD	STOP	=	K	
	10	4	PRNT	X	3	=	CD	STOP	=	K	5	
	20	PRNT	PF	PF	I	N	T	E	R	P	O	
	30	L	A	T	I	O	N	S	:	SPC	PF	
	40	E	N	T	E	R	SPC	X	SPC	V	A	
	50	L	U	E	S	;	PF	SPC	SPC	P	R	
	60	E	S	S	SPC	C	O	N	T	LBL	Ø	LBL Ø
	70	PF	PF	X	=	CD	STOP	=	K	6	PRNT	Interpolation
	80	Y	=	-	K	4	)	÷	(	K	5	
	90	-	K	4	)	×	3	=	K	7	K	
2	00	3	÷	(	K	7	-	3	)	-	K	
	10	Ø	÷	K	7	=	÷	3	+	K	1	
	20	÷	(	K	7	-	1	)	-	K	2	
	30	÷	(	K	7	-	2	=	÷	2	×	
	40	K	7	×	(	K	7	-	1	)	×	
		0	1	2	3	4	5	6	7	8	9	

	0	1	2	3	4	5	6	7	8	9
50	(	K	7	-	2	)	×	(	K	7
60	-	3	=	K	7	K	6	-	K	5
70	)	IF=Ø	K	3	=	K	7	CONT	K	6
80	-	K	4	)	IF=Ø	K	Ø	=	K	7
90	CONT	CLR	K	7	PRNT	IFFG	PF	PF	(E)	(N)
3 00	(D)	PF	PF	RSET	CONT	CONT*	EXC	Ø		
10										
20										
30										
40										
50										
60										
70										
80										
90										
00										
10										
20										
30										
40										
50										
60										
70										
80										
90										
	0	1	2	3	4	5	6	7	8	9

COMMENTS

*If a printer is not used, replace CONT with STOP to view calculated value for y_i.

NOTES

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

$K_{\emptyset}$: x
K_1 : not used
K_2 : not used
K_3 : not used
K_4 : f(x)
K_5 : working
K_6 : logic
K_7 : logic
K_8 : logic
K_9 : working

$R_{\emptyset\emptyset\emptyset}$: ind. address
$R_{\emptyset 1}$: working
$R_{\emptyset 2}$: working

Subroutine labels used:
STRT, 1, 2, 3
(A) through (G)

PROGRAM DESCRIPTION

This program offers two simple methods for finding real roots for the equation f(x) = 0, as well as a routine for evaluating f(x) for any real x.

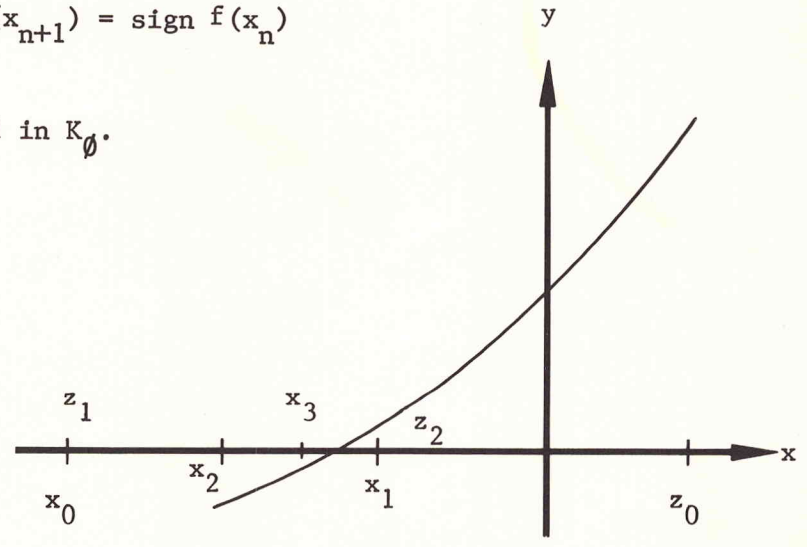
LBL 1: Real roots by the Halving Method.

This method is straightforward and surefire but does require knowledge of the approximate location of the roots. Given an initial interval in which the curve crosses the axis, this method repeatedly halves the interval in which the intersection occurs. The initial interval (x_0, z_0) must be given such that the function crosses the axis an odd number of times in the interval. The formula used is:

$$x_{n+1} = \frac{x_n + z_n}{2}$$

$$z_{n+1} = \begin{cases} x_n : \text{If sign } f(x_{n+1}) = \text{sign } f(z_n) \\ z_n : \text{If sign } f(x_{n+1}) = \text{sign } f(x_n) \end{cases}$$

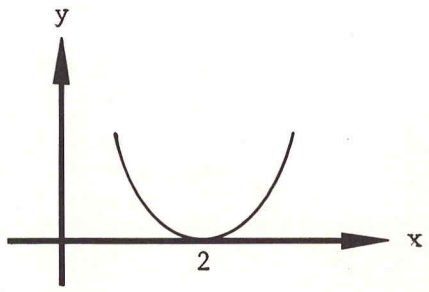
The calculated root is stored in $K_{\emptyset}$.



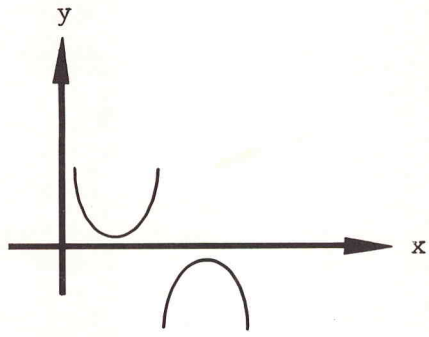
In case the number of roots and/or their approximate locations are not known, Step C, shown in the Program Execution Section, should be done. Omit this, however, and continue to Step D if the number and approximate locations of the roots are known.

Two considerations should be taken into account to avoid errors.

1. The program will not find the correct roots of a function which may have real roots but never crosses the x axis, such as $y = (x - 2)^2$. In this case the correct root will be found only if the root is entered as one of the initial values.



2. The program will give an erroneous result for a function which may seem to cross the x axis but never really does, such as y = sec x.



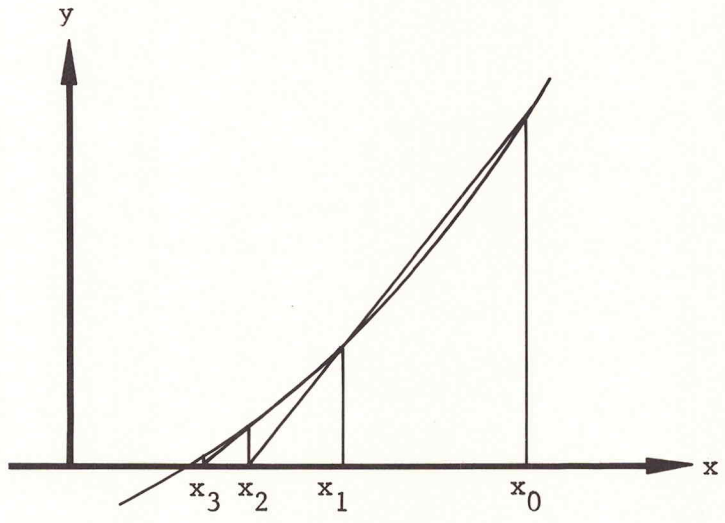
Any calculated result can be checked to see whether it really is a root by pressing EXC 3, KØ, CONT. If the result stored in KØ is a root, the calculator will print Ø (or a small residual) for f(x).

LBL 2: Real roots by the Secant Method.

This routine finds roots for the equation f(x) = 0 by generating successive approximations which converge to the desired root, providing appropriate initial approximations, x_0 and x_1, have been entered. The recurrence formula used is

$$x_{n+1} = x_n - \frac{(x_{n-1} - x_n)\, f(x_n)}{f(x_{n-1}) - f(x_n)}$$

The calculated root is stored in KØ:



Two considerations should be taken into account to avoid error.

1. If the calculator finds the same values for successive $f(x_n)$ and $f(x_{n-1})$, but different values for x_n and x_{n-1}, two conditions are possible:
   a) x_n and x_{n-1} are sufficiently close in value such that the calculator can not distinguish between them in subsequent calculations. Hence x_n will be taken to be a close approximation to the desired root by the calculator.
   b) x_n and x_{n-1} differ by more than 1×10^{-6}. In this case the calculator in the course of its calculations has happened to evaluate two points on the opposite sides of a minima or a maxima, at which the values of the function are equal. In this case the program prints out a "flashing" Ø. Restart the program with different initial values.

2. In case the programmed function has no roots, the calculator will stay in the "busy" mode searching for a non-existent root. It will only terminate execution if the "Stop" key is pressed.

LBL 3: Function Evaluation.

This routine calculates and prints the value for f(x) for any real x entered. This routine may be used to search for approximate locations of roots prior to using routines LBL 1 and LBL 2, or it may be used independently to generate values for graphing the function f(x).

NOTES

EXAMPLES

Find the roots of the function

$$f(x) = x^3 - 2x^2 - x + 2 = 0.$$

First we must program f(x) as f(KØ), that is,

$$f(K\emptyset) = (K\emptyset)^2 \times K\emptyset - 2(K\emptyset)^2 - K\emptyset + 2.$$

To enter the programmed function, press RSET LRN , and then enter f(KØ) as

0000	CLR	F0			
0001	K_	F0			
0002	0	F0	0011	0	F0
0003	X↑2	F0	0012	X↑2	F0
0004	*	F0	0013	-	F0
0005	K_	F0	0014	K_	F0
0006	0	F0	0015	0	F0
0007	-	F0	0016	+	F0
0008	2	F0	0017	2	F0
0009	*	F0	0018	=	F0
0010	K_	F0	0019	EXC_	F0
			0020	0	F0

Press LRN to exit the LEARN mode.

EXC STRT to print Title.

EXC 3 to search for x-axis crossings.

Try x = -1.5, 0, 1.5, 3

EXC 1 to find roots using Halving Method:

Try $x_0 = 1.5$, $z_0 = 3$ Root = 2

Try $x_0 = 1.5$, $z_0 = .5$ Root = 1

Try $x_0 = -1.5$, $z_0 = 0$ Root = -1

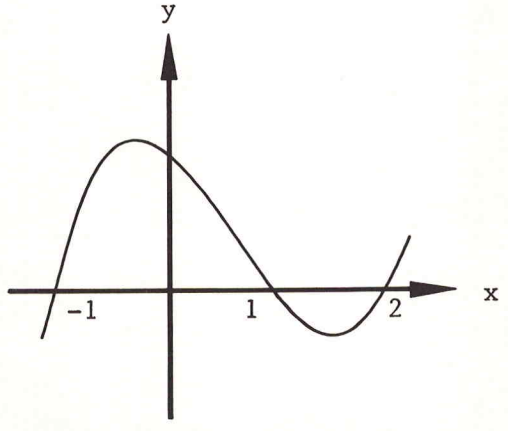
EXC 2 to find roots using Secant Method:

Try $x_0 = 20$, $x_1 = 10$ Root = 2

Try $x_0 = 0$, $x_1 = .5$ Root = 1

Try $x_0 = -20$, $x_1 = -10$ Root = -1

```
ROOT FINDERS:

HALVING, EXC 1
SECANT, EXC 2
FNCT EVAL, EXC 3
                    X0=?
**                  -          1.5
                    Z0=?
X=??  THEN F(X)=              0.

-         1.5       ROOT =
--      4.375       -          1.

#                   **

          0.
          2.

#

          1.5
-         .625      X0=?
                              20.
#                   X1=?
                              10.
          3.
          8.        ROOT =
                              2.
#
                    **

X0=?                X0=?
          1.5                 0.
Z0=?                X1=?
          3.                  .5

ROOT =              ROOT =
          2.                  1.

**                  **

X0=?                X0=?
          1.5       -         20.
Z0=?                X1=?
          .5        -         10.

ROOT =              ROOT =
          1.        -          1.

**                  **
```



PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
A	To program your function f(x) as f(KØ):			
1		RSET LRN		
	Enter f(KØ)	= EXC Ø LRN		
B	To print Title	EXC START	0	Title
C	To search for x-axis crossings, or to evaluate f(x) for various x:			
1		EXC 3	0	x = ?; F(x) =
2	trial x	CONT	f(x)	trial x
	new x	CONT	f(x)	f(x)
	for as many x as desired.			new x
	If f(x) changes sign between any two x's, at least one root exists between them.			f(x) and so on
D	To find a root of f(x) between two initial values of x found in C:			
1	for Halving Method	EXC 1	0	
2	x_0	CONT		x_0
	z_0	CONT	Root	z_0 Root
	Repeat D for finding additional roots.			
E	To find a root using the Secant Method when approximate locations are difficult to find:			
1		EXC 2	0	
2	x_0	CONT		x_0
	x_1	CONT	Root	x_1 Root
	E may be repeated using new x_0 and x_1; occasionally the program will find the same root from several starting points.			

PROGRAM STEPS

COMMENTS

	0	1	2	3	4	5	6	7	8	9	
0				Program your function							
00		f(x) as f(KØ). Finish with the sequence									
10		"= EXC Ø." You may use registers K1,									
20		K2, and K3 for intermediate storage as									
30		necessary. As you program the function									
40		f(x) = f(KØ), do not modify the contents									
50		of register KØ. You may use steps 000									
60		through 099 for your function.									
70											
80											
90											
1	LBL	Ø	=	K	4	K	8	IF=Ø	K	4	LBL Ø
00	PRNT	PF	(#)	PF	EXC	(G)	CONT	K	6	IF≥Ø	
10	EXC	(B)	CONT	K	7	IF≥Ø	EXC	(A)	CONT	K	
20	4	×	K	9	10^{oo}	=	10^{oo}	Ø	Ø	e^x	
30	int	x^a	+	1	=	Rxxx	Ø	Ø	Ø	K	
40	Ø	=	Rxx	Rxx	Ø	Ø	Rxx	Ø	1	Ø	
50	-	Rxx	Ø	2	Ø	)	IF=Ø	EXC	(E)	CONT	
60	LBL	(A)	Rxx	Ø	1	+	Rxx	Ø	2	)	LBL (A)
70	÷	2	=	K	Ø	K	7	+/-	IF≥Ø	STRT	
80	CONT	=	K	7	K	4	=	K	9	STRT	
90	LBL	(B)	IFFG	K	Ø	=	Rxx	Ø	1	K	LBL (B)
2	4	=	Rxx	Ø	2	CD	(X)	(1)	(=)	(?)	
00	STOP	=	K	Ø	PRNT	CLFG	STRT	CONT	K	4	
10	-	Rxx	Ø	2	)	IF=Ø	EXC	(C)	CONT	K	
20	4	×	(	K	Ø	-	Rxx	Ø	1	)	
30	0	1	2	3	4	5	6	7	8	9	
40											

PROGRAM STEPS

		0	1	2	3	4	5	6	7	8	9	COMMENTS
	50	÷	(	Rxx	Ø	2	-	K	4	)	+	
	60	K	Ø	=	K	5	K	Ø	=	Rxx	Ø	
	70	1	K	5	=	K	Ø	K	4	=	Rxx	
	80	Ø	2	STRT	LBL	1	PF	X	Ø	=	?	LBL 1
	90	CLR	STOP	=	Rxxx	Ø	Ø	1	PRNT	Z	Ø	
3	00	=	?	CLR	STOP	=	Rxx	Ø	2	=	K	
	10	Ø	PRNT	π	=	K	7	=	K	8	+/-	
	20	=	K	6	CLFG	STRT	LBL	2	PF	X	Ø	LBL 2
	30	=	?	CLR	STOP	=	K	Ø	PRNT	π	=	
	40	K	6	=	K	8	SFG	STRT	LBL	3	PF	LBL 3
	50	X	=	?	;	SPC	T	H	E	N	SPC	
	60	F	(	X	)	=	PF	LBL	G	PF	CLR	LBL G
	70	STOP	=	K	Ø	PRNT	CD	=	K	8	CLFG	
	80	STRT	LBL	C	Rxx	Ø	1	-	K	Ø	)	LBL C
	90	EXC	D	K	4	EXC	D	EXC	E	LBL	D	LBL D
4	00	x^a	1	-	1	10^{oo}	+/-	6	)	IF≥Ø	CD	
	10	1/x	CD	PF	EXC	F	CONT	RADR	GODP	LBL	E	LBL E
	20	PF	R	O	O	T	SPC	=	K	Ø	LBL	
	30	F	PRNT	PF	*	*	PF	RSET	LBL	STRT	PF	LBL F , STRT
	40	R	O	O	T	SPC	F	I	N	D	E	
	50	R	S	:	PF	PF	H	A	L	V	I	
	60	N	G	,	SPC	E	X	C	SPC	1	PF	
	70	S	E	C	A	N	T	,	SPC	E	X	
	80	C	SPC	2	PF	F	N	C	T	SPC	E	
	90	V	A	L	,	SPC	E	X	C	SPC	3	
		0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

COMMENTS

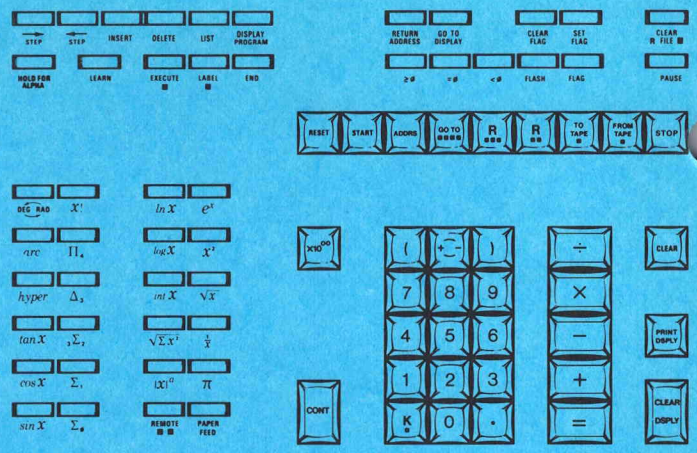
		0	1	2	3	4	5	6	7	8	9
5	00	PF	*	*	PF	RSET					
	10										
	20										
	30										
	40										
	50										
	60										
	70										
	80										
	90										
	00										
	10										
	20										
	30										
	40										
	50										
	60										
	70										
	80										
	90										
	00										
	10										
	20										
	30										
	40										
		0	1	2	3	4	5	6	7	8	9

SECTION 4 SIMULTANEOUS EQUATIONS AND MATRICES

OCTAL CODE	PRINT OUT	KEY	SYMBOL
001	LBL_	LABEL ■	LBL
002	FTP_	FROM TAPE ■	FTP
003	TTP_	TO TAPE ■	TTP
004	EXC_	EXECUTE ■	EXC
005	CFI_	CLEAR R FILE ■	CFI
007	GOTO	GO TO ■■■■	GT
010	R___	R ■■■	Rxxx
011	R__	R ■■	Rxx
040	CLDP	CLEAR DSPLY	CD
041	IFFL	FLASH	IFFL
042	S FG	SET FLAG	SFG
043	STOP	STOP	STOP
044	PRNT	PRINT DSPLY	PRNT
045	CLR	CLEAR	CLR
046	PAUS	PAUSE	PAUS
047	CLFG	CLEAR FLAG	CLFG
050	(	(	(
051	)	)	)
052	*	×	×
053	+	+	+
054	RSET	RESET	RSET
055	-	−	-
056	.	•	•
057	/	÷	÷
060	0	0	ø
061	1	1	1
062	2	2	2
063	3	3	3
064	4	4	4
065	5	5	5
066	6	6	6
067	7	7	7
070	8	8	8
071	9	9	9
072	ADR	ADDRS	ADR
073	STRT	START	STRT
074	IF<0	<ø	IF<ø
075	=	=	=
076	IF>=	≥ø	IF≥ø
077	IF=0	=ø	IF=ø

OCTAL CODE	PRINT OUT	KEY	SYMBOL
100	K_	K ■	K
101	DG/R	DEG RAD	D/R
102	ARC	arc	arc
103	HYP	$hyper$	hyp
104	TAN	$\tan x$	tan
105	COS	$\cos x$	cos
106	SIN	$\sin x$	sin
107	X!	$x!$	x!
110	PI4	Π_4	Π_4
111	DLT3	Δ_3	Δ_3
112	3SM2	${}_3\Sigma_2$	${}_3\Sigma_2$
113	SUM1	Σ_1	Σ_1
114	SUM0	$\Sigma_ø$	Σ_0
115	LN	$\ln x$	ln
116	LOG	$\log x$	log
117	INT	$int\, x$	int
120	RSSQ	$\sqrt{\Sigma x^2}$	$\sqrt{\Sigma x^2}$
121	X↑A	$\lvert x\rvert^a$	x^a
122	RM__	REMOTE ■■	RMT
123	E↑X	e^x	e^x
124	X↑2	x^2	x^2
125	SQRT	$\sqrt{x}$	$\sqrt{x}$
126	1/X	$\frac{1}{x}$	1/x
127	PI	π	π
130	PAPR	PAPER FEED	PF
131	*10↑	$\times 10^{oo}$	10^{oo}
132	CONT	CONT	CONT
133	R AD	RETURN ADDRESS	RADR
134	+/-	+/-	+/-
135	GODP	GO TO DISPLAY	GODP
136	IFFG	FLAG	IFFG
137	ENDR	END	ENDR

OCTAL CODE	PRINT OUT	KEY	SYMBOL
201		LABEL ■	LBL
202		FROM TAPE ■	FTP
203		TO TAPE ■	TTP
204		EXECUTE ■	EXC
205		CLEAR R FILE ■	CFI
207		BELL	BELL
210		SPACE ←	SPC←
211		TAB	TAB
220		← STEP	STP←
221		INSERT	NSRT
222		DELETE	DLT
223		→ STEP	STP→
224		LIST	LIST
225		DISPLAY PROGRAM	DPRG
226		LEARN	LRN
240		SPACE	SPC
241	!	!	!
242	"	"	"
243	#	#	#
244	$	$	$
245	%	%	%
246	&	&	&
247	'	'	'
250	(	(	(
251	)	)	)
252	*	*	*
253	+	+	+
254	,	,	,
255	-	-	-
256	.	.	.
257	/	/	/
260	0	0	Ø
261	1	1	1
262	2	2	2
263	3	3	3
264	4	4	4
265	5	5	5
266	6	6	6
267	7	7	7
270	8	8	8
271	9	9	9
272	:	:	:
273	;	;	;
274	<	<	<
275	=	=	=
276	>	>	>
277	?	?	?



OCTAL CODE	PRINT OUT	KEY	SYMBOL
300	@	@	@
301	A	A	A
302	B	B	B
303	C	C	C
304	D	D	D
305	E	E	E
306	F	F	F
307	G	G	G
310	H	H	H
311	I	I	I
312	J	J	J
313	K	K	K
314	L	L	L
315	M	M	M
316	N	N	N
317	O	O	O
320	P	P	P
321	Q	Q	Q
322	R	R	R
323	S	S	S
324	T	T	T
325	U	U	U
326	V	V	V
327	W	W	W
330	X	X	X
331	Y	Y	Y
332	Z	Z	Z
333	[	[	[
334	\	\	\
335	]	]	]
336	↑	↑	↑
337	_	_	_

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

$K_{\emptyset}$ - K_2 : counters
K_3 : working
K_4 : determinant
K_5 - K_7 : working
K_8 : n
K_9 : n + 1
$R_{\emptyset\emptyset\emptyset}$: Ind. address
$R_{\emptyset 1}$: Ind. address
$R_{\emptyset 2}$: Ind. address
$R_{\emptyset 8}$ through $R_{n(n+1)+7}$: matrix $[a_{ij}]$ and [c]

Subroutine labels:

(A) through (Z)
(@) , (%) , (&)
($) , (#) , (STRT)
CD, PRNT, D/R, ENDR

PROGRAM DESCRIPTION

This program calculates the inverse of the real matrix $[A] = [a_{ij}]$ of order $n \leq 7$ on the basic Model 31 with 64 R-registers and 512 program steps, and up to $n \leq 31$ with various expanded memory options. Provision is included in the program for the solution of n simultaneous equations in n unknowns, once the inverse matrix $[A]^{-1} \equiv [B]$ is obtained.

The program uses a Gauss-Jordan normalization and elimination technique with row pivot interchange to improve accuracy. Storage in place is used to reduce the memory requirements. As each row of the matrix [A] is entered, the calculator appends and prints the row number.

The program prints out the determinant of matrix [A] as D[A], and the inverse matrix $[A]^{-1}$ as [B]. Then for the solution to simultaneous equations, where in matrix notation

$$[A] \times [x] = [c] ,$$

the constant vector [c] is entered, and the program prints out the solution vector [x] from

$$[x] = [B] \times [c]$$

This portion of the program may be repeated for as many constant vectors [c] as desired.

The program includes a checking option whereby the inverse [B] may be re-inversed to obtain [A]. The discrepancy between the original [A] and the re-inversed [A] is an indication of the accumulation of round-off error in the program, and of the ill-conditioning of the matrix [A]. The check procedure is recommended whenever the absolute value of the determinant D[A] is small.

The program requires two blocks on the tape cartridge; any two blocks may be used. The selected block numbers "M" and "N" must be entered in the final program steps as indicated on the program step lists. As the program requires two tape searches, "M" to "N", and "N" to "M", total search time is independent of "M" and "N".

The maximum matrix order n depends on the number of installed registers as follows:

Number of R-registers	n max	Number of R-registers	n max
64	7	448	20
128	10	640	24
192	13	1000	31
256	15		

If an order n greater than allowed by the installed memory is entered, an E2 "no such register" error message will be displayed. (With 1000 registers installed, no error message will be displayed regardless of n, but n is limited to $n \leq 31$.)

Note that the basic inversing routine execution time is roughly proportional to n^3, exclusive of tape search and printout; execution time for a 6th order matrix is about one minute plus search and print time. Inversion of a 15th order identity matrix requires approximately 11 minutes.

INPUT DATA CORRECTION PROCEDURE:

Following entry of [A] or [C] the calculator stops to allow checking and correction of any data accidentally entered incorrectly (watch sign errors). Incorrect data must be corrected by manually entering correct values to the proper registers using the Rxxx key and 3 digit register address. The register number may be determined by counting from the register number for the first entry in any given row, which is given by

$$Rxxx^{\#} = (r - 1)(n + 1) + 8$$

where r is the row number and n is the order of the matrix. Thus the first entry

a_{61} in the 6th row of the 12th order matrix is stored in register 8 + (6 -1)(12 - 1) = Ø73; a_{62} is in Rxxx Ø74; a_{63} in Rxxx Ø75, and so on.

The registers for constant vector entries can be determined from the formula

$$Rxxx^{\#} = r(n + 1) + 7$$

After manually entering data corrections, be sure to press Rxxx ØØØ to return the calculator to FILE Ø so that the program indirect address registers will operate correctly.

References:

1. S. S. Kuo, *Numerical Methods and Computers*, Addison-Wesley Publishing Co., Reading, Mass., 1965.

2. T. R. McCalla, *Introduction to Numerical Methods and FORTRAN Programming*, John Wiley and Sons, New York, 1967.

3. A Ralston and H. S. Wilf, *Mathematical Methods for Digital Computers, Vol. I*, John Wiley and Sons, New York, 1960.

EXAMPLES

Given the simultaneous equations:

$$x_1 + x_2/2 + x_3/3 = c_1$$
$$x_1/2 + x_2/3 + x_3/4 = c_2$$
$$x_1/3 + x_2/4 + x_3/5 = c_3$$

or in matrix notation

$$\begin{bmatrix} 1 & 1/2 & 1/3 \\ 1/2 & 1/3 & 1/4 \\ 1/3 & 1/4 & 1/5 \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

or $[A] \times [x] = [c]$

Solve for [B], the inverse of [A].

Then, given $c_1 = 1$, $c_2 = 1$, $c_3 = 1$
solve for $x_1 = 3$, $x_2 = -24$, $x_3 = 30$.

Given $c_1 = c_2 = c_3 = -1$, then
$x_1 = -3$, $x_2 = 24$, $x_3 = -30$.

Matrix [A] is a 3rd order Hilbert matrix. Hilbert matrices are often used to test computational algorithms because they are very ill-conditioned, that is, a small change in element values leads to a large change in the inverse [B]; as the matrix order increases the determinant rapidly diminishes toward 0.

```
MTRX [A] ORD N=?
             3.

[A]=?

             1.
             .5
 .3333333333
          1.00

             .5
 .3333333333
            .25
          2.00

 .3333333333
            .25
             .2
          3.00

D[A]=
 4.629629629E-04

[B]=

 9.000000001
-36.00000001
          30.
        1.00

-36.00000001
         192.
-        180.
        2.00

 30.00000001
-        180.
         180.
         3.00

[C]=?
             1.
             1.
             1.

[X]=
 3.000000001
-         24.
          30.

*

[C]=?
-            1.
-            1.
-            1.

[X]=
-3.000000001
          24.
-         30.

*
```

To re-inverse, EXC D/R

```
D[B]=
          2160.

[A]=

                1.
 .5000000001
 .3333333334
                1.00

 .5000000001
 .3333333334
 .2500000001
                2.00

 .3333333334
 .2500000001
 .2000000001
                3.00

END
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1	Load initial block	RSET FTP "M"		
2		START		Title
3	matrix order n	CONT	0	n
4	matrix [A] by rows			
	a_{11}	CONT	0	[A]
	.	.		
	a_{1n}	CONT	0	D[A]
	a_{21}	CONT	0	
	.	.		[B]
	.	.		
	a_{nn}	CONT	0	c_1
5	The calculator stops to allow correction of input data; see Program Description. When data is correct	CONT	0 during tape searches π	. . c_n
6	To solve simultaneous equations:	CONT	0	x_1 . .
7	constant vector [c]			x_n
	c_1	CONT	0	
	.		.	
	.		.	
	c_n	CONT	0	
8	Same as 5. If correct	CONT	π	
9	For a new constant vector [c]' return to Step 6.			
10	To inverse [B] to obtain [A] as an optional check	EXC D/R		D[B] [A] by re-inversing
	To terminate the program and reset the calculator for entry of a new matrix [A]'	EXC ENDR		

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
0	00 IFFG	EXC	(U)	CONT	EXC	STRT	LBL	(N)	Rxx	Rxx	LBL (N)
	10 Ø	Ø	=	K	6	Rxx	Rxx	Ø	1	=	
	20 Rxx	Rxx	Ø	Ø	K	6	=	Rxx	Rxx	Ø	
	30 1	RADR	GODP	LBL	(@)	Rxx	Ø	1	+	K	LBL (@)
	40 9	=	Rxx	Ø	1	RADR	GODP	LBL	(%)	K	LBL (%)
	50 8	+	8	=	Rxx	Ø	1	=	K	7	
	60 RADR	GODP	LBL	CD	CD	=	K	Ø	=	K	LBL CD
	70 1	8	=	Rxx	Ø	Ø	RADR	GODP	LBL	(P)	LBL (P)
	80 Rxx	Ø	Ø	+	1	=	Rxx	Ø	Ø	RADR	
	90 GODP	LBL	(W)	Rxx	Rxx	Ø	Ø	PRNT	RADR	=	LBL (W)
1	00 K	3	EXC	(P)	CD	1	Σ_1	K	1	-	
	10 K	8	)	IF=Ø	=	K	1	1	Σ_0	K	
	20 Ø	=	Rxx	Rxx	Ø	Ø	Ø	Ø	PRNT	PF	
	30 EXC	(P)	CONT	K	8	×	K	9	+	7	
	40 -	Rxx	Ø	Ø	=	K	2	K	3	GODP	
	50 LBL	(U)	CD	1	=	K	Ø	K	8	+	LBL (U)
	60 8	=	Rxx	Ø	2	LBL	(K)	K	Ø	-	LBL (K)
	70 K	8	)	IF=Ø	EXC	PRNT	CONT	K	Ø	-	
	80 Rxx	Rxx	Ø	2	)	IF=Ø	1	Σ_0	Rxx	Ø	
	90 2	+	K	9	=	Rxx	Ø	2	EXC	(K)	
2	00 CONT	K	Ø	+	7	=	Rxx	Ø	Ø	=	
	10 K	1	Rxx	Rxx	Ø	2	+	7	=	Rxx	
	20 Ø	1	LBL	(L)	EXC	(N)	Rxx	Ø	Ø	-	LBL (L)
	30 K	1	-	K	8	x^2	+	1	)	IF<Ø	
	40 Rxx	Ø	Ø	+	K	9	=	Rxx	Ø	Ø	

PROGRAM STEPS

MATRIX INVERSION BLOCK "M"

		0	1	2	3	4	5	6	7	8	9	COMMENTS
	50	EXC	@	EXC	L	CONT	K	Ø	×	K	9	
	60	+	7	=	Rxx	Ø	Ø	Rxx	Rxx	Ø	Ø	
	70	×	K	9	+	7	=	Rxx	Ø	1	EXC	
	80	N	EXC	K	LBL	PRNT	EXC	CD	LBL	Y	EXC	LBL PRNT, Y
	90	W	K	2	IF≥Ø	EXC	Y	CONT	LBL	V	π	LBL V
3	00	STOP	PF	[	C	]	=	?	EXC	%	LBL	LBL S
	10	S	CD	STOP	=	Rxx	Rxx	Ø	1	PRNT	Rxx	
	20	Ø	1	-	K	7	-	K	8	x^2	+	
	30	1	)	IF<Ø	EXC	@	EXC	S	CONT	PF	STOP	
	40	[	X	]	=	CD	1	=	K	2	=	
	50	K	3	LBL	T	CD	=	K	Ø	1	=	LBL T
	60	K	1	K	2	-	1	)	×	K	9	
	70	+	8	=	Rxx	Ø	Ø	EXC	%	LBL	X	LBL X
	80	Rxx	Rxx	Ø	Ø	×	Rxx	Rxx	Ø	1	)	
	90	Σ_0	K	1	-	K	8	)	IF<Ø	CD	1	
4	00	Σ_1	EXC	P	EXC	@	EXC	X	CONT	K	Ø	
	10	PRNT	K	2	-	K	8	)	IF<Ø	$_3\Sigma_2$	EXC	
	20	T	CONT	PF	*	EXC	V	LBL	STRT	PF	PF	LBL STRT
	30	M	T	R	X	SPC	[	A	]	SPC	O	
	40	R	D	SPC	N	=	?	CLR	STOP	=	K	
	50	8	PRNT	PF	+	1	=	K	9	x^2	-	
	60	5	=	Rxxx	Ø	Ø	Ø	Rxx	Rxx	Ø	Ø	
	70	[	A	]	=	?	EXC	CD	PF	LBL	A	LBL A
	80	CD	STOP	=	EXC	W	K	2	IF≥Ø	EXC	A	
	90	CONT	CD	STOP	CLFG	IFFG	LBL	D/R	SFG	CONT	CD	LBL D/R
		0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

MATRIX INVERSION BLOCK "M"

		0	1	2	3	4	5	6	7	8	9	COMMENTS
5	00	ADR	GODP	FTP	"N"	LBL	ENDR	E	N	D	CD	LBL ENDR
	10	PF	RSET									
	20											
	30											
	40											
	50											
	60											
	70											
	80											
	90											
	00											
	10											
	20											
	30											
	40											
	50											
	60											
	70											
	80											
	90											
	00											
	10											
	20											
	30											
	40											
		0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

MATRIX INVERSION BLOCK "N"

		0	1	2	3	4	5	6	7	8	9	COMMENTS
0	00	IFFG	CD	1	=	K	Ø	K	8	+	8	
	10	=	Rxx	Ø	Ø	LBL	Z	K	Ø	=	Rxx	LBL Z
	20	Rxx	Ø	Ø	-	K	8	)	IF<Ø	CD	1	
	30	Σ_0	EXC	&	EXC	Z	CONT	EXC	M	LBL	Q	LBL Q
	40	Rxx	Rxx	Ø	Ø	=	K	6	RADR	GODP	LBL	LBL O
	50	O	Rxx	Ø	1	+	1	=	Rxx	Ø	1	
	60	LBL	P	Rxx	Ø	Ø	+	1	=	Rxx	Ø	LBL P
	70	Ø	RADR	GODP	LBL	F	CD	1	Σ_1	K	1	LBL F
	80	-	K	8	)	IF≥Ø	EXC	D	CONT	LBL	&	LBL &
	90	Rxx	Ø	Ø	+	K	9	=	Rxx	Ø	Ø	
1	00	RADR	GODP	LBL	R	=	Rxx	Rxx	Ø	Ø	RADR	LBL R
	10	GODP	LBL	$	K	1	-	K	Ø	)	IF=Ø	LBL $
	20	1	Σ_1	CONT	RADR	GODP	LBL	M	CD	1	=	LBL M
	30	K	Ø	=	K	4	LBL	B	K	Ø	=	LBL B
	40	K	1	=	K	5	K	9	+	1	)	
	50	×	K	Ø	-	K	8	+	6	=	Rxx	
	60	Ø	Ø	=	K	7	EXC	Q	K	8	-	
	70	K	Ø	)	IF=Ø	EXC	G	CONT	K	7	+	
	80	K	9	=	Rxx	Ø	Ø	LBL	C	K	6	LBL C
	90	x^a	1	-	Rxx	Rxx	Ø	Ø	x^a	1	)	
2	00	IF≥Ø	EXC	F	EXC	C	CONT	K	1	+	1	
	10	=	K	5	EXC	Q	EXC	F	EXC	C	LBL	LBL D
	20	D	K	5	-	1	)	×	K	9	+	
	30	8	=	Rxx	Ø	1	K	7	-	K	Ø	
	40	+	1	=	Rxx	Ø	Ø	-	Rxx	Ø	1	
		0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

MATRIX INVERSION BLOCK "N"

	0	1	2	3	4	5	6	7	8	9	COMMENTS
50	)	IF<Ø	CD	1	+/-	=	K	1	Π_4	LBL	LBL (E)
60	(E)	EXC	(Q)	Rxx	Rxx	Ø	1	EXC	(R)	K	
70	6	=	Rxx	Rxx	Ø	1	CD	1	Σ_1	K	
80	1	-	K	8	)	IF<Ø	EXC	(O)	EXC	(E)	
90	CONT	LBL	(G)	K	7	=	Rxx	Ø	Ø	EXC	LBL (G)
3 00	(Q)	K	6	Π_4	CD	1	EXC	(R)	K	7	
10	-	K	Ø	+	1	=	Rxx	Ø	Ø	=	
20	K	3	+	K	8	-	1	=	K	5	
30	LBL	(H)	Rxx	Rxx	Ø	Ø	÷	K	6	EXC	LBL (H)
40	(R)	Rxx	Ø	Ø	-	K	5	)	IF<Ø	EXC	
50	(P)	EXC	(H)	CONT	CD	1	=	K	1	LBL	LBL (I)
60	(I)	K	3	=	Rxx	Ø	1	EXC	($)	K	
70	7	+	K	9	×	(	K	1	-	K	
80	Ø	=	Rxx	Ø	Ø	EXC	(Q)	CD	EXC	(R)	
90	Rxx	Ø	Ø	-	K	Ø	+	1	=	Rxx	
4 00	Ø	Ø	LBL	(J)	Rxx	Rxx	Ø	Ø	-	K	LBL (J)
10	6	×	Rxx	Rxx	Ø	1	EXC	(R)	Rxx	Ø	
20	1	-	K	5	)	IF<Ø	EXC	(O)	EXC	(J)	
30	CONT	CD	1	Σ_1	EXC	($)	K	8	-	K	
40	1	)	IF≥Ø	EXC	(I)	CONT	CD	1	Σ_0	K	
50	8	-	K	Ø	)	IF≥Ø	EXC	(B)	CONT	IFFG	
60	PF	(D)	([)	(B)	(])	(=)	K	4	PRNT	PF	
70	([)	(A)	(])	(=)	EXC	(#)	CONT	(D)	([)	(A)	
80	(])	(=)	K	4	PRNT	PF	([)	(B)	(])	(=)	
90	LBL	(#)	PF	SFG	CD	ADR	GODP	FTP	"M"		LBL (#)
	0	1	2	3	4	5	6	7	8	9	

Matrix Inversion and Simultaneous Equations for 1024 Step Calculator

TITLE

MINIMUM HARDWARE REQUIRED

TEK 31 (1024 Steps and 128 Registers)

REGISTERS USED

$K_{\emptyset}$ - K_2 : counters
K_3 : working
K_4 : determinant
K_5 - K_7 : working
K_8 : n
K_9 : n + 1
$R_{\emptyset\emptyset\emptyset}$: ind. adr. reg.
$R_{\emptyset 1}$: ind. adr. reg.
$R_{\emptyset 2}$: ind. adr. reg.
$R_{\emptyset 8}$ through
$R_{n(n+1)+7}$:
matrix $[a_{ij}]$ and [c]

Subroutine labels:
(A) through (Z)
(@) , (%) , (&)
($) , STRT, PRNT
D/R, ENDR

PROGRAM DESCRIPTION

This program calculates the inverse of the real matrix $[A] = [a_{ij}]$ of order $n \leq 7$ on the basic Model 31 with 64 R-registers and 1024 program steps, and up to $n \leq 31$ with various expanded memory options. Provision is included in the program for the solution of n simultaneous equations in n unknowns, once the inverse matrix $[A]^{-1} \equiv [B]$ is obtained.

The program uses a Gauss-Jordan normalization and elimination technique with row pivot interchange to improve accuracy. Storage in place is used to reduce memory requirements. As each row of the matrix [A] is entered, the calculator appends and prints the row number.

The program prints out the determinant of matrix [A] as D[A], and the inverse matrix $[A]^{-1}$ as [B]. Then for the solution to simultaneous equations, where in matrix notation

$$[A] \times [x] = [c] ,$$

the constant vector [c] is entered, and the program prints out the solution vector [x] from

$$[x] = [A]^{-1} \times [c] \equiv [B] \times [c] .$$

The portion of the program may be repeated for as many constant vectors [c] as desired.

The program includes a checking option whereby the inverse matrix [B] may be re-inversed to obtain [A]. The discrepancy between the original [A] and the re-inversed [A] is an indication of the accumulation of round-off error in the program, and of the ill-conditioning of the matrix [A]. The check procedure is recommended whenever the absolute value of the determinant D[A] is small. Execution of the checking routine terminates the program and resets the calculator.

The program requires 964 steps. It calls for data entry on the printer, and once data is entered, its operation is entirely automatic. The maximum matrix order n depends on the number of installed registers as follows:

Number of R-registers	n max	Number of R-registers	n max
64	7	448	20
128	10	640	24
192	13	1000	31
256	15		

If an order n greater than allowed by the installed memory is entered, an E2 "no such register" error message will be displayed. (With 1000 registers installed, no error message will be displayed regardless of n, but n is limited to $n \leq 31$.)

Note that the basic inversing routine execution time is roughly proportional to n^3, exclusive of data entry and printout; execution time for a 6th order matrix is about one minute. Inversion of a 15th order matrix requires 11 to 12 minutes.

INPUT DATA CORRECTION PROCEDURE:

Following entry of [A] or [C] the calculator stops to allow checking and correction of any data accidentally entered incorrectly (watch sign errors). Incorrect data must be corrected by manually entering correct values to the proper registers using the Rxxx key and 3 digit register address. The register number may be determined by counting from the register number for the first entry in any given row, which is given by

$$Rxxx^{\#} = (r - 1)(n + 1) + 8$$

where r is the row number and n is the order of the matrix. Thus the first entry a_{61} in the 6th row of the 12th order matrix is stored in register $8 + (6 - 1)(12 + 1) = \emptyset 73$; a_{62} is in Rxxx Ø74; a_{63} in Rxxx Ø75, and so on.

The registers for constant vector entries can be determined from the formula

$$Rxxx^{\#} = r(n + 1) + 7$$

After manually entering data corrections, be sure to press Rxxx ØØØ to return the calculator to FILE Ø so that the program indirect address registers will operate correctly.

References:

1. S. S. Kuo, *Numerical Methods and Computers*, Addison-Wesley Publishing Co., Reading, Mass., 1965.

2. T. R. McCalla, *Introduction to Numerical Methods and FORTRAN Programming*, John Wiley and Sons, New York, 1967.

3. A. Ralston and H. S. Wilf, *Mathematical Methods for Digital Computers, Vol. I*, John Wiley and Sons, New York, 1960.

EXAMPLES

Given the simultaneous equations:

$$x_1 + x_2/2 + x_3/3 = c_1$$
$$x_1/2 + x_2/3 + x_3/4 = c_2$$
$$x_1/3 + x_2/4 + x_3/5 = c_3$$

or in matrix notation

$$\begin{bmatrix} 1 & 1/2 & 1/3 \\ 1/2 & 1/3 & 1/4 \\ 1/3 & 1/4 & 1/5 \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

or $[A] \times [x] = [c]$

Solve for [B], the inverse of [A].

Then, given $c_1 = 1$, $c_2 = 1$, $c_3 = 1$
solve for $x_1 = 3$, $x_2 = -24$, $x_3 = 30$.

Given $c_1 = c_2 = c_3 = -1$, then
$x_1 = -3$, $x_2 = 24$, $x_3 = -30$.

Matrix [A] is a 3rd order Hilbert matrix. Hilbert matrices are often used to test computational algorithms because they are very ill-conditioned, that is, a small change in element values leads to a large change in the inverse [B]; as the matrix order increases the determinant rapidly diminishes toward 0.

```
MTRX [A] ORD N?
             3.

[A]=?

             1.
             .5
 .3333333333
          1.00

             .5
 .3333333333
            .25
          2.00

 .3333333333
            .25
             .2
          3.00

D[A]=
 4.629629629E-04

[B]=

 9.000000001
-36.00000001
          30.
          1.00

-36.00000001
         192.
-        180.
         2.00

 30.00000001
-        180.
         180.
         3.00

[C]=?
             1.
             1.
             1.

[X]=
 3.000000001
-         24.
          30.

*

[C]=?
-            1.
-            1.
-            1.

[X]=
-3.000000001
          24.
-         30.

*
```

To re-inverse, EXC D/R

```
D[B]=
          2160.

[A]=

             1.
 .5000000001
 .3333333334
          1.00

 .5000000001
 .3333333334
 .2500000001
          2.00

 .3333333334
 .2500000001
 .2000000001
          3.00

END
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1		START		Title
2	matrix order n	CONT	0	n
3	matrix [A] by rows:			[A] by rows
	a_{11}	CONT	0	with row
	.	.	.	numbers
	a_{1n}	CONT	0	appended as
	a_{21}	CONT	0	1.00, 2.00,
	.	.	.	..., n.00
	.	.	.	
	a_{nn}	CONT	0	D[A]
4	The calculator stops to allow correction of input data; see Program Description.			[B] with row numbers
	When data is correct	CONT	π	
				[c]
5	To solve simultaneous equations	CONT	0	[x]
6	constant vector [c]			
	c_1	CONT	0	
	.	.	.	
	c_n	CONT	0	
7	Same as 4. If correct	CONT	π	
8	For a new constant vector [c]' return to Step 5.			
9	To inverse [B] to obtain check	[A] as an optional EXC D/R		D[B]
				[A]
	To terminate the program and reset the calculator for entry of a new matrix [A]'	EXC ENDR		END

		0	1	2	3	4	5	6	7	8	9	COMMENTS
0	00	EXC	STRT	LBL	(N)	Rxx	Rxx	Ø	Ø	=	K	LBL (N)
	10	6	Rxx	Rxx	Ø	1	=	Rxx	Rxx	Ø	Ø	
	20	K	6	=	Rxx	Rxx	Ø	1	RADR	GODP	LBL	LBL (O)
	30	(O)	Rxx	Ø	1	+	1	=	Rxx	Ø	1	
	40	LBL	(P)	Rxx	Ø	Ø	+	1	=	Rxx	Ø	LBL (P)
	50	Ø	RADR	GODP	LBL	(Q)	Rxx	Rxx	Ø	Ø	=	LBL (Q)
	60	K	6	RADR	GODP	LBL	(F)	CD	1	Σ_1	K	LBL (F)
	70	1	−	K	8	)	IF≥Ø	EXC	(D)	CONT	LBL	LBL (&)
	80	(&)	Rxx	Ø	Ø	+	K	9	=	Rxx	Ø	
	90	Ø	RADR	GODP	LBL	(@)	Rxx	Ø	1	+	K	LBL (@)
1	00	9	=	Rxx	Ø	1	RADR	GODP	LBL	(%)	K	LBL (%)
	10	8	+	8	=	Rxx	Ø	1	=	K	7	
	20	RADR	GODP	LBL	($)	K	1	−	K	Ø	)	LBL ($)
	30	IF=Ø	1	Σ_1	CONT	RADR	GODP	LBL	(R)	CD	=	LBL (R)
	40	K	Ø	=	K	1	8	=	Rxx	Ø	Ø	
	50	RADR	GODP	LBL	(W)	Rxx	Rxx	Ø	Ø	PRNT	RADR	LBL (W)
	60	=	K	3	EXC	(P)	CD	1	Σ_1	K	1	
	70	−	K	8	)	IF=Ø	=	K	1	1	Σ_0	
	80	K	Ø	=	Rxx	Rxx	Ø	Ø	Ø	Ø	PRNT	
	90	PF	EXC	(P)	CONT	K	8	×	K	9	+	
2	00	7	−	Rxx	Ø	Ø	=	K	2	K	3	
	10	GODP	LBL	STRT	PF	PF	(M)	(T)	(R)	(X)	(SPC)	LBL STRT
	20	([)	(A)	(])	(SPC)	(O)	(R)	(D)	(SPC)	(N)	(?)	
	30	CLR	STOP	=	K	8	PRNT	PF	+	1	=	
	40	K	9	x^2	−	5	=	Rxxx	Ø	Ø	Ø	
		0	1	2	3	4	5	6	7	8	9	

	0	1	2	3	4	5	6	7	8	9	COMMENTS
50	Rxx	Rxx	Ø	Ø	[	A	]	=	?	EXC	
60	R	PF	LBL	A	CD	STOP	=	EXC	W	K	LBL A
70	2	IF≥Ø	EXC	A	CONT	CD	STOP	LBL	M	CD	LBL M
80	1	=	K	Ø	=	K	4	LBL	B	K	LBL B
90	Ø	=	K	1	=	K	5	K	9	+	
3 00	1	)	×	K	Ø	−	K	8	+	6	
10	=	Rxx	Ø	Ø	=	K	7	EXC	Q	K	
20	8	−	K	Ø	)	IF=Ø	EXC	G	CONT	K	
30	7	+	K	9	=	Rxx	Ø	Ø	LBL	C	LBL C
40	K	6	x^a	1	−	Rxx	Rxx	Ø	Ø	x^a	
50	1	)	IF≥Ø	EXC	F	EXC	C	CONT	K	1	
60	+	1	=	K	5	EXC	Q	EXC	F	EXC	
70	C	LBL	D	K	5	−	1	)	×	K	LBL D
80	9	+	8	=	Rxx	Ø	1	K	7	−	
90	K	Ø	+	1	=	Rxx	Ø	Ø	−	Rxx	
4 00	Ø	1	)	IF<Ø	CD	1	+/−	=	K	1	
10	Π_4	LBL	E	EXC	N	CD	1	Σ_1	K	1	LBL E
20	−	K	8	)	IF<Ø	EXC	O	EXC	E	CONT	
30	LBL	G	K	7	=	Rxx	Ø	Ø	EXC	Q	LBL G
40	K	6	Π_4	CD	1	=	Rxx	Rxx	Ø	Ø	
50	K	7	−	K	Ø	+	1	=	Rxx	Ø	
60	Ø	=	K	3	+	K	8	−	1	=	
70	K	5	LBL	H	Rxx	Rxx	Ø	Ø	÷	K	LBL H
80	6	=	Rxx	Rxx	Ø	Ø	Rxx	Ø	Ø	−	
90	K	5	)	IF<Ø	EXC	P	EXC	H	CONT	CD	
	0	1	2	3	4	5	6	7	8	9	

		0	1	2	3	4	5	6	7	8	9	COMMENTS
5	00	1	=	K	1	LBL	(I)	K	3	=	Rxx	LBL (I)
	10	Ø	1	EXC	($)	K	7	+	K	9	×	
	20	(	K	1	−	K	Ø	=	Rxx	Ø	Ø	
	30	EXC	(Q)	CD	=	Rxx	Rxx	Ø	Ø	Rxx	Ø	
	40	Ø	−	K	Ø	+	1	=	Rxx	Ø	Ø	
	50	LBL	(J)	Rxx	Rxx	Ø	Ø	−	K	6	×	LBL (J)
	60	Rxx	Rxx	Ø	1	=	Rxx	Rxx	Ø	Ø	Rxx	
	70	Ø	1	−	K	5	)	IF<Ø	EXC	(O)	EXC	
	80	(J)	CONT	CD	1	Σ_1	EXC	($)	K	8	−	
	90	K	1	)	IF≥Ø	EXC	(I)	CONT	CD	1	Σ_0	
6	00	K	8	−	K	Ø	)	IF≥Ø	EXC	(B)	CONT	
	10	IFFG	PF	(D)	([)	(B)	(])	(=)	K	4	PRNT	
	20	PF	([)	(A)	(])	(=)	EXC	(U)	CONT	(D)	([)	
	30	(A)	(])	(=)	K	4	PRNT	PF	([)	(B)	(])	
	40	(=)	LBL	(U)	PF	CD	1	=	K	Ø	K	LBL (U)
	50	8	+	8	=	Rxx	Ø	2	LBL	(K)	K	LBL (K)
	60	Ø	−	K	8	)	IF=Ø	EXC	PRNT	CONT	K	
	70	Ø	−	Rxx	Rxx	Ø	2	)	IF=Ø	1	Σ_0	
	80	Rxx	Ø	2	+	K	9	=	Rxx	Ø	2	
	90	EXC	(K)	CONT	K	Ø	+	7	=	Rxx	Ø	
7	00	Ø	=	K	1	Rxx	Rxx	Ø	2	+	7	
	10	=	Rxx	Ø	1	LBL	(L)	EXC	(N)	Rxx	Ø	LBL (L)
	20	Ø	−	K	1	−	K	8	x^2	+	1	
	30	)	IF<Ø	EXC	(&)	EXC	(@)	EXC	(L)	CONT	K	
	40	Ø	×	K	9	+	7	=	Rxx	Ø	Ø	
		0	1	2	3	4	5	6	7	8	9	

	0	1	2	3	4	5	6	7	8	9	COMMENTS
50	Rxx	Rxx	Ø	Ø	×	K	9	+	7	=	
60	Rxx	Ø	1	EXC	(N)	EXC	(K)	LBL	PRNT	EXC	LBL PRNT
70	(R)	LBL	(Y)	EXC	(W)	K	2	IF≥Ø	EXC	(Y)	LBL (Y)
80	CONT	IFFG	EXC	ENDR	CONT	LBL	(V)	π	STOP	PF	LBL (V)
90	([)	(C)	(])	(=)	(?)	EXC	(%)	LBL	(S)	CD	LBL (S)
8 00	STOP	=	Rxx	Rxx	Ø	1	PRNT	Rxx	Ø	1	
10	−	K	7	−	K	8	x^2	+	1	)	
20	IF<Ø	EXC	(@)	EXC	(S)	CONT	PF	CD	STOP	([)	
30	(X)	(])	(=)	CD	1	=	K	2	=	K	
40	3	LBL	(T)	CD	=	K	Ø	1	=	K	LBL (T)
50	1	K	2	−	1	)	×	K	9	+	
60	8	=	Rxx	Ø	Ø	EXC	(%)	LBL	(X)	Rxx	LBL (X)
70	Rxx	Ø	Ø	×	Rxx	Rxx	Ø	1	)	Σ_0	
80	K	1	−	K	8	)	IF<Ø	CD	1	Σ_1	
90	EXC	(P)	EXC	(@)	EXC	(X)	CONT	K	Ø	PRNT	
9 00	K	2	−	K	8	)	IF<Ø	$_3\Sigma_2$	EXC	(T)	
10	CONT	PF	(*)	EXC	(V)	LBL	D/R	STFG	CD	1	LBL D/R
20	=	K	Ø	K	8	+	8	=	Rxx	Ø	
30	Ø	LBL	(Z)	K	Ø	=	Rxx	Rxx	Ø	Ø	LBL (Z)
40	−	K	8	)	IF<Ø	CD	1	Σ_0	EXC	(&)	
50	EXC	(Z)	CONT	EXC	(M)	LBL	ENDR	PF	(E)	(N)	LBL ENDR
60	(D)	CD	PF	PF	RSET						
70											
80											
90											
	0	1	2	3	4	5	6	7	8	9	

NOTES

MINIMUM HARDWARE REQUIRED

TEK 31, Printer (512 Steps, 64 Registers)

REGISTERS USED

$K_{\emptyset}$ - K_9 : working

$R_{\emptyset\emptyset\emptyset}$: ind. address
$R_{\emptyset 1}$: ind. address
$R_{\emptyset 2}$: ind. address

$R_{\emptyset 8}$ through R_{xxx}
for matrix storage

Subroutine labels used:
STRT, (<)
Ø through 9
(A) through (Z)

PROGRAM DESCRIPTION

This program consists of five BLOCKS for performing the following matrix arithmetic operations.

BLOCK Ø

LBL STRT: Title and instructions
LBL (A) : $[A_{k1}]$ initial data entry
LBL Ø: $[A_{k1}] \times [A_{k1}] = [R_{k1}]$ where k = 1

BLOCK 1

LBL 1: $[A_{k1}] \times [B_{mn}] = [R_{kn}]$ where 1 = m

BLOCK 2

LBL 2: $[B_{mn}] \times [A_{k1}] = [R_{m1}]$ where n = k

BLOCK 3

LBL 4: $[A_{k1}] + [B_{mn}] = [R_{k1}]$ where k = m, 1 = n
LBL 5: $[A_{k1}] - [B_{mn}] = [R_{k1}]$ where k = m, 1 = n
LBL 6: $[B_{mn}] - [A_{k1}] = [R_{k1}]$ where k = m, 1 = n
LBL 7: Transpose $[A_{k1}] = [A]' = [R_{1k}]$

BLOCK 4

LBL 3: $S \times [A_{k1}] = [R_{k1}]$, S a constant
LBL 8: Adds row numbers to $[A_{k1}]$ as required for matrix inversion program 4-1.
LBL 9: Removes row numbers from $[A_{k1}]$ following operation of program 4-1.

The result of each operation is stored as matrix $[A_{k1}]$ in place of the previous data. Thus the five BLOCKS may be used to perform almost any sequence of matrix arithmetic operations, so long as they are mathematically defined and the calculator storage capacity is not exceeded.

Each BLOCK includes calling routines for automatic loading of any of the other BLOCKs as a sequence of operations is executed. Labels (A) , 1, 2, 3, and 4 will run automatically when called from any other BLOCK. Labels STRT, 5, 6, 7, 8, and 9 require two EXECUTE's when called from another block: the first calls and loads the proper BLOCK, the second starts execution of the desired routine after loading is complete.

For matrix multiplication, storage requirements will be minimized if the smaller of the two matrices is called matrix $[A_{kl}]$ since all elements of [A] must be stored initially, while only one column or row of [B] must be stored during operation of [A] × [B] or [B] × [A], respectively. Storage requirements for any of the operations may be determined from the following equations, where R is the number of R-registers installed in your calculator:

[A] × [A], [A]': $2kl \leq R - 8$
[A] × [B]: $kl + m + kn \leq R - 8$
[B] × [A]: $kl + n + ml \leq R - 8$
S × [A], [A] ± [B], [B] - [A]: $kl \leq R - 8$
LBL 8, Add row numbers: $k(1 + 1) \leq R - 8$

Row and column indices k, 1 for [A] and m, n for [B] must be entered as indicated during operation of the program. If an inappropriate row/column index is entered, in most cases the program will print INDX ? and STOP to wait for your correction. Correction is easily accomplished by reexecuting the desired operation and reentering the correct matrix indices. Storage requirements are tested for each operation; if your calculator does not have sufficient storage for the matrix [A] and operation you have entered, the program will display the E2 "no such register" error message.

When LBL 8 is used to add row numbers for inversing by program 4-1, the program will print 4-1 loading instructions for any square matrix. For a non-square matrix, the program will add and print out the matrix with row numbers with the message "NO INVRS" to indicate that the non-square matrix cannot be inversed.

When LBL 9 is used to delete row numbers from a matrix, the program STOPs to ask "INDX OK?". At this point recall K_6 and K_7, and be sure that they are equal to the row (k) and column (1) indices, respectively, of the matrix. When deleting row numbers from a matrix inversed by program 4-1, it will be necessary to manually set K_6 and K_7 equal to the matrix order n (n = k = 1 for a square matrix; enter $n = K_6 = K_7$ manually).

EXAMPLES

1. To perform the following series of operations, load BLK Ø

2. EXC STRT

3. Enter k = 3, l = 4

$$A_{kl} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix}$$

4. Multiply [A] × [B]:
   EXC 1 and the program will auto load and execute BLK 1
   Enter m = 4, n = 3

$$B_{mn} = \begin{bmatrix} 12 & 8 & 4 \\ 11 & 7 & 3 \\ 10 & 6 & 2 \\ 9 & 5 & 1 \end{bmatrix}$$

5. Multiply by scalar S = .01
   EXC 3 and the program will auto load and execute
   Enter S = 0.01 Press CONT

6. Subtract [B]: EXC 5
   The program will load BLK 3.
   Again EXC 5 and enter m = 3, n = 3

$$B_{mn} = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 3 & 3 & 3 \end{bmatrix}$$

7. Multiply [B] × [A]:
   EXC 2 and the program will auto load and execute BLK 2.
   Enter m = 4, n = 3

$$B_{mn} = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ -1 & -2 & -3 \\ -3 & -2 & -1 \end{bmatrix}$$

```
MTRX ARITH

FOR,EXC:
A*A, 0
A*B, 1
B*A, 2
S*A, 3
A+B, 4
A-B, 5
B-A, 6
A',  7

ROW #'S:
ADD, 8
DLT, 9

**

A: K,L=?
                3.
                4.

A=?
BY ROWS
                1.
                2.
                3.
                4.
                5.
                6.
                7.
                8.
                9.
               10.
               11.
               12.
**

A*B:
B: M,N=?
                3.
                4.

#  INDX?

A*B:
B: M,N=?
                4.
                3.

B=?
BY COL'S
               12.
               11.
               10.
                9.
                8.
                7.
                6.
                5.
                4.
                3.
                2.
                1.
```

```
A*B=
BY ROWS

K,L=
                3.
                3.
              100.
               60.
               20.
              268.
              164.
               60.
              436.
              268.
              100.
**

S*A=
BY ROWS

S=?
              .01
K,L=
                3.
                3.
                1.
               .6
               .2
             2.68
             1.64
               .6
             4.36
             2.68
               1.
**

A-B:
B: M,N=?
                3.
                3.

B=?
BY ROWS
                1.
                1.
                1.
                2.
                2.
                2.
                3.
                3.
                3.
**
```

```
A-B=
BY ROWS

K,L=
                3.
                3.
               0.
             - .4
             - .8
              .68
             - .36
             - 1.4
             1.36
             - .32
             - 2.
**

B*A:
B: M,N=?
                4.
                3.

B=?
BY ROWS
                1.
                2.
                3.
                3.
                2.
                1.
             - 1.
             - 2.
             - 3.
             - 3.
             - 2.
             - 1.

B*A=
BY ROWS

K,L=
                4.
                3.
             5.44
             - 2.08
             - 9.6
             2.72
             - 2.24
             - 7.2
             - 5.44
             2.08
               9.6
             - 2.72
               2.24
               7.2
**
```

Note the reversal error in entry of m and n indicated by the INDX ? message. Correct by pressing EXC 1 again and reentering m = 4 and n = 3.

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1	Load Block Ø	RSET FTP Ø		
2	Print Title	EXC STRT		Title and EXC instructions
3	k	CONT		k
	1	CONT		1
	a_{11}	CONT		a_{11}
	⋮	⋮		⋮
	a_{k1}	CONT		a_{k1}
4	Select desired operation			as required
	A × A	EXC Ø		s
	A × B	EXC 1		m
	B × A	EXC 2		n
	S × A	EXC 3		b elements
	A + B	EXC 4		
	A - B	EXC 5		Results
	B - A	EXC 6		k
	A'	EXC 7		1
	Add row numbers	EXC 8		a_{11}
	Delete row numbers	EXC 9		⋮
				a_{k1}
5	Enter S, m, n, [B], etc., as requested by the PRINTOUT.			See examples for detailed printouts.
6	To perform another operation on the result of 4 and 5, return to Step 4.			
7	To enter a new matrix A without reprinting the title, etc., press EXC (A) . Note that LBLs (A) , 1, 2, 3, and 4 execute automatically from any BLOCK. LBLs Ø, 5, 6, 7, 8, 9 will require a second EXC following completion of tape loading when called from another BLOCK.			

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9
0	IFFG	EXC	(A)	CONT	STOP	LBL	(Y)	Rxx	Rxx	Ø
10	1	IFFG	CD	STOP	CONT	=	Rxx	Rxx	Ø	Ø
20	PRNT	Rxx	Ø	Ø	−	7	)	÷	K	7
30	=	K	Ø	−	K	Ø	int	)	IF=Ø	PF
40	CONT	Rxx	Ø	1	+	1	=	Rxx	Ø	1
50	Rxx	Ø	Ø	+	1	=	Rxx	Ø	Ø	−
60	K	5	−	8	)	IF<Ø	EXC	(Y)	CONT	(*)
70	(*)	CD	PF	RSET	LBL	Ø	(A)	(*)	(A)	(=)
80	EXC	(W)	(K)	(,)	(L)	(=)	K	6	PRNT	−
90	K	7	PRNT	PF	PF	)	x^2	+/−	IF<Ø	(I)
1 00	(N)	(D)	(X)	(?)	PF	PF	STOP	CONT	K	6
10	x^2	=	K	5	×	2	=	K	4	EXC
20	(I)	CD	=	K	Ø	EXC	(J)	LBL	(C)	CD
30	8	=	Rxx	Ø	2	+	K	5	=	K
40	1	LBL	(D)	Rxx	Rxx	Ø	Ø	×	Rxx	Rxx
50	Ø	2	)	Σ_0	Rxx	Ø	Ø	+	1	=
60	Rxx	Ø	Ø	Rxx	Ø	2	+	K	7	=
70	Rxx	Ø	2	−	K	1	)	IF<Ø	EXC	(D)
80	CONT	K	Ø	=	Rxx	Rxx	Ø	1	CD	=
90	K	Ø	1	Σ_1	Rxx	Ø	1	+	1	=
2 00	Rxx	Ø	1	Rxx	Ø	2	−	K	5	−
10	K	7	−	7	)	IF<Ø	Rxx	Ø	Ø	−
20	K	7	=	Rxx	Ø	Ø	Rxx	Ø	2	−
30	K	5	+	1	=	Rxx	Ø	2	EXC	(D)
40	CONT	Rxx	Ø	1	−	K	4	−	8	)
	0	1	2	3	4	5	6	7	8	9

COMMENTS

LBLs
(Y)
Ø
(C)
(D)

PROGRAM STEPS

		0	1	2	3	4	5	6	7	8	9	COMMENTS
												LBLs
	50	IF<Ø	EXC	C	CONT	EXC	J	CLFG	EXC	Y	LBL	
	60	W	B	Y	SPC	R	O	W	S	RADR	PF	W
	70	GODP	LBL	I	+	7	=	Rxx	Ø	Ø	Ø	I
	80	Rxx	Rxx	Ø	Ø	RADR	GODP	LBL	J	CD	8	J
	90	=	Rxx	Ø	Ø	+	K	5	=	Rxx	Ø	
3	00	1	RADR	GODP	LBL	STRT	M	T	R	X	SPC	STRT
	10	A	R	I	T	H	CLR	PF	F	O	R	
	20	,	E	X	C	:	PF	A	*	A	,	
	30	SPC	Ø	PF	A	*	B	,	SPC	1	PF	
	40	B	*	A	,	SPC	2	PF	S	*	A	
	50	,	SPC	3	PF	A	+	B	,	SPC	4	
	60	PF	A	-	B	,	SPC	5	PF	B	-	
	70	A	,	SPC	6	PF	A	'	,	SPC	SPC	
	80	7	PF	PF	R	O	W	SPC	#	'	S	
	90	:	PF	A	D	D	,	SPC	8	PF	D	
4	00	L	T	,	SPC	9	PF	PF	*	*	PF	
	10	PF	LBL	A	A	:	SPC	K	,	L	=	A
	20	?	CLR	STOP	=	K	6	PRNT	CD	STOP	=	
	30	K	7	PRNT	PF	×	K	6	=	K	5	
	40	IF=Ø	I	N	D	X	?	PF	PF	STOP	CONT	
	50	EXC	I	A	=	?	EXC	W	CD	=	Rxx	
	60	Ø	1	8	=	Rxx	Ø	Ø	SFG	EXC	Y	
	70	LBL	1	CD	ADR	GODP	FTP	1	LBL	2	CD	1, 2
	80	ADR	GODP	FTP	2	LBL	4	SFG	LBL	5	LBL	4, 5
	90	6	LBL	7	CD	ADR	GODP	FTP	3	LBL	3	6, 7, 3
			0	1	2	3	4	5	6	7	8	9

PROGRAM STEPS

BLOCK Ø

		0	1	2	3	4	5	6	7	8	9	LBLs	COMMENTS
5	00	SFG	LBL	8	LBL	9	CD	ADR	GODP	FTP	4	8, 9	
	10												
	20												
	30												
	40												
	50												
	60												
	70												
	80												
	90												
	00												
	10												
	20												
	30												
	40												
	50												
	60												
	70												
	80												
	90												
	00												
	10												
	20												
	30												
	40												
		0	1	2	3	4	5	6	7	8	9		

PROGRAM STEPS

BLOCK 1

		0	1	2	3	4	5	6	7	8	9	COMMENTS LBLs
0	00	EXC	1	LBL	(B)	K	4	+	7	=	LBL	(B)
	10	(E)	Rxx	Ø	1	+	1	=	Rxx	Ø	1	(E)
	20	CD	STOP	=	Rxx	Rxx	Ø	1	PRNT	Rxx	Ø	
	30	1	−	K	4	−	K	8	−	7	)	
	40	IF<Ø	EXC	(E)	CONT	CD	=	K	Ø	=	K	
	50	1	8	=	Rxx	Ø	Ø	+	K	4	−	
	60	1	=	LBL	(G)	Rxx	Ø	1	+	1	=	(G)
	70	Rxx	Ø	1	Rxx	Rxx	Ø	Ø	×	Rxx	Rxx	
	80	Ø	1	)	Σ_0	Rxx	Ø	Ø	+	1	=	
	90	Rxx	Ø	Ø	Rxx	Ø	1	−	K	4	−	
1	00	K	8	−	7	)	IF<Ø	EXC	(G)	CONT	K	
	10	Ø	=	Rxx	Rxx	Ø	2	CD	=	K	Ø	
	20	1	Σ_1	Rxx	Ø	2	+	K	9	=	Rxx	
	30	Ø	2	K	1	−	K	6	)	IF<Ø	Rxx	
	40	Ø	1	−	K	8	=	Rxx	Ø	1	EXC	
	50	(G)	CONT	Rxx	Ø	2	−	K	6	×	K	
	60	9	+	1	=	Rxx	Ø	2	−	K	5	
	70	−	K	9	−	8	)	IF<Ø	PF	EXC	(B)	
	80	CONT	PF	(A)	(*)	(B)	(=)	CD	(B)	(Y)	(SPC)	
	90	(R)	(O)	(W)	(S)	8	=	Rxx	Ø	Ø	+	
2	00	K	5	=	Rxx	Ø	1	PF	K	6	(K)	
	10	(,)	(L)	(=)	PRNT	K	9	=	K	7	PRNT	
	20	×	K	6	=	K .	5	PF	PF	LBL	(X)	(X)
	30	Rxx	Rxx	Ø	1	=	Rxx	Rxx	Ø	Ø	PRNT	
	40	Rxx	Ø	Ø	−	7	)	÷	K	7	=	
		0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

BLOCK 1

	0	1	2	3	4	5	6	7	8	9	COMMENTS
											LBLs
50	K	Ø	-	K	Ø	int	)	IF=Ø	PF	CONT	
60	Rxx	Ø	1	+	1	=	Rxx	Ø	1	Rxx	
70	Ø	Ø	+	1	=	Rxx	Ø	Ø	-	K	
80	5	-	8	)	IF<Ø	EXC	(X)	CONT	(*)	(*)	
90	CD	PF	RSET	LBL	1	K	6	×	K	7	1
3 00	=	K	5	(A)	(*)	(B)	(:)	CLR	(B)	(:)	
10	(SPC)	(M)	(,)	(N)	(=)	(?)	STOP	=	K	8	
20	PRNT	+/-	IF≥Ø	EXC	(?)	CONT	CD	STOP	=	K	
30	9	PRNT	PF	+/-	IF≥Ø	EXC	(?)	CONT	+/-	×	
40	K	6	+	K	5	=	K	4	+	K	
50	8	+	7	=	Rxx	Ø	Ø	Ø	Rxx	Rxx	
60	Ø	Ø	K	8	-	K	7	)	IF=Ø	K	
70	5	+	8	=	Rxx	Ø	2	(B)	(=)	(?)	
80	PF	(B)	(Y)	(SPC)	(C)	(O)	(L)	(')	(S)	PF	
90	PF	EXC	(B)	CONT	LBL	(?)	(I)	(N)	(D)	(X)	(?)
4 00	(?)	CD	PF	STOP	LBL	(A)	STFG	LBL	STRT	LBL	(A), STRT
10	Ø	CD	ADR	GODP	FTP	Ø	LBL	2	CD	ADR	Ø, 2
20	GODP	FTP	2	LBL	4	SFG	LBL	5	LBL	6	4, 5, 6
30	LBL	7	CD	ADR	GODP	FTP	3	LBL	3	SFG	7, 3
40	LBL	8	LBL	9	CD	ADR	GODP	FTP	4		8, 9
50											
60											
70											
80											
90											
	0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

		0	1	2	3	4	5	6	7	8	9	COMMENTS LBLs
0	00	EXC	2	LBL	B	K	4	+	7	=	LBL	B
	10	F	Rxx	Ø	1	+	1	=	Rxx	Ø	1	F
	20	CD	STOP	=	Rxx	Rxx	Ø	1	PRNT	Rxx	Ø	
	30	1	−	K	4	−	K	9	−	7	)	
	40	IF<Ø	EXC	F	CONT	CD	=	K	Ø	=	K	
	50	1	8	=	Rxx	Ø	Ø	+	K	4	−	
	60	1	=	LBL	H	Rxx	Ø	1	+	1	=	H
	70	Rxx	Ø	1	Rxx	Rxx	Ø	Ø	×	Rxx	Rxx	
	80	Ø	1	)	Σ_0	Rxx	Ø	Ø	+	K	7	
	90	=	Rxx	Ø	Ø	Rxx	Ø	1	−	K	4	
1	00	−	K	9	−	7	)	IF<Ø	EXC	H	CONT	
	10	K	Ø	=	Rxx	Rxx	Ø	2	CD	=	K	
	20	Ø	1	Σ_1	Rxx	Ø	[illegible]	+	1	=	Rxx	
	30	Ø	2	K	1	−	K	7	)	IF<Ø	Rxx	
	40	Ø	1	−	K	9	=	Rxx	Ø	1	Rxx	
	50	Ø	Ø	−	K	5	+	1	=	Rxx	Ø	
	60	Ø	EXC	H	CONT	Rxx	Ø	2	−	K	4	
	70	−	8	)	IF<Ø	PF	EXC	B	CONT	PF	B	
	80	*	A	=	EXC	W	K	,	L	=	CD	
	90	8	=	Rxx	Ø	Ø	+	K	5	=	Rxx	
2	00	Ø	1	K	8	=	K	6	PRNT	×	K	
	10	7	PRNT	PF	PF	=	K	5	LBL	X	Rxx	X
	20	Rxx	Ø	1	=	Rxx	Rxx	Ø	Ø	PRNT	Rxx	
	30	Ø	Ø	−	7	)	÷	K	7	=	K	
	40	Ø	−	K	Ø	int	)	IF=Ø	PF	CONT	Rxx	
		0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
											LBLs
50	Ø	1	+	1	=	Rxx	Ø	1	Rxx	Ø	
60	Ø	+	1	=	Rxx	Ø	Ø	-	K	5	
70	-	8	)	IF<Ø	EXC	(X)	CONT	(*)	(*)	CD	
80	PF	RSET	LBL	2	K	6	×	K	7	=	2
90	K	5	(B)	(*)	(A)	(:)	CLR	(B)	(:)	(SPC)	
3 00	(M)	(,)	(N)	(=)	(?)	STOP	=	K	8	PRNT	
10	+/-	IF≥Ø	EXC	(?)	CONT	+/-	×	K	7	+	
20	K	5	=	K	4	CD	STOP	=	K	9	
30	PRNT	PF	+/-	IF≥Ø	EXC	(?)	CONT	+/-	+	K	
40	4	+	7	=	Rxxx	Ø	Ø	Ø	Rxx	Rxx	
50	Ø	Ø	K	9	-	K	6	)	IF=Ø	K	
60	5	+	8	=	Rxx	Ø	2	(B)	(=)	(?)	
70	EXC	(W)	EXC	(B)	CONT	LBL	(?)	(I)	(N)	(D)	(?)
80	(X)	(?)	CD	PF	STOP	LBL	(W)	(B)	(Y)	(SPC)	(W)
90	(R)	(O)	(W)	(S)	RADR	PF	GODP	LBL	(A)	SFG	(A)
4 00	LBL	STRT	LBL	Ø	CD	ADR	GODP	FTP	Ø	LBL	STRT, Ø
10	1	CD	ADR	GODP	FTP	1	LBL	4	SFG	LBL	1, 4
20	5	LBL	6	LBL	7	CD	ADR	GODP	FTP	3	5, 6, 7
30	LBL	3	SFG	LBL	8	LBL	9	CD	ADR	GODP	3, 8, 9
40	FTP	4									
50											
60											
70											
80											
90											
	0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS LBLs
0	IFFG	CLFG	EXC	4	CONT	STOP	LBL	(P)	Rxx	Ø	(P)
10	Ø	−	7	)	÷	K	7	=	K	Ø	
20	−	K	Ø	int	)	IF=Ø	PF	CONT	Rxx	Ø	
30	Ø	+	1	=	Rxx	Ø	Ø	RADR	GODP	LBL	
40	(Q)	Rxx	Ø	1	+	1	=	Rxx	Ø	1	(Q)
50	RADR	GODP	LBL	(U)	RADR	=	K	1	CLR	8	(U)
60	=	Rxxx	Ø	Ø	Ø	+	K	5	=	K	
70	4	LBL	(V)	CD	STOP	=	PRNT	×	K	3	(V)
80	+	K	2	×	Rxx	Rxx	Ø	Ø	=	Rxx	
90	Rxx	Ø	Ø	EXC	(P)	Rxx	Ø	Ø	−	K	
1 00	4	)	IF<Ø	EXC	(V)	CONT	K	1	GODP	LBL	
10	(T)	Rxx	Rxx	Ø	1	=	Rxx	Rxx	Ø	Ø	(T)
20	PRNT	EXC	(P)	EXC	(Q)	Rxx	Ø	Ø	−	K	
30	5	−	8	)	IF<Ø	EXC	(T)	CONT	(*)	(*)	
40	CD	PF	RSET	LBL	7	(A)	(')	(=)	EXC	(W)	7
50	(K)	(,)	(L)	(=)	K	7	=	K	1	K	
60	6	=	K	7	K	1	=	K	6	PRNT	
70	×	K	7	PRNT	PF	PF	=	K	5	×	
80	2	=	K	4	+	7	=	Rxx	Ø	Ø	
90	Ø	Rxx	Rxx	Ø	Ø	EXC	(J)	Rxx	Ø	1	
2 00	=	K	1	LBL	(K)	Rxx	Rxx	Ø	Ø	=	(K)
10	Rxx	Rxx	Ø	1	EXC	(Q)	Rxx	Ø	Ø	+	
20	K	6	=	Rxx	Ø	Ø	−	K	1	)	
30	IF<Ø	EXC	(K)	CONT	Rxx	Ø	Ø	−	K	5	
40	+	1	=	Rxx	Ø	Ø	CD	1	Σ_1	Rxx	
	0	1	2	3	4	5	6	7	8	9	

	0	1	2	3	4	5	6	7	8	9	LBLs
50	Ø	Ø	-	K	4	-	8	)	IF<Ø	EXC	
60	K	CONT	EXC	J	EXC	T	LBL	O	K	,	O
70	L	=	K	6	PRNT	K	7	PRNT	PF	PF	
80	CD	8	=	Rxx	Ø	Ø	=	Rxx	Ø	1	
90	EXC	T	LBL	R	B	:	SPC	M	,	N	R
3 00	=	?	K		×	K	7	=	K	5	
10	CD	STOP	=	K	8	PRNT	CD	STOP	=	K	
20	9	PRNT	PF	-	K	7	)	IF=Ø	K	8	
30	-	K	6	)	IF=Ø	B	=	?	LBL	W	W
40	B	Y	SPC	R	O	W	S	RADR	PF	GODP	
50	CONT	I	N	D	X	?	PF	PF	STOP	LBL	
60	J	CD	8	=	Rxx	Ø	Ø	+	K	5	J
70	=	Rxx	Ø	1	RADR	GODP	LBL	4	A	+	4
80	B	:	EXC	R	CLR	1	=	K	2	=	
90	K	3	EXC	U	PF	A	+	B	=	EXC	
4 00	W	EXC	O	LBL	5	A	-	B	:	EXC	5
10	R	CLR	1	=	K	2	+/-	=	K	3	
20	EXC	U	PF	A	-	B	=	EXC	W	EXC	
30	O	LBL	6	B	-	A	:	EXC	R	CLR	6
40	1	=	K	3	+/-	=	K	2	EXC	U	
50	PF	B	-	A	=	EXC	W	EXC	O	LBL	
60	A	SFG	LBL	STRT	LBL	Ø	CD	ADR	GODP	FTP	A , STRT
70	Ø	LBL	1	CD	ADR	GODP	FTP	1	LBL	2	Ø, 1, 2
80	CD	ADR	GODP	FTP	2	LBL	3	SFG	LBL	8	3, 8
90	LBL	9	CD	ADR	GODP	FTP	4				9
	0	1	2	3	4	5	6	7	8	9	

COMMENTS

PROGRAM STEPS

		0	1	2	3	4	5	6	7	8	9	COMMENTS LBLs
0	00	IFFG	CLFG	EXC	3	CONT	STOP	LBL	(M)	CLFG	RADR	(M)
	10	=	K	1	K	2	×	Rxx	Rxx	Ø	Ø	
	20	=	EXC	(Z)	LBL	(N)	SFG	RADR	=	K	1	(N)
	30	EXC	(Z)	LBL	(<)	CLFG	RADR	=	K	1	Rxx	(<)
	40	Rxx	Ø	2	=	LBL	(Z)	Rxx	Rxx	Ø	Ø	(Z)
	50	PRNT	Rxx	Ø	2	+	1	=	Rxx	Ø	2	
	60	Rxx	Ø	Ø	+	1	=	Rxx	Ø	Ø	-	
	70	8	IFFG	+	1	CONT	)	÷	(	K	7	
	80	IFFG	+	1	CONT	=	K	Ø	-	K	Ø	
	90	int	)	IF=Ø	IFFG	Rxx	Rxx	Ø	Ø	Ø	Ø	
1	00	PRNT	Rxx	Ø	Ø	+	1	=	Rxx	Ø	Ø	
	10	CD	CONT	IF=Ø	PF	Rxx	Ø	2	+	1	=	
	20	Rxx	Ø	2	CONT	Rxx	Ø	Ø	-	K	5	
	30	-	8	IFFG	-	K	6	CONT	)	IF<Ø	K	
	40	1	-	2	)	GODP	CONT	K	1	GODP	LBL	
	50	8	(A)	(D)	(D)	(SPC)	(R)	(O)	(W)	(SPC)	(#)	8
	60	(')	(S)	(,)	(SPC)	(A)	(=)	EXC	(W)	EXC	(L)	
	70	K	6	+	7	=	Rxxx	Ø	Ø	Ø	-	
	80	K	6	=	Rxx	Ø	1	Rxx	Rxx	Ø	Ø	
	90	LBL	(S)	Rxx	Ø	Ø	-	7	)	÷	(	(S)
2	00	K	7	+	1	=	K	Ø	-	K	Ø	
	10	int	)	IF=Ø	K	1	=	Rxx	Rxx	Ø	Ø	
	20	CD	1	+/-	Σ_1	+	Rxx	Ø	Ø	=	Rxx	
	30	Ø	Ø	CONT	Rxx	Rxx	Ø	1	=	Rxx	Rxx	
	40	Ø	Ø	Rxx	Ø	1	-	1	=	Rxx	Ø	
		0	1	2	3	4	5	6	7	8	9	

	0	1	2	3	4	5	6	7	8	9	LBLs	COMMENTS
50	1	Rxx	Ø	Ø	−	1	=	Rxx	Ø	Ø		
60	+/−	+	7	)	IF<Ø	EXC	S	CONT	CD	8		
70	=	Rxx	Ø	Ø	EXC	N	CLFG	CD	PF	K		
80	6	−	K	7	)	x^a	)	IF=Ø	K	6		
90	=	K	8	+	1	=	K	9	L	O		
3 00	A	D	SPC	I	N	V	R	S	SPC	P		
10	R	O	G	CD	B	L	K	SPC	"	N		
20	"	;	SPC	E	X	C	SPC	M	CD	PF		
30	*	*	CD	PF	RSET	CONT	N	O	SPC	I		
40	N	V	R	S	CD	*	*	PF	PF	RSET		
50	LBL	3	S	*	A	=	EXC	W	S	=	3	
60	?	CD	STOP	=	K	2	PRNT	PF	EXC	L		
70	CLR	=	Rxxx	Ø	Ø	2	8	=	Rxx	Ø		
80	Ø	EXC	M	*	*	CD	PF	RSET	LBL	9	9	
90	D	L	T	SPC	R	O	W	SPC	#	'		
4 00	S	,	SPC	A	=	EXC	W	I	N	D		
10	X	SPC	O	K	?	STOP	EXC	L	CLR	8		
20	=	Rxxx	Ø	Ø	Ø	=	Rxx	Ø	2	EXC		
30	<	*	*	CD	PF	RSET	LBL	W	B	Y	W	
40	SPC	R	O	W	S	RADR	PF	GODP	LBL	L	L	
50	K	,	L	=	K	6	=	K	1	PRNT		
60	×	K	7	PRNT	PF	PF	=	K	5	+		
70	RADR	GODP	LBL	A	SFG	LBL	STRT	LBL	Ø	CD	A	, STRT, Ø
80	ADR	GODP	FTP	Ø	LBL	1	CD	ADR	GODP	FTP	1	
90	1	LBL	2	CD	ADR	GODP	FTP	2	LBL	4	2, 4	
	0	1	2	3	4	5	6	7	8	9		

PROGRAM STEPS

		0	1	2	3	4	5	6	7	8	9
5	00	SFG	LBL	5	LBL	6	LBL	7	CD	ADR	GODP
	10	FTP	3								
	20										
	30										
	40										
	50										
	60										
	70										
	80										
	90										
	00										
	10										
	20										
	30										
	40										
	50										
	60										
	70										
	80										
	90										
	00										
	10										
	20										
	30										
	40										
		0	1	2	3	4	5	6	7	8	9

COMMENTS

LBLs
5, 6, 7

NOTES

TITLE

Mathematics 4-4

MINIMUM HARDWARE REQUIRED

TEK 31, Printer (1024 Steps, 128 Registers)

REGISTERS USED

$K_{\emptyset} - K_9$: working

$R_{\emptyset\emptyset\emptyset}$: ind. address
$R_{\emptyset 1}$: ind. address
$R_{\emptyset 2}$: ind. address

$R_{\emptyset 8}$ through R_{xxx}
for matrix storage

Subroutine labels used
Ø through 9
(A) through (Z)
(?) , (*) ,
Σ_0, Σ_1, $_3\Sigma_2$,
IF<Ø, IF=Ø, IFFG,
STRT, ADR

PROGRAM DESCRIPTION

This program consists of two BLOCKS for performing the following matrix arithmetic operations.

BLOCK Ø

LBL STRT: Title and instructions
LBL (A) : $[A_{k1}]$ initial data entry
LBL Ø: $[A_{k1}] \times [A_{k1}] = [R_{k1}]$ where k = 1
LBL 1: $[A_{k1}] \times [B_{mn}] = [R_{kn}]$ where 1 = m
LBL 2: $[B_{mn}] \times [A_{k1}] = [R_{m1}]$ where n = k

BLOCK 1

LBL 3: $S \times [A_{k1}] = [R_{k1}]$, S a constant
LBL 4: $[A_{k1}] + [B_{mn}] = [R_{k1}]$ where k = m, 1 = n
LBL 5: $[A_{k1}] - [B_{mn}] = [R_{k1}]$ where k = m, 1 = n
LBL 6: $[B_{mn}] - [A_{k1}] = [R_{k1}]$ where k = m, 1 = n
LBL 7: Transpose $[A_{k1}] = [A]' = [R_{1k}]$
LBL 8: Adds row numbers to $[A_{k1}]$ as required for matrix inversion program 4-2.
LBL 9: Removes row numbers from $[A_{k1}]$ following operation of program 4-2.

The result of each operation is stored as matrix $[A_{k1}]$ in place of the previous data. Thus the two BLOCKS may be used to perform almost any sequence of matrix arithmetic operations, so long as they are mathematically defined and the calculator storage capacity is not exceeded.

Each BLOCK includes calling routines for automatic loading of the other BLOCK as a sequence of operations is executed. Each of the LABELed routines listed on page 1 runs automatically when called from either BLOCK.

For matrix multiplication, storage requirements will be minimized if the smaller of the two matrices is called matrix $[A_{k1}]$ since all elements of [A] must be stored initially, while only one column or row of [B] must be stored during operation of [A] × [B] or [B] × [A], respectively. Storage requirements for any of the operations may be determined from the following equations, where R is the number of R-registers installed in your calculator:

[A] × [A], [A]': $2k1 \leq R - 8$

[A] × [B]: $k1 + m + kn \leq R - 8$

[B] × [A]: $k1 + n + m1 \leq R - 8$

S × [A], [A] ± [B], [B] - [A]: $k1 \leq R - 8$

LBL 8, Add row numbers: $k(1 + 1) \leq R - 8$

Row and column indices k, 1 for [A] and m, n for [B] must be entered as indicated during operation of the program. If an inappropriate row/column index is entered, in most cases the program will print INDX ? and STOP to wait for your correction. Correction is easily accomplished by reexecuting the desired operation and reentering the correct matrix indices. Storage requirements are tested for each operation; if your calculator does not have sufficient storage for the matrix [A] and operation you have entered, the program will display the E2 "no such register" error message.

When LBL 8 is used to add row numbers for inversing by program 4-2, the program will print 4-2 loading instructions for any square matrix. For a non-square matrix, the program will add and print out the matrix with row numbers with the message "NO INVRS" to indicate that the non-square matrix cannot be inversed.

When LBL 9 is used to delete row numbers from a matrix, the program STOPs to ask "INDX OK?". At this point recall K_6 and K_7, and be sure that they are equal to the row (k) and column (1) indices, respectively, of the matrix. When deleting row numbers from a matrix inversed by program 4-2, it will be necessary to manually set K_6 and K_7 equal to the matrix order n (n = k = 1 for a square matrix; enter $n = K_6 = K_7$ manually).

NOTES

EXAMPLES

1. To perform the following series of operations, load BLK Ø

2. EXC STRT

3. Enter k = 3, l = 4

$$A_{kl} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix}$$

4. Multiply [A] × [B]:
   EXC 1
   Enter m = 4, n = 3

$$B_{mn} = \begin{bmatrix} 12 & 8 & 4 \\ 11 & 7 & 3 \\ 10 & 6 & 2 \\ 9 & 5 & 1 \end{bmatrix}$$

5. Multiply by scalar S = .01
   EXC 3 and the program will auto load BLOCK 1 and execute.
   Enter S = 0.01 Press CONT

6. Subtract [B]: EXC 5
   Enter m = 3, n = 3

$$B_{mn} = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 3 & 3 & 3 \end{bmatrix}$$

7. Multiply [B] × [A]:
   EXC 2 and the program will auto load BLOCK Ø and execute.
   Enter m = 4, n = 3

$$B_{mn} = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ -1 & -2 & -3 \\ -3 & -2 & -1 \end{bmatrix}$$

```
MTRX ARITH

FOR,EXC:
A*A, 0
A*B, 1
B*A, 2
S*A, 3
A+B, 4
A-B, 5
B-A, 6
A',  7

ROW #'S:
ADD, 8
DLT, 9

**

A: K,L=?
              3.
              4.

A=?
BY ROWS
              1.
              2.
              3.
              4.
              5.
              6.
              7.
              8.
              9.
             10.
             11.
             12.

**

A*B:
B: M,N=?
              3.
              4.

#  INDX?

A*B:
B: M,N=?
              4.
              3.

B=?
BY COL'S
             12.
             11.
             10.
              9.
              8.
              7.
              6.
              5.
              4.
              3.
              2.
              1.

A*B=
BY ROWS

K,L=
              3.
              3.

            100.
             60.
             20.

            268.
            164.
             60.

            436.
            268.
            100.

**

S*A=
BY ROWS

S=?
            .01

K,L=
              3.
              3.

              1.
             .6
             .2

           2.68
           1.64
             .6

           4.36
           2.68
              1.

**

A-B:
B: M,N=?
              3.
              3.

B=?
BY ROWS
              1.
              1.
              1.
              2.
              2.
              2.
              3.
              3.
              2.
              1.

**

A-B=
BY ROWS

K,L=
              3.
              3.

              0.
-            .4
-            .8

            .68
-           .36
-          1.4

           1.36
-           .32
-            2.

**

B*A:
B: M,N=?
              4.
              3.

B=?
BY ROWS
              1.
              2.
              3.
              3.
              2.
              1.
-             1.
-             2.
-             3.
-             3.
-             2.
-             1.

B*A=
BY ROWS

K,L=
              4.
              3.

           5.44
-          2.08
-           9.6

           2.72
-          2.24
-           7.2

-          5.44
           2.08
            9.6

-          2.72
           2.24
            7.2

**
```

Note the reversal error in entry of m and n indicated by the INDX ? message. Correct by pressing EXC 1 again and reentering m = 4 and n = 3.

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1	Load Block Ø	RSET FTP Ø		
2	Print Title	EXC STRT		Title and EXC instructions
3	k	CONT		k
	1	CONT		1
	a_{11}	CONT		a_{11}
	⋮	⋮		⋮
	a_{k1}	CONT		a_{k1}
4	Select desired operation			as required
	A × A	EXC Ø		s
	A × B	EXC 1		m
	B × A	EXC 2		n
	S × A	EXC 3		b elements
	A + B	EXC 4		
	A − B	EXC 5		Results
	B − A	EXC 6		k
	A'	EXC 7		1
	Add row numbers	EXC 8		a_{11}
	Delete row numbers	EXC 9		⋮
				a_{k1}
5	Enter S, m, n, [B], etc., as requested by the PRINTOUT.			See examples for detailed printouts.
6	To perform another operation on the result of 4 and 5, return to Step 4.			
7	To enter a new matrix A without reprinting the title, etc., press EXC (A) .			
	Note that LBLs (A) , Ø through 9, and STRT execute automatically from either BLOCK.			

PROGRAM STEPS

		0	1	2	3	4	5	6	7	8	9	COMMENTS LBLs
0	50	IFFG	CLFG	K	1	GODP	CONT	EXC	STRT	LBL	Σ_1	Σ_1
	60	Rxx	Ø	1	+	1	=	Rxx	Ø	1	RADR	
	70	GODP	LBL	(P)	RADR	=	K	1	CD	STOP	=	(P)
	80	LBL	(Y)	Rxx	Rxx	Ø	Ø	PRNT	Rxx	Ø	Ø	(Y)
	90	-	7	)	÷	K	7	=	K	Ø	-	
	00	K	Ø	int	)	IF=Ø	PF	CONT	EXC	Σ_1	Rxx	
	10	Ø	Ø	+	1	=	Rxx	Ø	Ø	-	K	
	20	5	-	8	)	IF<Ø	K	1	-	2	)	
	30	GODP	CONT	(*)	(*)	CD	PF	RSET	LBL	(B)	K	(B)
	40	4	+	7	=	LBL	(E)	EXC	Σ_1	CD	STOP	(E)
1	50	=	Rxx	Rxx	Ø	1	PRNT	Rxx	Ø	1	-	
	60	K	4	-	K	2	-	7	)	IF<Ø	EXC	
	70	(E)	CONT	CD	=	K	Ø	=	K	1	8	
	80	=	Rxx	Ø	Ø	+	K	4	-	1	=	
	90	LBL	(G)	EXC	Σ_1	Rxx	Rxx	Ø	Ø	×	Rxx	(G)
	00	Rxx	Ø	1	)	Σ_0	Rxx	Ø	Ø	+	1	
	10	IFFG	×	K	7	CONT	=	Rxx	Ø	Ø	Rxx	
	20	Ø	1	-	K	4	-	K	2	-	7	
	30	)	IF<Ø	EXC	(G)	CONT	K	Ø	=	Rxx	Rxx	
	40	Ø	2	CD	=	K	Ø	1	Σ_1	Rxx	Ø	
2	50	2	+	K	9	IFFG	x^a	CONT	=	Rxx	Ø	
	60	2	K	1	-	K	3	)	IF<Ø	EXC	IF<Ø	
	70	CONT	Rxx	Ø	2	-	IFFG	EXC	IFFG	CONT	K	
	80	6	×	K	9	+	1	=	Rxx	Ø	2	
	90	-	K	5	-	K	9	-	8	)	IF<Ø	
		0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS LBLs
00	PF	EXC	(B)	CONT	PF	(A)	(*)	(B)	(=)	LBL	
10	(H)	EXC	(W)	(K)	(,)	(L)	(=)	CD	8	=	(H)
20	Rxx	Ø	Ø	+	K	5	=	Rxx	Ø	1	
30	PF	IFFG	K	8	=	CONT	K	6	PRNT	IFFG	
40	×	K	7	PRNT	PF	PF	=	K	5	EXC	
3 50	(Q)	CONT	K	9	=	K	7	PRNT	PF	PF	
60	×	K	6	=	K	5	EXC	(Q)	LBL	(Q)	(Q)
70	RADR	=	K	1	Rxx	Rxx	Ø	1	=	EXC	
80	(Y)	LBL	IF<Ø	Rxx	Ø	1	-	K	2	=	IF<Ø
90	Rxx	Ø	1	IFFG	Rxx	Ø	Ø	-	K	5	
00	+	1	=	Rxx	Ø	Ø	CONT	EXC	(G)	LBL	
10	IFFG	K	4	-	8	)	IF<Ø	PF	EXC	(B)	IFFG
20	CONT	PF	(B)	(*)	(A)	(=)	EXC	(H)	LBL	Ø	Ø
30	(A)	(*)	(A)	(=)	EXC	(W)	(K)	(,)	(L)	(=)	
40	K	6	PRNT	-	K	7	PRNT	PF	PF	)	
4 50	x^2	+/-	IF<Ø	EXC	(?)	CONT	K	6	x^2	=	
60	K	5	×	2	=	K	4	EXC	(I)	CD	
70	=	K	Ø	EXC	(J)	LBL	(C)	CD	8	=	(C)
80	Rxx	Ø	2	+	K	5	=	K	1	LBL	
90	(D)	Rxx	Rxx	Ø	Ø	×	Rxx	Rxx	Ø	2	(D)
00	)	Σ_0	Rxx	Ø	Ø	+	1	=	Rxx	Ø	
10	Ø	Rxx	Ø	2	+	K	7	=	Rxx	Ø	
20	2	-	K	1	)	IF<Ø	EXC	(D)	CONT	K	
30	Ø	=	Rxx	Rxx	Ø	1	CD	=	K	Ø	
40	1	Σ_1	EXC	Σ_1	Rxx	Ø	2	-	K	5	
	0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

		0	1	2	3	4	5	6	7	8	9	COMMENTS
												LBLs
5	00	-	K	7	-	7	)	IF<Ø	Rxx	Ø	Ø	
	10	-	K	7	=	Rxx	Ø	Ø	Rxx	Ø	2	
	20	-	K	5	+	1	=	Rxx	Ø	2	EXC	
	30	D	CONT	Rxx	Ø	1	-	K	4	-	8	
	40	)	IF<Ø	EXC	C	CONT	EXC	J	EXC	Q	LBL	
	50	J	CD	8	=	Rxx	Ø	Ø	+	K	5	J
	60	=	Rxx	Ø	1	RADR	GODP	LBL	1	CLFG	A	1
	70	*	B	:	EXC	X	K	6	+	K	5	
	80	=	K	4	+	K	8	EXC	I	K	6	
	90	=	K	3	K	8	=	K	2	-	K	
6	00	7	)	IF=Ø	EXC	IF=Ø	CONT	EXC	?	LBL	2	2
	10	SFG	B	*	A	:	EXC	X	CLR	K	7	
	20	=	K	3	×	K	8	+	K	5	=	
	30	K	4	+	K	9	EXC	I	K	9	=	
	40	K	2	-	K	6	)	IF=Ø	EXC	IF=Ø	CONT	
	50	EXC	?	LBL	X	K	6	×	K	7	=	X
	60	K	5	B	:	SPC	M	,	N	=	?	
	70	CD	STOP	=	K	8	PRNT	+/-	IF≥Ø	EXC	?	
	80	CONT	CD	STOP	=	K	9	PRNT	PF	+/-	IF≥Ø	
	90	EXC	?	CONT	+/-	×	RADR	GODP	LBL	IF=Ø	K	IF=Ø
7	00	5	+	8	=	Rxx	Ø	2	B	=	?	
	10	IFFG	EXC	W	EXC	B	CONT	B	Y	SPC	C	
	20	O	L	'	S	PF	PF	EXC	B	LBL	I	I
	30	+	7	=	Rxxx	Ø	Ø	Ø	Rxx	Rxx	Ø	
	40	Ø	RADR	GODP	LBL	W	B	Y	SPC	R	O	W
		0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

		0	1	2	3	4	5	6	7	8	9	COMMENTS LBLs
	50	W	S	RADR	PF	GODP	LBL	?	I	N	D	?
	60	X	?	CD	PF	STOP	LBL	STRT	M	T	R	STRT
	70	X	SPC	A	R	I	T	H	CLR	PF	F	
	80	O	R	,	E	X	C	:	PF	A	*	
	90	A	,	SPC	Ø	PF	A	*	B	,	SPC	
8	00	1	PF	B	*	A	,	SPC	2	PF	S	
	10	*	A	,	SPC	3	PF	A	+	B	,	
	20	SPC	4	PF	A	-	B	,	SPC	5	PF	
	30	B	-	A	,	SPC	6	PF	A	'	,	
	40	SPC	SPC	7	PF	PF	R	O	W	SPC	#	
	50	'	S	:	PF	A	D	D	,	SPC	8	
	60	PF	D	L	T	,	SPC	9	PF	PF	*	
	70	*	PF	PF	LBL	A	A	:	SPC	K	,	A
	80	L	=	?	CLR	STOP	=	K	6	PRNT	CD	
	90	STOP	=	K	7	PRNT	×	K	6	=	K	
9	00	5	IF=Ø	EXC	?	CONT	EXC	I	A	=	?	
	10	EXC	W	CD	=	Rxx	Ø	1	8	=	Rxx	
	20	Ø	Ø	EXC	P	LBL	3	CLR	6	4	2	3
	30	EXC	ADR	LBL	4	CLR	6	7	3	EXC	ADR	4
	40	LBL	5	CLR	7	Ø	Ø	EXC	ADR	LBL	6	5, 6
	50	CLR	7	2	8	EXC	ADR	LBL	7	CLR	2	7
	60	5	4	EXC	ADR	LBL	8	CLR	3	8	Ø	8
	70	EXC	ADR	LBL	9	CLR	5	7	2	LBL	ADR	9, ADR
	80	=	K	1	CD	ADR	GODP	FTP	1			
	90											
		0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS LBLs
0	CLR	K	1	GODP	LBL	U	RADR	=	K	1	U
10	CLR	8	=	Rxxx	Ø	Ø	Ø	+	K	5	
20	=	K	4	LBL	V	CD	STOP	=	PRNT	×	V
30	K	3	+	K	2	×	Rxx	Rxx	Ø	Ø	
40	=	Rxx	Rxx	Ø	Ø	Rxx	Ø	Ø	-	7	
50	)	÷	K	7	=	K	Ø	-	K	Ø	
60	int	)	IF=Ø	PF	CONT	EXC	Σ_0	Rxx	Ø	Ø	
70	-	K	4	)	IF<Ø	EXC	V	CONT	K	1	
80	GODP	LBL	Σ_0	Rxx	Ø	Ø	+	1	=	Rxx	Σ_0
90	Ø	Ø	RADR	GODP	LBL	Σ_1	Rxx	Ø	1	+	Σ_1
1 00	1	=	Rxx	Ø	1	RADR	GODP	LBL	$_3\Sigma_2$	Rxx	$_3\Sigma_2$
10	Ø	2	+	1	=	Rxx	Ø	2	RADR	GODP	
20	LBL	M	CLFG	RADR	=	K	1	K	2	×	M
30	Rxx	Rxx	Ø	Ø	=	EXC	Z	LBL	N	SFG	N
40	RADR	=	K	1	EXC	Z	LBL	T	CLFG	RADR	T
50	=	K	1	Rxx	Rxx	Ø	1	=	EXC	Z	
60	LBL	F	CLFG	RADR	=	K	1	Rxx	Rxx	Ø	F
70	2	=	LBL	Z	Rxx	Rxx	Ø	Ø	PRNT	EXC	Z
80	Σ_0	EXC	Σ_1	EXC	$_3\Sigma_2$	Rxx	Ø	Ø	-	8	
90	IFFG	+	1	CONT	)	÷	(	K	7	IFFG	
2 00	+	1	CONT	=	K	Ø	-	K	Ø	int	
10	)	IF=Ø	IFFG	Rxx	Rxx	Ø	Ø	Ø	Ø	PRNT	
20	EXC	Σ_0	CD	CONT	IF=Ø	PF	EXC	$_3\Sigma_2$	CONT	Rxx	
30	Ø	Ø	-	K	5	-	8	IFFG	-	K	
40	6	CONT	)	IF<Ø	K	1	-	2	)	GODP	
	0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
											LBLs
	50	CONT	K	1	GODP	LBL	7	A	'	=	EXC	7
	60	W	K	,	L	=	K	7	=	K	1	
	70	K	6	=	K	7	K	1	=	K	6	
	80	PRNT	×	K	7	PRNT	PF	PF	=	K	5	
	90	×	2	=	K	4	+	7	=	Rxxx	Ø	
3	00	Ø	Ø	Rxx	Rxx	Ø	Ø	EXC	J	Rxx	Ø	
	10	1	=	K	1	LBL	K	Rxx	Rxx	Ø	Ø	K
	20	=	Rxx	Rxx	Ø	1	EXC	Σ_1	Rxx	Ø	Ø	
	30	+	K	6	=	Rxx	Ø	Ø	-	K	1	
	40	)	IF<Ø	EXC	K	CONT	Rxx	Ø	Ø	-	K	
	50	5	+	1	=	Rxx	Ø	Ø	CD	1	Σ_1	
	60	Rxx	Ø	Ø	-	K	4	-	K	8	)	
	70	IF<Ø	EXC	K	CONT	EXC	J	EXC	T	EXC	*	
	80	LBL	8	A	D	D	SPC	R	O	W	SPC	8
	90	#	'	S	,	SPC	A	=	EXC	W	EXC	
4	00	L	K	6	+	7	=	Rxxx	Ø	Ø	Ø	
	10	-	K	6	=	Rxx	Ø	1	Rxx	Rxx	Ø	
	20	Ø	LBL	S	Rxx	Ø	Ø	-	7	)	÷	S
	30	(	K	7	+	1	=	K	Ø	-	K	
	40	Ø	int	)	IF=Ø	K	1	=	Rxx	Rxx	Ø	
	50	Ø	CD	1	+/-	Σ_1	+	Rxx	Ø	Ø	=	
	60	Rxx	Ø	Ø	CONT	Rxx	Rxx	Ø	1	=	Rxx	
	70	Rxx	Ø	Ø	Rxx	Ø	1	-	1	=	Rxx	
	80	Ø	1	Rxx	Ø	Ø	-	1	=	Rxx	Ø	
	90	Ø	+/-	+	7	)	IF<Ø	EXC	S	CONT	CD	
		0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

		0	1	2	3	4	5	6	7	8	9	COMMENTS LBLs
5	00	8	=	Rxx	Ø	Ø	EXC	N	CLFG	CD	PF	
	10	K	6	-	K	7	)	x^a	)	IF=Ø	K	
	20	6	=	K	8	+	1	=	K	9	L	
	30	O	A	D	SPC	I	N	V	R	S	SPC	
	40	P	R	G	M	SPC	4	-	2	;	SPC	
	50	E	X	C	SPC	M	CD	PF	EXC	*	CONT	
	60	N	O	SPC	I	N	V	R	S	CD	PF	
	70	EXC	*	LBL	9	D	L	T	SPC	R	O	9
	80	W	SPC	#	'	S	,	SPC	A	=	EXC	
	90	W	I	N	D	X	SPC	O	K	?	CLR	
6	00	STOP	EXC	L	CLR	8	=	Rxxx	Ø	Ø	Ø	
	10	=	Rxx	Ø	2	EXC	F	EXC	*	LBL	L	L
	20	K	,	L	=	K	6	=	K	1	PRNT	
	30	×	K	7	PRNT	PF	PF	=	K	5	+	
	40	RADR	GODP	LBL	3	S	*	A	=	EXC	W	3
	50	S	=	?	CD	STOP	=	K	2	PRNT	PF	
	60	EXC	L	CLR	8	=	Rxxx	Ø	Ø	Ø	EXC	
	70	M	EXC	*	LBL	4	A	+	B	:	EXC	4
	80	R	CLR	1	=	K	2	=	K	3	EXC	
	90	U	PF	A	+	B	=	EXC	W	EXC	O	
7	00	LBL	5	A	-	B	:	EXC	R	CLR	1	5
	10	=	K	2	+/-	=	K	3	EXC	U	PF	
	20	A	-	B	=	EXC	W	EXC	O	LBL	6	6
	30	B	-	A	:	EXC	R	CLR	1	=	K	
	40	3	+/-	=	K	2	EXC	U	PF	B	-	
		0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
	A	=	EXC	W	EXC	O	LBL	R	B	:	LBLs R
	SPC	M	,	N	=	?	K	6	×	K	
	7	=	K	5	CD	STOP	=	K	8	PRNT	
	CD	STOP	=	K	9	PRNT	PF	−	K	7	
	)	IF=Ø	K	8	−	K	6	)	IF=Ø	B	
8	=	?	LBL	W	B	Y	SPC	R	O	W	W
	S	RADR	PF	GODP	CONT	I	N	D	X	?	
	PF	PF	STOP	LBL	O	K	,	L	=	K	O
	6	PRNT	K	7	PRNT	PF	PF	CD	8	=	
	Rxx	Ø	Ø	=	Rxx	Ø	1	EXC	T	LBL	
	*	*	*	CD	PF	RSET	LBL	J	CD	8	* , J
	=	Rxx	Ø	Ø	+	K	5	=	Rxx	Ø	
	1	RADR	GODP	LBL	STRT	CLR	7	6	5	EXC	STRT
	ADR	LBL	A	CLR	8	7	3	EXC	ADR	LBL	A
	Ø	CLR	3	7	8	EXC	ADR	LBL	1	CLR	Ø, 1
9	5	6	6	EXC	ADR	LBL	2	CLR	6	0	2
	8	LBL	ADR	=	K	1	SFG	CD	ADR	GODP	ADR
	FTP	Ø									
	0	1	2	3	4	5	6	7	8	9	

NOTES

SECTION 5 VECTORS

5-1 Vector Addition and Subtraction, Polar and Spherical Coordinates

5-2 Vector Cross Products, Rectangular and Spherical Coordinates

5-3 Rotation and Translation of Axes, Rectangular and Polar Coordinates

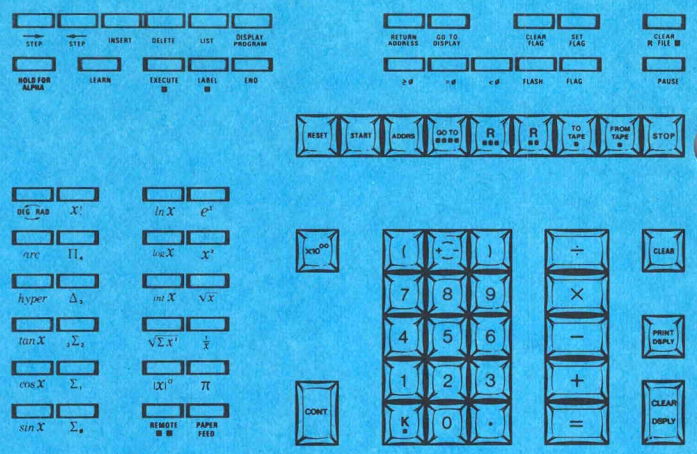
5-4 Coordinate Transforms: Rectangular to Polar or Spherical, and Inverse

5-5 Vector Inner (Scalar) Product and Angle, N-Dimensional Vectors

OCTAL CODE	PRINT OUT	KEY	SYMBOL
001	LBL_	LABEL ■	LBL
002	FTP_	FROM TAPE ■	FTP
003	TTP_	TO TAPE ■	TTP
004	EXC_	EXECUTE ■	EXC
005	CFI_	CLEAR R FILE ■	CFI
007	GOTO	GO TO ■■■■	GT
010	R___	R ■■■	Rxxx
011	R__	R ■■	Rxx
040	CLDP	CLEAR DSPLY	CD
041	IFFL	FLASH	IFFL
042	S FG	SET FLAG	SFG
043	STOP	STOP	STOP
044	PRNT	PRINT DSPLY	PRNT
045	CLR	CLEAR	CLR
046	PAUS	PAUSE	PAUS
047	CLFG	CLEAR FLAG	CLFG
050	(	(	(
051	)	)	)
052	*	×	×
053	+	+	+
054	RSET	RESET	RSET
055	-	−	-
056	.	•	•
057	/	÷	÷
060	0	0	ø
061	1	1	1
062	2	2	2
063	3	3	3
064	4	4	4
065	5	5	5
066	6	6	6
067	7	7	7
070	8	8	8
071	9	9	9
072	ADR	ADDRS	ADR
073	STRT	START	STRT
074	IF<0	<ø	IF<ø
075	=	=	=
076	IF>=	≥ø	IF≥ø
077	IF=0	=ø	IF=ø

OCTAL CODE	PRINT OUT	KEY	SYMBOL
100	K_	K	K
101	DG/R	DEG RAD	D/R
102	ARC	arc	arc
103	HYP	$hyper$	hyp
104	TAN	$\tan x$	tan
105	COS	$\cos x$	cos
106	SIN	$\sin x$	sin
107	X!	$x!$	x!
110	PI4	Π_4	Π_4
111	DLT3	Δ_3	Δ_3
112	3SM2	${}_3\Sigma_2$	${}_3\Sigma_2$
113	SUM1	Σ_1	Σ_1
114	SUM0	$\Sigma_ø$	Σ_0
115	LN	$\ln x$	ln
116	LOG	$\log x$	log
117	INT	$int\,x$	int
120	RSSQ	$\sqrt{\Sigma x^2}$	$\sqrt{\Sigma x^2}$
121	X↑A	$\|x\|^a$	x^a
122	RM__	REMOTE ■■	RMT
123	E↑X	e^x	e^x
124	X↑2	x^2	x^2
125	SQRT	$\sqrt{x}$	$\sqrt{x}$
126	1/X	$\frac{1}{x}$	1/x
127	PI	π	π
130	PAPR	PAPER FEED	PF
131	*10↑	$\times 10^{oo}$	10^{oo}
132	CONT	CONT	CONT
133	R AD	RETURN ADDRESS	RADR
134	+/-	+/-	+/-
135	GODP	GO TO DISPLAY	GODP
136	IFFG	FLAG	IFFG
137	ENDR	END	ENDR

OCTAL CODE	PRINT OUT	KEY	SYMBOL
201		LABEL ■	LBL
202		FROM TAPE ■	FTP
203		TO TAPE ■	TTP
204		EXECUTE ■	EXC
205		CLEAR R FILE ■	CFI
207		BELL	BELL
210		SPACE ←	SPC←
211		TAB ←	TAB
220		STEP	STP←
221		INSERT	NSRT
222		DELETE	DLT
223		STEP →	STP→
224		LIST	LIST
225		DISPLAY PROGRAM	DPRG
226		LEARN	LRN
240		SPACE	SPC
241	!	!	!
242	"	"	"
243	#	#	#
244	$	$	$
245	%	%	%
246	&	&	&
247	'	'	'
250	(	(	(
251	)	)	)
252	*	*	*
253	+	+	+
254	,	,	,
255	-	—	-
256	.	.	.
257	/	/	/
260	0	0	∅
261	1	1	1
262	2	2	2
263	3	3	3
264	4	4	4
265	5	5	5
266	6	6	6
267	7	7	7
270	8	8	8
271	9	9	9
272	:	:	:
273	;	;	;
274	<	<	<
275	=	=	=
276	>	>	>
277	?	?	?



OCTAL CODE	PRINT OUT	KEY	SYMBOL
300	@	@	@
301	A	A	A
302	B	B	B
303	C	C	C
304	D	D	D
305	E	E	E
306	F	F	F
307	G	G	G
310	H	H	H
311	I	I	I
312	J	J	J
313	K	K	K
314	L	L	L
315	M	M	M
316	N	N	N
317	O	O	O
320	P	P	P
321	Q	Q	Q
322	R	R	R
323	S	S	S
324	T	T	T
325	U	U	U
326	V	V	V
327	W	W	W
330	X	X	X
331	Y	Y	Y
332	Z	Z	Z
333	[	[	[
334	\	\	\
335	]	]	]
336	↑	↑	↑
337	_	_	_

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

$K_{\emptyset}$: dimension
K_1 : r_1, r'
K_2 : θ_1, θ'
K_3 : ϕ_1, ϕ'
K_4 : r_2
K_5 : θ_2
K_6 : ϕ_2
K_7 : working
K_8 : working
K_9 : RADR

$R_{\emptyset\emptyset 1}$: r_1
$R_{\emptyset 2}$: θ_1
$R_{\emptyset 3}$: ϕ_1

Subroutine Labels:

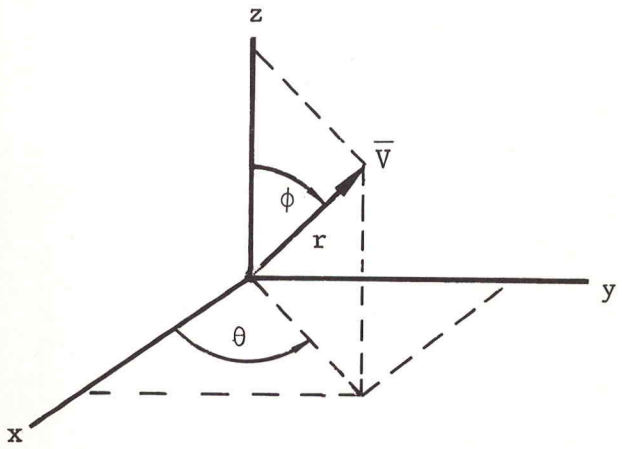
(C) (F) (L) (W)

(X) D/R, +, -, 10^^,

IFFG, ENDR

PROGRAM DESCRIPTION

This program adds or subtracts vectors in 2-dimensional polar coordinates, or in 3-dimensional spherical coordinates:



Given the vectors

$$\overline{V}_1 = r_1, \theta_1 \quad \text{or} \quad = r_1, \theta_1, \phi_1$$

$$\overline{V}_2 = r_2, \theta_2 \quad \text{or} \quad = r_2, \theta_2, \phi_2$$

then

$$\overline{V}_1 + \overline{V}_2 = r', \theta' \quad \text{or} \quad = r', \theta', \phi'$$

$$\overline{V}_1 - \overline{V}_2 = \overline{V}_1 + (-\overline{V}_2) = r', \theta' \quad \text{or} \quad = r', \theta', \phi'$$

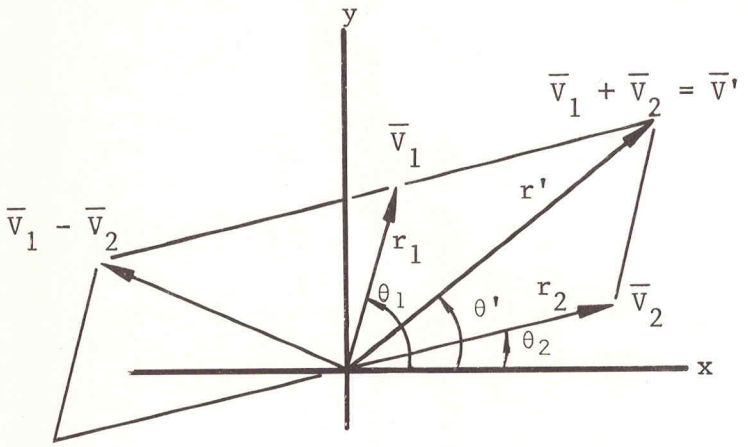
The program operates in degrees or radians as selected by the operator. Addition or subtraction is selected simply by EXC + or EXC - commands. Input data, selected operators, and results are automatically printed out.

Each calculated result is stored in place of $\overline{V}_1$, and the calculator stops ready for selection of a new operator and vector data entry. Thus a series of operations can be chained together as

$$\overline{V}_1 + \overline{V}_2 = \overline{V}' - \overline{V}_3 = \overline{V}' - \overline{V}_4 = \overline{V}' + \ldots - \ldots + \overline{V}_n = \overline{V}'$$

with all the data, operators, and intermediate results automatically printed out. Note that with each operation the original content of K_1, K_2, and K_3 is stored in $R_{\emptyset\emptyset 1}$, $R_{\emptyset 2}$, and $R_{\emptyset 3}$ for verification when the printer is not used.

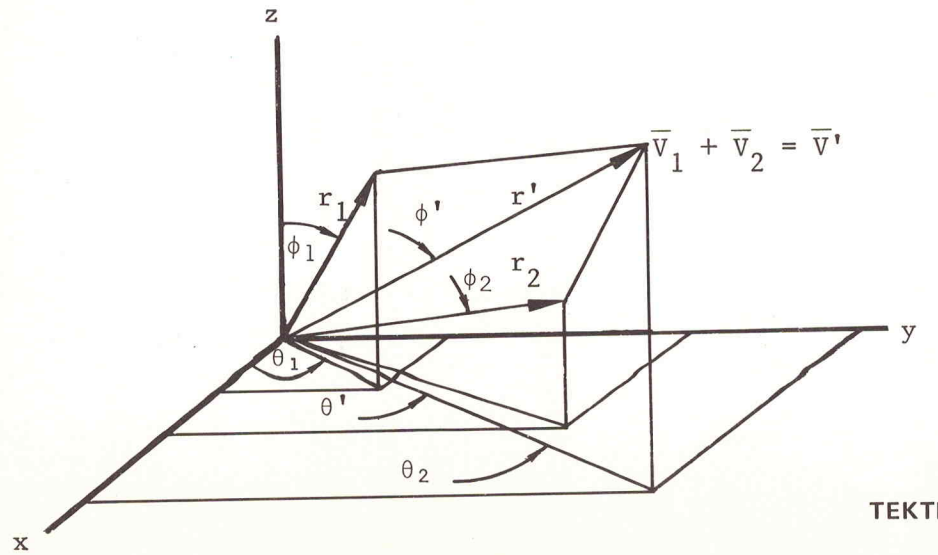
2-dimensional polar coordinates:



when $-\pi \leq \theta_1' \leq \pi$

or $-180^o \leq \theta_1' \leq 180^o$

3-dimensional spherical coordinates:



NOTES

EXAMPLES

1. 2-dimensions

$\overline{V}_1 = (4, 60^o)$

$\overline{V}_2 = (3, -30^o)$

$\overline{V}_1 + \overline{V}_2 = (5, 23.13010235^o)$

$\overline{V}_1 - \overline{V}_2 = (5, 96.86989765^o)$

2. 3-dimensions

$\overline{V}_1 = (3, 30^o, 90^o)$

$\overline{V}_2 = (4, -60^o, 90^o)$

$\overline{V}_3 = (12, 0^o, 0^o)$

$\overline{V}_1 + \overline{V}_2 =$
$(5, -23.13010235^o, 90^o)$

$\overline{V}_1 - \overline{V}_2 =$
$(5, 83.13010235^o, 90^o)$

$+ \overline{V}_3 =$
$(13, 83.13010235^o, 22.61986495^o)$

```
VECTOR ADD & SUB

2- OR 3-DIM.?
             2.

SET FLAG FOR DEG
CLR FLAG FOR RAD
#
R 1
             4.
0 1
            60.
DEG

+
R 2
             3.
0 2
            30.
-
DEG

=

R':K1
             5.
0':K2
  23.13010235
DEG

#
R 1
             4.
0 1
            60.
DEG

-
R 2
             3.
0 2
            30.
-
DEG

=

R':K1
             5.
0':K2
  96.86989765
DEG

END
```

```
VECTOR ADD & SUB

2- OR 3-DIM.?
             3.

SET FLAG FOR DEG
CLR FLAG FOR RAD
#
R 1
             3.
0 1
            30.
DEG
PHI 1
            90.
DEG

+
R 2
             4.
0 2
            60.
-
DEG
PHI 2
            90.
DEG

=

R':K1
             5.
0':K2
  -23.13010235
DEG
PHI':K3
            90.
DEG

#
R 1
             3.
0 1
            30.
DEG
PHI 1
            90.
DEG

-
R 2
             4.
0 2
            60.
-
DEG
PHI 2
            90.
DEG

=

R':K1
             5.
0':K2
  83.13010235
DEG
PHI':K3
            90.
DEG

+
R 2
            12.
0 2
             0.
DEG
PHI 2
             0.
DEG

=

R':K1
            13.
0':K2
  83.13010235
DEG
PHI':K3
  22.61986495
DEG

END
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1		EXC (C)		TITLE
2	Enter DIMENSION	2 or 3, CONT		DIM instruction
3	For DEG	SET FLAG		FLAG instruction
	For RAD	CLR FLAG		
				#
4	For data entry	EXC D/R	0	
	enter initial r_1	CONT	0	r_1
	θ_1	CONT	0	θ_1
	(3-DIM only) ϕ_1	CONT	0	ϕ_1
5	Select operator			
	for ADD	EXC +	0	+
	for SUBTRACT	EXC -	0	-
6	Enter r_2	CONT	0	r_2
	θ_2	CONT	0	θ_2
	(3-DIM only) ϕ_2	CONT	π*	ϕ_2
7	To add or subtract another vector to this result return to Step 5.			=
				$r' : K_1$
				$\theta' : K_2$
8*	To add or subtract a new pair of vectors, return to 3.			$\phi' : K_3$
9	To terminate	EXC ENDR		

The DIMENSION entry may be changed from 2 to 3 or 3 to 2 prior to Step 4 without reprinting the title:

for 2	CD 2 = $K_\emptyset$
for 3	CD 3 = $K_\emptyset$

*If the π STOP steps are replaced with EXC D/R as shown on the PROGRAM STEP list, the program will automatically return to Step 4 and await entry of new vector data.

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
0	LBL	C	PF	PF	V	E	C	T	O	R	LBL C
10	SPC	A	D	D	SPC	&	SPC	S	U	B	
20	PF	2	−	SPC	O	R	SPC	3	−	D	
30	I	M	.	?	CD	STOP	=	K	Ø	PRNT	
40	CD	PF	S	E	T	SPC	F	L	A	G	
50	SPC	F	O	R	SPC	D	E	G	C	L	
60	R	SPC	F	L	A	G	SPC	F	O	R	
70	SPC	R	A	D	RSET	LBL	D/R	PF	#	CLR	LBL D/R
80	PF	R	SPC	1	STOP	=	K	1	PRNT	Ø	
90	SPC	1	CD	STOP	=	K	2	PRNT	EXC	IFFG	
1 00	K	Ø	−	3	)	IF=Ø	P	H	I	SPC	
10	1	CD	STOP	=	K	3	PRNT	EXC	IFFG	CONT	
20	CD	STOP	LBL	−	PF	−	EXC	W	K	4	LBL −
30	+/−	=	K	4	EXC	X	LBL	+	PF	+	LBL +
40	EXC	W	LBL	X	EXC	F	CLR	IFFG	D/R	CONT	LBL X
50	K	Ø	−	3	)	IF=Ø	K	4	×	K	
60	6	sin	=	K	7	K	1	×	K	3	
70	sin	=	K	9	×	K	2	cos	+	K	
80	7	×	K	5	cos	=	K	8	K	9	
90	×	K	2	sin	+	K	7	×	K	5	
2 00	sin	=	K	9	$\sqrt{\Sigma x^2}$	K	8	=	K	7	
10	EXC	10^{00}	K	1	×	K	3	cos	+	K	
20	4	×	K	6	cos	=	K	9	$\sqrt{\Sigma x^2}$	K	
30	7	=	K	1	1/x	×	K	9	)	arc	
40	cos	=	×	K	1	x^a	=	K	3	EXC	
	0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
50	L	CONT	K	1	×	K	2	cos	+	K	
60	4	×	K	5	cos	=	K	8	K	1	
70	×	K	2	sin	+	K	4	×	K	5	
80	sin	=	K	9	$\sqrt{\Sigma x^2}$	K	8	=	K	7	
90	=	K	1	EXC	10^{oo}	LBL	L	CLR	IFFG	D/R	LBL L
3 00	CONT	PF	R	'	:	K	1	K	1	PRNT	
10	Ø	/	:	K	2	K	2	PRNT	EXC	IFFG	
20	K	Ø	-	3	)	IF=Ø	P	H	I	'	
30	:	K	3	K	3	PRNT	EXC	IFFG	CONT	π#	
40	STOP#	LBL	IFFG	CD	IFFG	D	E	G	π	CONT	LBL IFFG
50	IF=Ø	R	A	D	CONT	RADR	GODP	LBL	10^{oo}	K	LBL 10^{oo}
60	9	10^{oo}	=	10^{oo}	Ø	Ø	e^x	int	x^a	×	
70	2	-	1	=	×	K	7	x^a	×	(	
80	K	8	÷	K	7	)	arc	cos	=	K	
90	2	RADR	GODP	LBL	W	RADR	=	K	9	CD	LBL W
4 00	PF	R	SPC	2	STOP	=	K	4	PRNT	Ø	
10	SPC	2	CD	STOP	=	K	5	PRNT	EXC	IFFG	
20	K	Ø	-	3	)	IF=Ø	P	H	I	SPC	
30	2	CD	STOP	=	K	6	PRNT	EXC	IFFG	CONT	
40	PF	=	K	9	GODP	LBL	F	K	1	=	LBL F
50	Rxxx	Ø	Ø	1	K	2	=	Rxx	Ø	2	
60	K	3	=	Rxx	Ø	3	RADR	GODP	LBL	ENDR	LBL ENDR
70	PF	E	N	D	CD	PF	PF	RSET			
80											
90											
	0	1	2	3	4	5	6	7	8	9	

Replace π STOP with EXC D/R for automatic repetitive operation.

NOTES

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

$K_{\emptyset}$: working
K_1 : r_1 , r_2 , r'
K_2 : θ_1 , θ_2 , θ'
K_3 : ϕ_1 , ϕ_2 , ϕ'
K_9 : return address

$R_{\emptyset\emptyset\emptyset}$: indirect address
$R_{\emptyset 1}$: x_1 , x'
$R_{\emptyset 2}$: y_1 , y'
$R_{\emptyset 3}$: z_1 , z'
$R_{\emptyset 4}$: x_2
$R_{\emptyset 5}$: y_2
$R_{\emptyset 6}$: z_2

Labeled Subroutines:
(G) , (H) , (I)
(J) , (K) , (P)
(Q) , × , IFFG , ENDR

PROGRAM DESCRIPTION

Given two 3-dimensional vectors $\overline{V}_1$ and $\overline{V}_2$, this program calculates their vector or cross-product $\overline{V}'$ which is another vector that may be defined as

$$\overline{V}' = \overline{V}_1 \times \overline{V}_2 = \begin{bmatrix} \overline{i} & \overline{j} & \overline{k} \\ x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \end{bmatrix} = x'\overline{i} + y'\overline{j} + z'\overline{k} \ ,$$

or $\overline{V}' = (y_1z_2 - z_1y_2 \ , \ z_1x_2 - x_1z_2 \ , \ x_1y_2 - y_1x_2)$

where $\overline{i}$, $\overline{j}$, and $\overline{k}$ are the mutually orthogonal unit vectors taken in the positive directions of the xyz coordinate system axes.

This program solves for x' , y' , and z' . It is arranged for convenient chain multiplication of a series of vectors, with automatic printout of the input data and resultant cross product vectors.

Thus given $\overline{V}_1 \times \overline{V}_2 \times \overline{V}_3 \ . \ . \ . \times \overline{V}_n \equiv \overline{V}'$

this program finds $\overline{V}_1 \times \overline{V}_2 = \overline{V}' \times \overline{V}_3 = \overline{V}' \times \overline{V}_4 = \overline{V}' \ . \ . \ . = \overline{V}' \times \overline{V}_n = \overline{V}'$

The vectors to be multiplied may be entered in either rectangular coordinates as $(x_1 \ , \ y_1 \ , \ z_1) \times (x_2 \ , \ y_2 \ , \ z_2) \equiv (x' \ , \ y' \ , \ z')$, or in spherical coordinates as $(r_1 \ , \ \theta_1 \ , \ \phi_1) \times (r_2 \ , \ \theta_2 \ , \ \phi_2) \equiv (r' \ , \ \theta' \ , \ \phi')$. Angle measure for spherical coordinates may be either degrees or radians. When vectors are entered in spherical coordinates, the program converts the data to rectangular coordinates before performing the multiplication, and then re-converts the resulting (x' , y' , z') to (r' , θ' , φ'). The x' , y' , z' values may be manually recalled from $R_{\emptyset 1}$, $R_{\emptyset 2}$, and $R_{\emptyset 3}$.

With steps 098 and 207 changed from CONT to STOP either $\overline{V}_1$ and/or $\overline{V}_2$ may be entered in xyz or r$\theta\phi$ coordinates. That is, $\overline{V}_1$ may be entered as (x_1 , y_1 , z_1) and $\overline{V}_2$ as $(r_2 , \theta_2 , \phi_2)$ with the solution given as r' , θ' , ϕ' ; or $\overline{V}_1$ may be entered as $(r_1 , \theta_1 , \phi_1)$ and $\overline{V}_2$ as (x_1 , y_1 , z_1) with the solution given by (x' , y' , z').

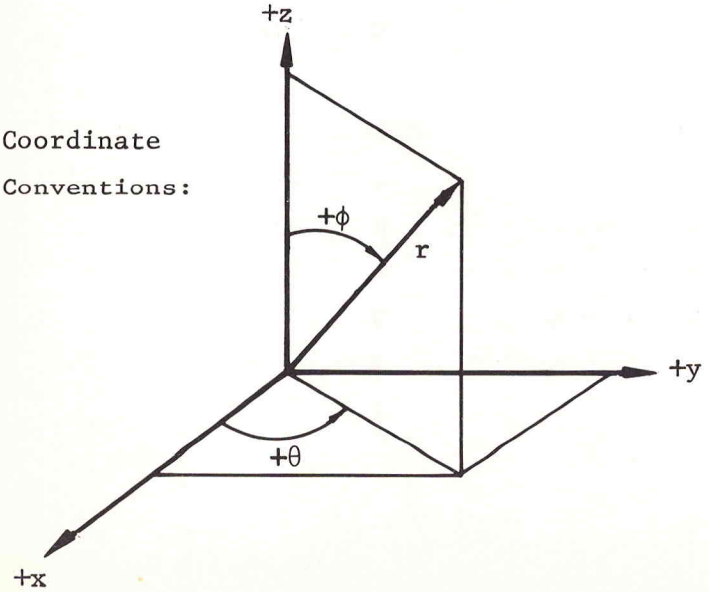
Program execution is modified as follows:

1. Press START; STFG for DEG; CLFG for RAD
2. For $\overline{V}_1$ as (x_1 , y_1 , z_1): EXC (G) .
3. Enter x_1 , CONT ; y_1 , CONT ; z_1 , CONT ; program will STOP ;
4. For $\overline{V}_2$ as $(r_2 , \theta_2 , \phi_2)$: EXC (K) .
5. Enter r_2 , CONT ; θ_2 , CONT ; ϕ_2 , CONT ;
6. Results will print out as (r' , θ' , ϕ') .

or

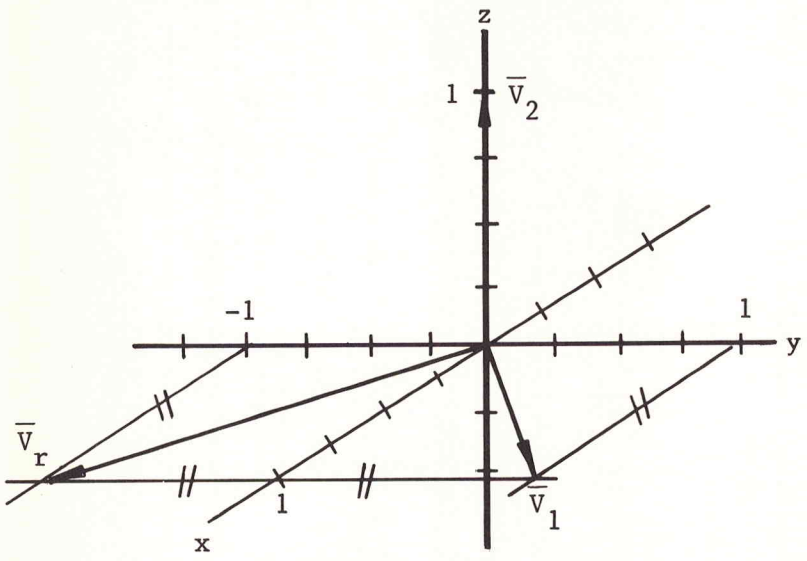
1. Press START; STFG for DEG; CLFG for RAD
2. For $\overline{V}_1$ as $(r_1 , \theta_1 , \phi_1)$: EXC (H) .
3. Enter r_1 , CONT ; θ_1 , CONT ; ϕ_1 , CONT ; program will STOP ;
4. For $\overline{V}_2$ as (x_2 , y_2 , z_2): EXC (J) .
5. Enter x_2 , CONT ; y_2 , CONT ; z_2 , CONT ;
6. Results will print out as (x' , y' , z') .

Coordinate Conventions:



NOTES

EXAMPLES



Given $\overline{V}_1$ and $\overline{V}_2$:

1. In rectangular coordinates

$\overline{V}_1 \times \overline{V}_2 = (1, 1, 0) \times (0, 0, 1) =$

$\overline{V}_r = (x' , y' , z') = (1, -1, 0)$

2. In spherical coordinates

$\overline{V}_1 \times \overline{V}_2 = (\sqrt{2}, 45^\circ, 90^\circ) \times (1, 0^\circ, 0^\circ) =$

$\overline{V}_r = (r' , \theta' , \phi') = (\sqrt{2}, -45^\circ, 90^\circ)$

Chained operations

$\overline{V}_1 \times \overline{V}_2) \times \overline{V}_3) \times \overline{V}_4 = \overline{V}_r$

$(1, 1, 0) \times (0, 0, 1)] \times (1.5, .5, .5)] \times$

$(-1, .5, 1)] = \overline{V}_r = (-1.5, -1.5, -.75)$

```
VEC X-PROD

DEG: STFG
RAD: CLFG

RECT: EXC G
SPHR: EXC H

#G

X 1
          1.
Y 1
          1.
Z 1
          0.

*

X 2
          0.
Y 2
          0.
Z 2
          1.

=

X'
          1.
Y'
          1.
Z'
          0.

#H

R 1
  1.414213562
Θ 1
         45.
DEG
PHI 1
         90.
DEG

*

R 2
          1.
Θ 2
          0.
DEG
PHI 2
          0.
DEG

=

R'
  1.414213562
Θ'
        -45.
DEG
PHI'
         90.
DEG

END
```

```
CHAINED
   OPERATIONS:

#G

X 1
          1.
Y 1
          1.
Z 1
          0.

*

X 2
          0.
Y 2
          0.
Z 2
          1.

=

X'
          1.
Y'
         -1.
Z'
          0.

*

X 2
          1.5
Y 2
          .5
Z 2
          .5

=

X'
         -.5
Y'
         -.5
Z'
          2.

*

X 2
         -1.
Y 2
          .5
Z 2
          1.

=

X'
         -1.5
Y'
         -1.5
Z'
         -.75

END
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1		EXC ×	0	TITLE
				FLAG Instruction
2	For DEG	SET FLAG	0	EXC Instruction
	For RAD	CLR FLAG	0	#G #H
				x_1 r_1
3	For rect. coordinates	EXC (G)	0	y_1 θ_1
	For sphr. coordinates	EXC (H)		z_1 ϕ_1
4	Enter x_1 r_1	CONT	0	*
	y_1 θ_1	CONT	0	x_2 r_2
	z_1 ϕ_1	CONT	0	y_2 θ_2
	or			
	x_2 r_2	CONT	0	z_2 ϕ_2
	y_2 θ_2	CONT	0	=
	z_2 ϕ_2	CONT	π	x' in $R_{\emptyset 1}$
				y' in $R_{\emptyset 2}$
5	To multiply this result			z' in $R_{\emptyset 3}$
	by $\overline{V}_3$;			or
	for rect. coordinates	EXC (J)		r' in K_1
	for sphr. coordinates	EXC (K)		θ' in K_2
	x_3 r_3	CONT	0	φ' in K_3
	y_3 or θ_3	CONT	0	
	z_3 ϕ_3	CONT	π	
	Step 5 may be repeated for as many vectors as desired.			
6	To enter a new $\overline{V}_1$ to begin a new operation, return to Step 3.			
7	To terminate the program	EXC ENDR		

If the π STOP steps are replaced as described on the step list, the program will automatically return to Step 3 following printout of the results in Step 4.

If vectors in rect. and spher. coordinates are to be multiplied together, see special instructions in the Program Description.

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
0	LBL	×	PF	PF	V	E	C	SPC	X	−	LBL ×
10	P	R	O	D	CLR	PF	D	E	G	:	
20	SPC	S	T	F	G	PF	R	A	D	:	
30	SPC	C	L	F	G	PF	PF	R	E	C	
40	T	:	SPC	E	X	C	SPC	G	PF	S	
50	P	H	R	:	SPC	E	X	C	SPC	H	
60	RSET	LBL	G	PF	#	G	CLR	PF	X	SPC	LBL G
70	1	STOP	=	Rxxx	Ø	Ø	1	PRNT	Y	SPC	
80	1	CD	STOP	=	Rxx	Ø	2	PRNT	Z	SPC	
90	2	CD	STOP	=	Rxx	Ø	3	PRNT	CONT$	LBL	LBL J
1 00	J	PF	*	CD	PF	X	SPC	2	STOP	=	
10	Rxx	Ø	4	PRNT	Y	SPC	2	CD	STOP	=	
20	Rxx	Ø	5	PRNT	Z	SPC	2	CD	STOP	=	
30	Rxx	Ø	6	PRNT	PF	=	EXC	I	PF	Rxx	
40	Ø	1	X	'	PRNT	Rxx	Ø	2	Y	'	
50	PRNT	Rxx	Ø	3	Z	'	PRNT	π#	STOP#	LBL	LBL H
60	H	PF	#	H	CLR	1	=	Rxxx	Ø	Ø	
70	Ø	PF	R	SPC	1	CD	STOP	=	K	1	
80	PRNT	Ø	SPC	1	CD	STOP	=	K	2	PRNT	
90	EXC	IFFG	P	H	I	SPC	1	CD	STOP	=	
2 00	K	3	PRNT	EXC	IFFG	EXC	P	CONT$	LBL	K	LBL K
10	PF	*	CLR	4	=	Rxx	Ø	Ø	PF	R	
20	SPC	2	CD	STOP	=	K	1	PRNT	Ø	SPC	
30	2	CD	STOP	=	K	2	PRNT	EXC	IFFG	P	
40	H	I	SPC	2	CD	STOP	=	K	3	PRNT	
	0	1	2	3	4	5	6	7	8	9	

Replace π STOP with EXC G for repetitive operation.

$ Replace two CONT's with STOP's for intermixing of rectangular and spherical vectors. See Program Description.

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
50	EXC	IFFG	EXC	(P)	PF	(=)	EXC	(I)	PF	K	
60	1	$\sqrt{\Sigma x^2}$	K	2	=	K	Ø	K	2	10^{oo}	
70	=	10^{oo}	Ø	Ø	e^x	int	x^a	×	2	−	
80	1	=	×	K	Ø	x^a	×	(	K	1	
90	÷	K	Ø	)	arc	cos	=	K	2	K	
3 00	Ø	$\sqrt{\Sigma x^2}$	K	3	=	K	1	(R)	(')	PRNT	
10	x^a	×	(	K	3	÷	K	1	)	arc	
20	cos	=	K	3	CLR	IFFG	D/R	CONT	K	2	
30	(Ø)	(')	PRNT	EXC	IFFG	K	3	(P)	(H)	(I)	
40	(')	PRNT	EXC	IFFG	π*	STOP*	LBL	(P)	RADR	=	LBL (P)
50	K	9	CLR	IFFG	D/R	CONT	K	1	×	K	
60	2	cos	×	K	3	sin	=	EXC	(Q)	K	
70	1	×	K	2	sin	×	K	3	sin	=	
80	EXC	(Q)	K	1	×	K	3	cos	=	EXC	
90	(Q)	K	9	GODP	LBL	(Q)	Rxx	Rxx	Ø	Ø	LBL (Q)
4 00	Rxx	Ø	Ø	+	1	=	Rxx	Ø	Ø	RADR	
10	GODP	LBL	(I)	Rxx	Ø	6	×	Rxx	Ø	2	LBL (I)
20	−	Rxx	Ø	3	×	Rxx	Ø	5	=	K	
30	1	Rxx	Ø	3	×	Rxx	Ø	4	−	Rxx	
40	Ø	1	×	Rxx	Ø	6	=	K	2	Rxx	
50	Ø	1	×	Rxx	Ø	5	−	Rxx	Ø	2	
60	×	Rxx	Ø	4	=	K	3	=	Rxx	Ø	
70	3	K	1	=	Rxx	Ø	1	K	2	=	
80	Rxx	Ø	2	RADR	GODP	LBL	IFFG	CD	IFFG	(D)	LBL IFFG
90	(E)	(G)	π	CONT	IF=Ø	(R)	(A)	(D)	CONT	RADR	
	0	1	2	3	4	5	6	7	8	9	

* Replace π STOP with EXC (H) for repetitive operation.

PROGRAM STEPS

		0	1	2	3	4	5	6	7	8	9	COMMENTS
5	00	GODP	LBL	ENDR	PF	E	N	D	CD	PF	PF	LBL ENDR
	10	RSET										
	20											
	30											
	40											
	50											
	60											
	70											
	80											
	90											
___	00											
	10											
	20											
	30											
	40											
	50											
	60											
	70											
	80											
	90											
___	00											
	10											
	20											
	30											
	40											
		0	1	2	3	4	5	6	7	8	9	

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

$K_{\emptyset}$: α
K_1 : x, r
K_2 : y, θ
K_3 : x', r'
K_4 : y', θ'
K_5 : x_0 or s
K_6 : y_0 or γ
K_7 : working
K_8 : working

Subroutine Labels:

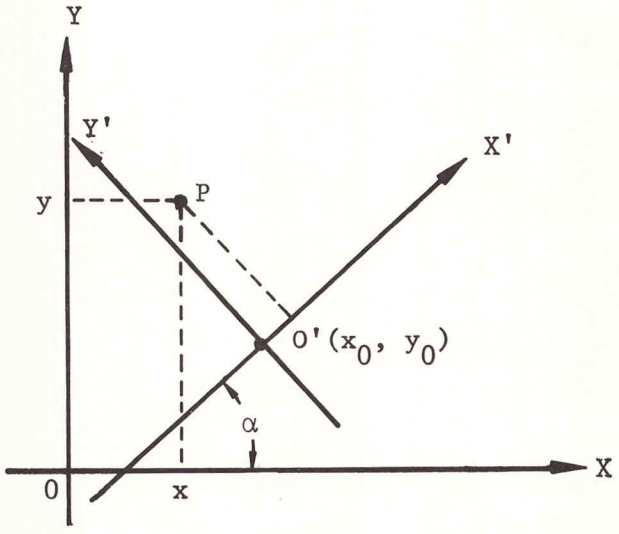
(M) , (N) , (O)

arc, hyp, IFFG, SFG, ENDR

PROGRAM DESCRIPTION

This program finds the new coordinates of a point in a translated and/or rotated system of coordinate axes, in either rectangular or polar notation.

Rectangular Coordinates:



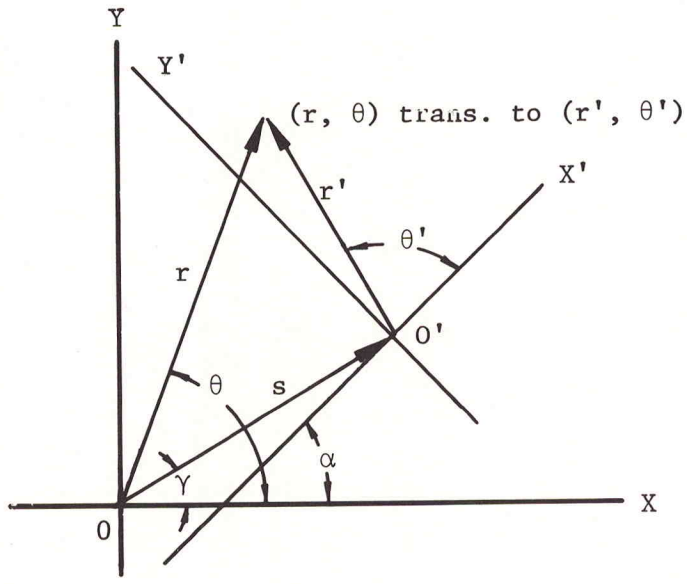
Given the rectangular xy coordinate system which is to be transformed to a new x'y' coordinate system through translation of the origin 0 to 0' by distances x_0 and y_0 relative to 0, and/or rotation of the xy axes by angle α, the program calculates the coordinates (x', y') in the new system for any point P(x, y) in the old system; where

$$x' = (x - x_0) \cos \alpha + (y - y_0) \sin \alpha \ ;$$

$$y' = (y - y_0) \cos \alpha - (x - x_0) \sin \alpha \ .$$

Polar Coordinates:

The program also performs the analogous rotation and translation of axes for data entered in polar notation (r, θ),

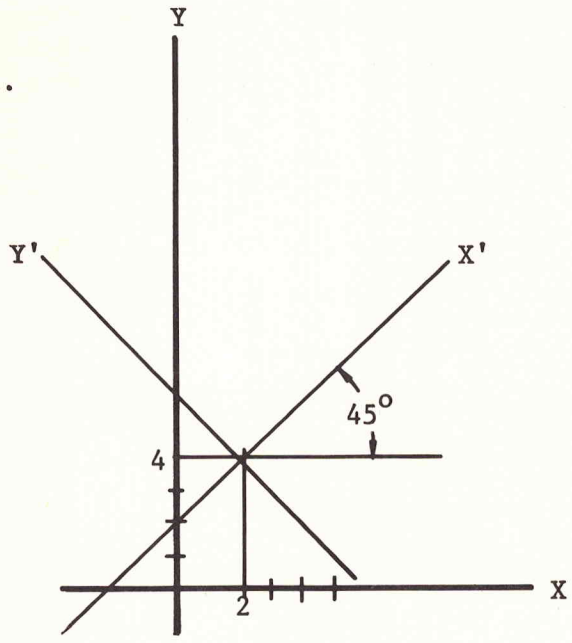


where 0' is the new origin translated from the old origin 0 by the vector s/γ, and r'/θ' are the polar coordinates of the point P relative to the new system, and r/θ are the polar coordinates of P relative to the old system.

NOTES

EXAMPLES

1.



x	y
3	2
7	10
6	5
-2	4

Translate and rotate the XY system by $x_0 = 2$, $y_0 = 4$, and $\alpha = 45^\circ$, and calculate the new coordinates for the points listed above.

```
RECT COORDINATE
AXES, TRANS/ROT:

SET FLAG FOR DEG
CLR FLAG FOR RAD

X 0
              2.
Y 0
              4.
ALPHA
             45.
DEG

#ARC

X
              3.
Y
              2.

TO

X':K3
-.7071067812
Y':K4
-2.121320344

#ARC

X
              7.
Y
             10.

TO

X':K3
 7.778174593
Y':K4
 .7071067812

#ARC

X
              6.
Y
              5.

TO

X':K3
 3.535533906
Y':K4
-2.121320344

#ARC

X
-             2.
Y
              4.

TO

X':K3
-2.828427125
Y':K4
 2.828427125

END
```

2. Translate and rotate the polar coordinate by $s = 7$, $\gamma = 35^\circ$, and $\alpha = 45^\circ$, and find the new r', θ' coordinates for these points:

r	θ
6	82°
5	135°
10	-29°

```
POLAR COORDINATE
AXES, TRANS/ROT:

SET FLAG FOR DEG
CLR FLAG FOR RAD

S
              7.
GAMMA
             35.
DEG
ALPHA
             45.
DEG

#HYP

R
              6.
θ
             82.
DEG

TO

R':K3
 5.264231925
θ':K4
 113.5324477
DEG

#HYP

R
              5.
θ
            135.
DEG

TO

R':K3
 9.281991836
θ':K4
 137.9611776
DEG

#HYP

R
             10.
θ
-            29.
DEG

TO

R':K3
 9.360984962
θ':K4
-116.2296848
DEG

END
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1	For RECT coordinates	EXC (M)	0	TITLE
2	For DEG	SET FLAG, CONT	0	FLAG
	For RAD	CLR FLAG, CONT	0	Instruction
3	Enter x_0	CONT	0	x_0
	y_0	CONT	0	y_0
	α	CONT	0	α
4	Enter x	CONT	0	x
	y	CONT	π	y
5	Now enter new point	EXC arc		x'
	and return to Step 4.			y'
6	To enter new initial conditions, return to			
	Step 1.			
7	To terminate	EXC ENDR		
1	For POLAR coordinates	EXC (N)	0	TITLE
2	For DEG	SET FLAG, CONT		FLAG
	For RAD	CLR FLAG, CONT		Instruction
3	Enter s	CONT	0	s
	γ	CONT	0	γ
	α	CONT	0	α
4	Enter r	CONT	0	r
	θ	CONT	π	θ
5	To enter a new point	EXC hyp		r'
	and return to Step 4.			θ'
6	To enter new initial conditions return to			
	Step 1.			
7	To terminate	EXC ENDR		
	If the π STOP steps are replaced as detailed			
	on the PROGRAM STEP list the program will			
	automatically return to Step 4 for entry of			
	new data points.			

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
0	LBL	M	PF	PF	R	E	C	T	SPC	C	LBL M
10	0	0	R	D	I	N	A	T	E	EXC	
20	0	EXC	SFG	PF	X	SPC	Ø	CD	STOP	=	
30	K	5	PRNT	Y	SPC	Ø	CD	STOP	=	K	
40	6	PRNT	A	L	P	H	A	CD	STOP	=	
50	K	Ø	PRNT	EXC	IFFG	LBL	arc	PF	#	A	LBL arc
60	R	C	CLR	IFFG	D/R	CONT	PF	X	STOP	=	
70	K	1	PRNT	Y	CD	STOP	=	K	2	PRNT	
80	PF	T	0	K	2	-	K	6	=	K	
90	8	K	1	-	K	5	=	K	7	×	
1 00	K	Ø	cos	+	K	8	×	K	Ø	sin	
10	=	K	3	PF	X	'	:	K	3	PRNT	
20	K	8	×	K	Ø	cos	-	K	7	×	
30	K	Ø	sin	=	K	4	Y	'	:	K	
40	4	PRNT	π#	STOP#	LBL	N	PF	PF	P	0	LBL N
50	L	A	R	SPC	C	O	O	R	D	I	
60	N	A	T	E	EXC	0	EXC	SFG	PF	S	
70	CD	STOP	=	K	5	PRNT	G	A	M	M	
80	A	CD	STOP	=	K	6	PRNT	EXC	IFFG	A	
90	L	P	H	A	CD	STOP	=	K	Ø	PRNT	
2 00	EXC	IFFG	LBL	hyp	PF	#	H	Y	P	CLR	LBL hyp
10	IFFG	D/R	CONT	PF	R	STOP	=	K	1	PRNT	
20	Ø	CD	STOP	=	K	2	PRNT	EXC	IFFG	PF	
30	T	0	K	1	×	K	2	cos	-	K	
40	5	×	K	6	cos	=	K	3	K	1	

Replace π STOP with EXC arc for automatic repetitive operation.

	0	1	2	3	4	5	6	7	8	9	COMMENTS
	50	×	K	2	sin	-	K	5	×	K	6	
	60	sin	=	K	4	×	K	Ø	sin	+	K	
	70	3	×	K	Ø	cos	=	K	7	K	4	
	80	×	K	Ø	cos	-	K	3	×	K	Ø	
	90	sin	=	K	8	√Σx2	K	7	=	K	3	
3	00	PF	R	'	:	K	3	PRNT	K	8	10^{oo}	
	10	=	10^{oo}	Ø	Ø	e^x	int	x^a	×	2	-	
	20	1	=	×	K	3	x^a	×	(	K	7	
	30	÷	K	3	)	arc	cos	=	K	4	CLR	
	40	IFFG	D/R	CONT	K	4	Ø	'	:	K	4	
	50	PRNT	EXC	IFFG	π*	STOP*	LBL	IFFG	CD	IFFG	D	LBL IFFG
	60	E	G	π	CONT	IF=Ø	R	A	D	CONT	RADR	
	70	GODP	LBL	0	A	X	E	S	,	SPC	T	LBL 0
	80	R	A	N	S	/	R	O	T	:	RADR	
	90	GODP	LBL	SFG	PF	S	E	T	SPC	F	L	LBL SFG
4	00	A	G	SPC	F	O	R	SPC	D	E	G	
	10	C	L	R	SPC	F	L	A	G	SPC	F	
	20	O	R	SPC	R	A	D	CD	STOP	RADR	GODP	
	30	LBL	ENDR	PF	E	N	D	CD	PF	PF	RSET	LBL ENDR
	40											
	50											
	60											
	70											
	80											
	90											
		0	1	2	3	4	5	6	7	8	9	

* Replace π STOP with EXC hyp for automatic repetitive operation.

NOTES

Coordinate Transforms: Rectangular to Polar or Spherical, and Inverse

TITLE

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

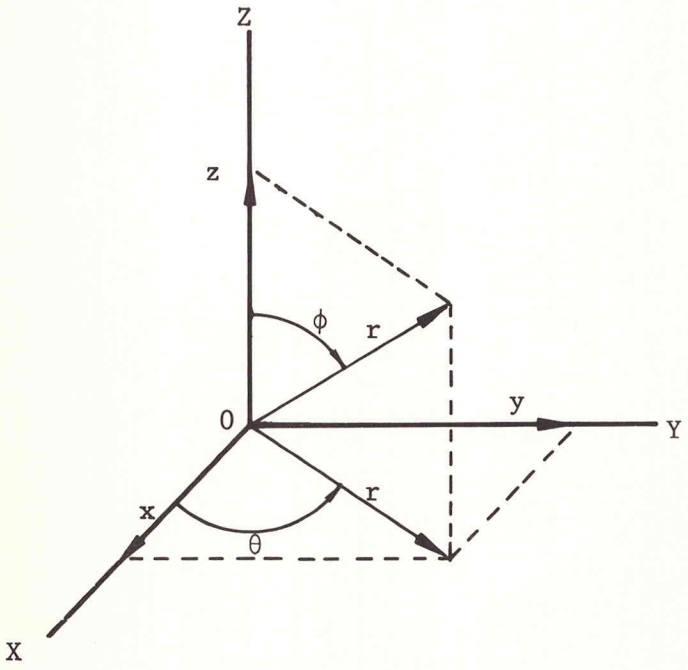
$K_{\emptyset}$: working
K_1 : x
K_2 : y
K_3 : z
K_4 : r
K_5 : θ
K_6 : ϕ
K_9 : dimension

Subroutine Labels:

(S) (T) (U) Π_4,

Δ_3, cos, IFFG, SFG, ENDR

PROGRAM DESCRIPTION



Given the right-handed 2-dimensional coordinate system defined by the XY axes, or the 3-dimensional system defined by the XYZ axes, as sketched above, this program performs the rectangular to polar transform (x, y) → (r, θ) or the rectangular to spherical transform (x, y, z) → (r, θ, ϕ). The program also performs the inverse transforms (r, θ) → (x, y) and (r, θ, ϕ) to (x, y, z).

EXAMPLES

1. RECT → POLAR

   Convert these (x, y) points to their equivalent (r, θ) coordinates:

   x = 3 , y = 4 ; → deg

   4 , -2 ; → rad

2. RECT → SPHER

   Convert these (x, y, z) points to their equivalent (r, θ, ϕ) coordinates:

   x = 3 , y = 8 , z = 2 ; → deg

   7 , 3 , -2 ; → rad

3. POLAR → RECT

   Convert these (r, θ) to the equivalent (x, y):

   r = 6 , θ = 35°

   4 , 1.5 rad

4. SPHER → RECT

   Convert these (r, θ, ϕ) to the equivalent (x, y, z):

   r = 6 , θ = 35° , ϕ = 20°

   7 , -1 rad, 1.5 rad

```
COOR TRANSFORMS:
2-DIM X,Y TO R,θ
OR 3-DIM X,Y,Z
TO R,θ,PHI ?
                2.
SET FLAG FOR DEG
CLR FLAG FOR RAD
#
X
                3.
Y
                4.
TO
R
                5.
θ
 53.13010235
DEG
#
X
                4.
Y
-               2.
TO
R
 4.472135955
θ
-26.56505118
DEG
#
X
                3.
Y
                8.
Z
                2.
TO
R
 8.774964387
θ
 69.44395478
DEG
PHI
 76.82528784
DEG
#
X
                7.
Y
                3.
Z
-               2.
TO
R
 7.874007874
θ
 .4048917863
RAD
PHI
 1.827610244
RAD
END
```

```
COOR TRANSFORMS:
2-DIM R,θ TO X,Y
OR 3-DIM R,θ,PHI
TO X,Y,Z ?
                2.
SET FLAG FOR DEG
CLR FLAG FOR RAD
#
R
                6.
θ
               35.
DEG
TO
X
 4.914912266
Y
 3.441458618
#
R
                4.
θ
              1.5
RAD
TO
X
 .2829488067
Y
 3.989979946
#
R
                6.
θ
               35.
DEG
PHI
               20.
DEG
TO
X
 1.680998998
Y
 1.177048!7
Z
 5.638155725
#
R
                7.
θ
-               1.
RAD
PHI
              1.5
RAD
TO
X
 3.77264189
Y
-5.875541621
Z
 .4951604117
END
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1	For Rect. to Polar or			
	Spherical	EXC (S)	0	TITLE
2	Enter dimension:	2 or 3, CONT	0	2 or 3
3	For DEG	SET FLAG CONT	0	INSTRUCTION
	or for RAD	CLEAR FLAG CONT		
4	Enter x	CONT	0	x
	y	CONT	0	y
	z (3-dim only)	CONT	π	z
				TO
5	To convert a new			r
	vector with the same			θ
	dimensions	EXC Π_4	π	ϕ
6	To terminate	EXC ENDR		
1	For Polar or Spherical			
	to Rect.	EXC (T)	0	TITLE
2	Enter dimension:	2 or 3, CONT	0	2 or 3
3	For DEG	SET FLAG CONT	0	INSTRUCTION
	or for RAD	CLEAR FLAG CONT	0	
4	Enter r	CONT	0	r
	θ	CONT	0	θ
	ϕ (3-dim only)	CONT	π	ϕ
				TO
5	To convert a new			x
	vector with the same			y
	dimensions	EXC Δ_3	π	z
6	To terminate	EXC ENDR		

In either program, dimension may be changed from 2 to 3 or 3 to 2 without reprinting title: CD, then 2 = K_9, or 3 = K_9.

The FLAG may be set or cleared as desired prior to any initial data entry (x or r).

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
0	LBL	S	EXC	U	2	-	D	I	M	SPC	LBL S
10	X	,	Y	SPC	T	O	SPC	R	,	Ø	
20	O	R	SPC	3	-	D	I	M	SPC	X	
30	,	Y	,	Z	PF	T	O	SPC	R	,	
40	Ø	,	P	H	I	SPC	?	CLR	STOP	=	
50	K	9	PRNT	EXC	SFG	LBL	Π_4	PF	#	CD	LBL Π_4
60	PF	X	STOP	=	K	1	PRNT	Y	CD	STOP	
70	=	K	2	PRNT	$\sqrt{\Sigma x^2}$	K	1	=	K	Ø	
80	CLR	IFFG	D/R	CONT	K	9	-	3	)	IF=Ø	
90	Z	CD	STOP	=	K	3	PRNT	PF	T	O	
1 00	$\sqrt{\Sigma x^2}$	K	Ø	=	K	4	PF	R	PRNT	x^a	
10	×	(	K	3	÷	K	4	)	arc	cos	
20	=	K	6	EXC	cos	EXC	IFFG	P	H	I	
30	K	6	PRNT	EXC	IFFG	π#	STOP#	CONT	PF	T	
40	O	K	Ø	=	K	4	PF	R	PRNT	EXC	
50	cos	EXC	IFFG	π#	STOP#	LBL	T	EXC	U	2	LBL T
60	-	D	I	M	SPC	R	,	Ø	SPC	T	
70	O	SPC	X	,	Y	O	R	SPC	3	-	
80	D	I	M	SPC	R	,	Ø	,	P	H	
90	I	T	O	SPC	X	,	Y	,	Z	SPC	
2 00	?	CLR	STOP	=	K	9	PRNT	EXC	SFG	LBL	LBL Δ_3
10	Δ_3	PF	#	CD	PF	R	STOP	=	K	4	
20	PRNT	Ø	CD	STOP	=	K	5	PRNT	EXC	IFFG	
30	CLR	IFFG	D/R	CONT	K	4	×	K	5	cos	
40	=	K	1	K	4	×	K	5	sin	=	

Replace π STOP with EXC Π_4 for repetitive operation.

	0	1	2	3	4	5	6	7	8	9	COMMENTS
50	K	2	K	9	-	3	)	IF=Ø	P	H	
60	I	CD	STOP	=	K	6	PRNT	sin	×	EXC	
70	IFFG	PF	T	O	K	1	=	K	1	PF	
80	X	PRNT	K	2	×	K	6	sin	=	K	
90	2	Y	PRNT	K	4	×	K	6	cos	=	
3 00	K	3	Z	PRNT	π*	STOP*	CONT	PF	T	O	
10	K	1	PF	X	PRNT	K	2	Y	PRNT	π*	
20	STOP*	LBL	IFFG	CD	IFFG	D	E	G	π	CONT	LBL IFFG
30	IF=Ø	R	A	D	CONT	RADR	GODP	LBL	cos	K	LBL cos
40	2	10^{oo}	=	10^{oo}	Ø	Ø	e^x	int	x^a	×	
50	2	-	1	=	×	K	Ø	x^a	×	(	
60	K	1	÷	K	Ø	)	arc	cos	=	K	
70	5	CLR	IFFG	D/R	CONT	Ø	K	5	PRNT	RADR	
80	GODP	LBL	SFG	PF	S	E	T	SPC	F	L	LBL SFG
90	A	G	SPC	F	O	R	SPC	D	E	G	
4 00	C	L	R	SPC	F	L	A	G	SPC	F	
10	O	R	SPC	R	A	D	CD	STOP	RADR	GODP	
20	LBL	U	PF	PF	C	O	O	R	SPC	T	LBL U
30	R	A	N	S	F	O	R	M	S	:	
40	PF	RADR	GODP	LBL	ENDR	PF	E	N	D	CD	LBL ENDR
50	PF	PF	RSET								
60											
70	* Replace π STOP with EXC Δ_3 for repetitive operation.										
80											
90											

NOTES

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

$K_\emptyset$: data pair counter
K_1 : u_i
K_2 : v_i
K_3 : $\sqrt{\Sigma u^2} = |\overline{U}|$
K_4 : $\sqrt{\Sigma v^2} = |\overline{V}|$
K_5 : $\overline{U} \cdot \overline{V}$
K_6 : θ
K_9 : n

Subroutine Labels:

(A) (B) (R)

ENDR

PROGRAM DESCRIPTION

This program calculates the inner or scalar (dot) product of two n-dimensional vectors $\overline{U}$ and $\overline{V}$ where

$$\overline{U} = (u_1 , u_2 , u_3 , \ldots , u_4) \text{ and}$$

$$\overline{V} = (v_1 , v_2 , v_3 , \ldots , v_4)$$

The inner product is given by

$$\overline{U} \cdot \overline{V} = u_1v_1 + u_2v_2 + \ldots + u_nv_n = \Sigma u_iv_i$$

The absolute magnitudes or norms for $\overline{U}$ and $\overline{V}$ are given by

$$|\overline{U}| = \overline{U} \cdot \overline{U} = [\Sigma(u_i)^2]^{\frac{1}{2}} \text{ and}$$

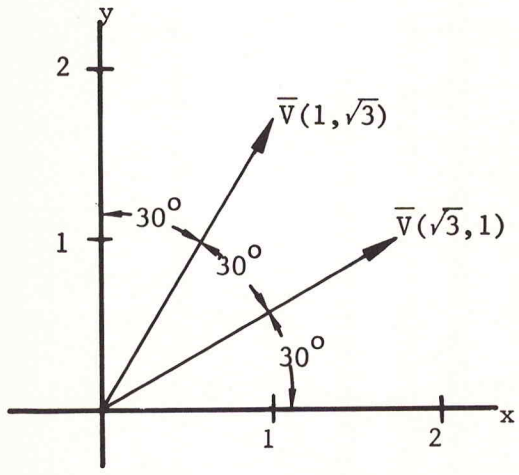
$$|\overline{V}| = \overline{V} \cdot \overline{V} = [\Sigma(v_i)^2]^{\frac{1}{2}} .$$

The angle between $\overline{U}$ and $\overline{V}$ is given by

$$\theta = \measuredangle \overline{U} \cdot \overline{V} = \arccos \frac{\overline{U} \cdot \overline{V}}{|\overline{U}| \ |\overline{V}|} .$$

The program operates in degrees or radians, at the user's option.

Example 1:



$n = 2$

$\overline{U} \cdot \overline{V} = 3.464101615$

$|\overline{U}| = 2$

$|\overline{V}| = 2$

$\theta = 30^{\circ}$

$= 0.5235987756$ rad

```
2 N-DIM VECTORS:
INNER/SCALAR/DOT
PROD; ABS MAG; &
INC ANGLE:

DEG: STFG
RAD: CLFG

#A

DIM N?
             2.

ENTER U,V PAIRS:

 1.732050808
             1.

             1.
 1.732050808

RESULTS:

DIM N:K0
             2.

INNER PRODUCT:K5
 3.464101615

ABS MAG
 U:K3
             2.
 V:K4
             2.

INC ANGLE:K6
            30.
DEG

END
```

```
2 N-DIM VECTORS:
INNER/SCALAR/DOT
PROD; ABS MAG; &
INC ANGLE:

DEG: STFG
RAD: CLFG

#A

DIM N?
             0.

ENTER U,V PAIRS:

            12.
             0.

             3.
            10.

             2.
             2.

             5.
             6.

-            2.
-            2.


RESULTS:

DIM N:K0
             5.

INNER PRODUCT:K5
            68.

ABS MAG
 U:K3
  13.6381817
 V:K4
            12.

INC ANGLE:K6
 1.142303721
RAD


END
```

Example 2:

$\overline{U} = (u_1 , u_2 , \ldots , u_5)$

$\overline{V} = (v_1 , v_2 , \ldots , v_5)$

$n = 5$

u_1 , v_1 : 12, 0

u_2 , v_2 : 3, 10

u_3 , v_3 : 2, 2

u_4 , v_4 : 5, 6

u_5 , v_5 : -2, -2

$K_{\emptyset}$: Dimension n = 5

K_5 : Inner Product $\overline{U} \cdot \overline{V} = 68$

K_3 : $|\overline{U}| = 13.6381817$

K_4 : $|\overline{V}| = 12$

K_6 : Included Angle θ =
1.142303721 rad

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1		EXC	0	TITLE
				FLAG Instruction
2	For DEG	SET FLAG CONT	0	
	For RAD	CLEAR FLAG CONT	0	#A
3	Enter dim n*	CONT	0	n
4	Enter u_1	CONT	0	u_1
	v_1	CONT	0	v_1
	u_2	CONT	0	u_2
	v_2	CONT	0	v_2
	.			.
	.			.
	.			.
	u_n	CONT	0	u_n
	v_n	CONT	π#	v_n
				Results
5	To enter a new pair of vectors $\overline{J}$, $\overline{K}$ to return the program to Step 3.	EXC (A)		$K_\emptyset$: n K_5 : Inner Prod $\overline{U} \cdot \overline{V}$ K_3 : Abs Mag $\overline{U}$ K_4 : Abs Mag $\overline{V}$
6	To terminate the program	EXC ENDR		K_6 : $\overline{U} \cdot \overline{V}$
	*If 0 is entered for n, then press following entry of the last data pair u_n, v_n	EXC (R)		

#If the π# STOP# steps are replaced with EXC (A), the program will automatically return to Step 3 for entry of a new vector.

PROGRAM STEPS

		0	1	2	3	4	5	6	7	8	9	COMMENTS
0	00	LBL	•	PF	PF	2	SPC	N	−	D	I	LBL •
	10	M	SPC	V	E	C	T	O	R	S	:	
	20	I	N	N	E	R	/	S	C	A	L	
	30	A	R	/	D	O	T	P	R	O	D	
	40	;	SPC	A	B	S	SPC	M	A	G	;	
	50	SPC	&	I	N	C	SPC	A	N	G	L	
	60	E	:	CLR	PF	D	E	G	:	SPC	S	
	70	T	F	G	PF	R	A	D	:	SPC	C	
	80	L	F	G	STOP	LBL	A	PF	#	A	CLR	LBL A
	90	=	K	Ø	=	K	3	=	K	4	=	
1	00	K	5	PF	D	I	M	SPC	N	?	STOP	
	10	=	K	9	PRNT	PF	E	N	T	E	R	
	20	SPC	U	,	V	SPC	P	A	I	R	S	
	30	:	LBL	B	PF	CLR	IFFG	D/R	CONT	STOP	=	LBL B
	40	K	1	PRNT	$\sqrt{\Sigma x^2}$	K	3	=	K	3	CD	
	50	STOP	=	K	2	PRNT	$\sqrt{\Sigma x^2}$	K	4	=	K	
	60	4	K	1	×	K	2	+	K	5	=	
	70	K	5	CD	1	Σ_0	K	Ø	−	K	9	
	80	)	IF=Ø	EXC	R	CONT	EXC	B	LBL	R	PF	LBL R
	90	R	E	S	U	L	T	S	:	K	Ø	
2	00	PF	D	I	M	SPC	N	:	K	Ø	PRNT	
	10	PF	K	5	I	N	N	E	R	SPC	P	
	20	R	O	D	U	C	T	:	K	5	PRNT	
	30	PF	A	B	S	SPC	M	A	G	K	3	
	40	SPC	U	:	K	3	PRNT	K	4	SPC	V	

	0	1	2	3	4	5	6	7	8	9	COMMENTS
50	:	K	4	PRNT	PF	×	K	3	=	K	
60	6	x^a	×	(	K	5	÷	K	6	)	
70	arc	cos	=	K	6	CLR	K	6	I	N	
80	C	SPC	A	N	G	L	E	:	K	6	
90	PRNT	CD	IFFG	D	E	G	π	CONT	IF=Ø	R	
3 00	A	D	CONT	PF	π#	STOP#	LBL	ENDR	PF	E	LBL ENDR
10	N	D	CD	PF	PF	RSET					
20											
30	#Replace π STOP with EXC A for repetitive										
40	operation.										
50											
60											
70											
80											
90											
00											
10											
20											
30											
40											
50											
60											
70											
80											
90											
	0	1	2	3	4	5	6	7	8	9	

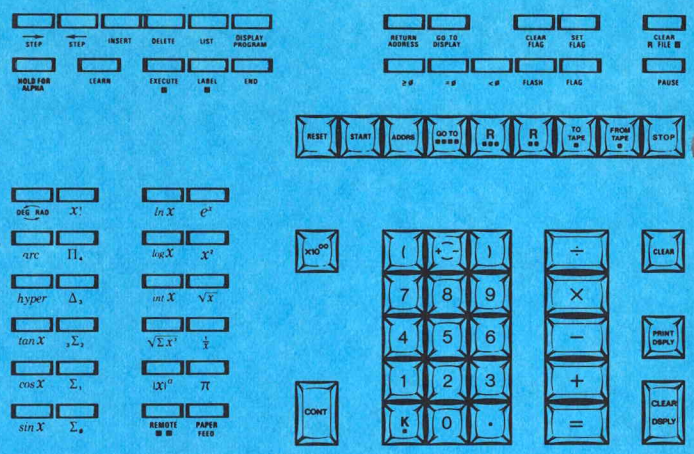
NOTES

SECTION 6 COMPLEX OPERATORS

OCTAL CODE	PRINT OUT	KEY	SYMBOL
001	LBL_	LABEL ■	LBL
002	FTP_	FROM TAPE ■	FTP
003	TTP_	TO TAPE ■	TTP
004	EXC_	EXECUTE ■	EXC
005	CFI_	CLEAR R FILE ■	CFI
007	GOTO	GO TO ■■■■	GT
010	R___	R ■■■	Rxxx
011	R__	R ■■	Rxx
040	CLDP	CLEAR DSPLY	CD
041	IFFL	FLASH	IFFL
042	S FG	SET FLAG	SFG
043	STOP	STOP	STOP
044	PRNT	PRINT DSPLY	PRNT
045	CLR	CLEAR	CLR
046	PAUS	PAUSE	PAUS
047	CLFG	CLEAR FLAG	CLFG
050	(	(	(
051	)	)	)
052	*	×	×
053	+	+	+
054	RSET	RESET	RSET
055	-	−	-
056	.	.	.
057	/	÷	÷
060	0	0	ø
061	1	1	1
062	2	2	2
063	3	3	3
064	4	4	4
065	5	5	5
066	6	6	6
067	7	7	7
070	8	8	8
071	9	9	9
072	ADR	ADDRS	ADR
073	STRT	START	STRT
074	IF<0	<ø	IF<ø
075	=	=	=
076	IF>=	≥ø	IF≥ø
077	IF=0	=ø	IF=ø

OCTAL CODE	PRINT OUT	KEY	SYMBOL
100	K_	K ●	K
101	DG/R	DEG RAD	D/R
102	ARC	*arc*	arc
103	HYP	*hyper*	hyp
104	TAN	$\tan x$	tan
105	COS	$\cos x$	cos
106	SIN	$\sin x$	sin
107	X!	$x!$	x!
110	PI4	Π_4	Π_4
111	DLT3	Δ_3	Δ_3
112	3SM2	${}_3\Sigma_2$	${}_3\Sigma_2$
113	SUM1	Σ_1	Σ_1
114	SUM0	$\Sigma_ø$	Σ_0
115	LN	$\ln x$	ln
116	LOG	$\log x$	log
117	INT	$int\, x$	int
120	RSSQ	$\sqrt{\Sigma x^2}$	$\sqrt{\Sigma x^2}$
121	X↑A	$\|x\|^a$	x^a
122	RM__	REMOTE ■■	RMT
123	E↑X	e^x	e^x
124	X↑2	x^2	x^2
125	SQRT	$\sqrt{x}$	$\sqrt{x}$
126	1/X	$\frac{1}{x}$	1/x
127	PI	π	π
130	PAPR	PAPER FEED	PF
131	*10↑	$\times 10^{oo}$	10^{oo}
132	CONT	CONT	CONT
133	R AD	RETURN ADDRESS	RADR
134	+/-	+/-	+/-
135	GODP	GO TO DISPLAY	GODP
136	IFFG	FLAG	IFFG
137	ENDR	END	ENDR

OCTAL CODE	PRINT OUT	KEY	SYMBOL
201		LABEL ■	LBL
202		FROM TAPE ■	FTP
203		TO TAPE ■	TTP
204		EXECUTE ■	EXC
205		CLEAR R FILE ■	CFI
207		BELL	BELL
210		SPACE ←	SPC←
211		TAB ←	TAB
220		STEP	STP←
221		INSERT	NSRT
222		DELETE	DLT
223		STEP →	STP→
224		LIST	LIST
225		DISPLAY PROGRAM	DPRG
226		LEARN	LRN
240		SPACE	SPC
241	!	!	!
242	"	"	"
243	#	#	#
244	$	$	$
245	%	%	%
246	&	&	&
247	'	'	'
250	(	(	(
251	)	)	)
252	*	*	*
253	+	+	+
254	,	,	,
255	-	-	-
256	.	.	.
257	/	/	/
260	0	0	Ø
261	1	1	1
262	2	2	2
263	3	3	3
264	4	4	4
265	5	5	5
266	6	6	6
267	7	7	7
270	8	8	8
271	9	9	9
272	:	:	:
273	;	;	;
274	<	<	<
275	=	=	=
276	>	>	>
277	?	?	?



OCTAL CODE	PRINT OUT	KEY	SYMBOL
300	@	@	@
301	A	A	A
302	B	B	B
303	C	C	C
304	D	D	D
305	E	E	E
306	F	F	F
307	G	G	G
310	H	H	H
311	I	I	I
312	J	J	J
313	K	K	K
314	L	L	L
315	M	M	M
316	N	N	N
317	O	O	O
320	P	P	P
321	Q	Q	Q
322	R	R	R
323	S	S	S
324	T	T	T
325	U	U	U
326	V	V	V
327	W	W	W
330	X	X	X
331	Y	Y	Y
332	Z	Z	Z
333	[	[	[
334	\	\	\
335	]	]	]
336	↑	↑	↑
337	_	_	_

TITLE

Complex Operators: Polar to Rectangular Conversion and Inverse

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

K_1 : x
K_2 : y
K_3 : r
K_4 : θ

Labeled Subroutines:
(S), arc, Rxx
(T), hyp, π
SFG, IFFG
10^{oo}, ENDR

PROGRAM DESCRIPTION

This program calculates either the rectangular to polar coordinate conversion

$$z = x + jy \rightarrow re^{j\theta} \equiv r, \theta$$

where $-\pi \leq \theta \leq \pi$,

or the inverse conversion from polar to rectangular coordinates

$$r, \theta \equiv re^{j\theta} \rightarrow x + jy = z$$

The conversion algorithms are written as independent labeled subroutines. The R → P conversion is LBL (S) ; the P → R is LBL (T)

These programs may be used with the other Complex Operator programs to solve programs of broad complexity. The P → R program stores its complex result x + jy in registers K_1 and K_2, where it is ready to serve as an operand for the Complex Operator programs. The R → P program converts x + jy in K_1 and K_2 to r and θ in K_3 and K_4.

Simple modifications to simplify the execution of these programs for repetitive operation are shown on the PROGRAM STEP pages.

Input data and results are printed out. In addition, each operator is printed out following the input data so that a complete record of the calculations is available for verification. When the printer is not used the results must be manually recalled from the indicated registers: x, y from K_1, K_2; r, θ from K_3, K_4.

EXAMPLES

1. RECTANGULAR to POLAR:

x + jy	r, θ in rad	or	r, θ in deg
0 + j0	0, 0		0, 0
3 + j4	5, 0.927295218		5, 53.13010235
3 - j4	5, -0.927295218		5, -53.13010235
-3 + j4	5, 2.214297436		5, 126.8698976
-3 - j4	5, -2.214297436		5, -126.8698976
-3 + j0	3, 3.141592654		3, +180°

Note that θ is always in the range $-\pi \leq \theta \leq \pi$ radians or $-180^{\circ} \leq \theta \leq 180^{\circ}$.

```
RECT TO POLAR

SET FLAG FOR DEG
CLR FLAG FOR RAD

#

X1
              3.
JY1
              4.

R TO P

R:K3
              5.
Θ:K4
   .927295218
RAD

#

X1
              3.
JY1
              4.

R TO P

R:K3
              5.
Θ:K4
  53.13010235
DEG

END
```

2. POLAR to RECTANGULAR:

r, θ	x + jy
0, 0	0 + j0
10, π/3	5 + j8.660254038
5, 60°	2.5 + j4.330127019
5, 420°	2.5 + j4.330127019
2, -120°	-1 - j1.732050808

Note that θ may be any positive or negative angle within the calculator's dynamic range for trigonometric functions, in radians or degrees.

```
POLAR TO RECT

SET FLAG FOR DEG
CLR FLAG FOR RAD

#

R1
             10.
Θ1
  1.047197551
RAD

P TO R

U:K1
              5.
JV:K2
  8.660254038

#

R1
             10.
Θ1
             60.
DEG

P TO R

U:K1
              5.
JV:K2
  8.660254038

END
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
	RECTANGULAR TO POLAR CONVERSION			
A-1	To print title	EXC (S)	0	Print title &
2	for degrees	SET FLAG		flag instruction
	for radians	CLEAR FLAG		
3	Data entry	EXC arc	0	
	x_1	CONT	0	x_1
	y_1	CONT	π	y_1
				R → P
4	To enter new (x, y) return to Step 3. The flag			r
	may be cleared or set as desired.			θ
5	To terminate	EXC ENDR		END
	POLAR TO RECTANGULAR CONVERSION			
B-1	To print title	EXC (T)	0	Print title &
2	for degrees	SET FLAG		flag instruction
	for radians	CLEAR FLAG		
3	Data entry	EXC hyp	0	
	r	CONT	0	r
	θ	CONT	π	θ
				P → R
4	To enter new (x, y) return to Step 3. The			x
	flag may be cleared or set as desired			y
5	To terminate	EXC ENDR		END
C	For use with other COMPLEX OPERATOR programs:			
	For R → P, with data already stored as x in K_1, y in K_2:			
	SET or CLR FLAG	EXC Rxx		
	For P → R, with data already stored as r in K_3, θ in K_4:			
	SET or CLR FLAG	EXC π		

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
0	LBL	S	PF	PF	R	E	C	T	SPC	T	LBL S R → P
10	0	SPC	P	0	L	A	R	EXC	SFG	RSET	
20	LBL	arc	PF	#	CD	PF	X	1	STOP	=	LBL arc
30	K	1	PRNT	J	Y	1	CD	STOP	=	K	
40	2	PRNT	LBL	Rxx	PF	CLR	IFFG	D/R	CONT	R	LBL Rxx
50	SPC	T	0	SPC	P	K	1	$\sqrt{\Sigma x^2}$	K	2	
60	=	D/R	D/R	=	K	3	PF	R	:	K	
70	3	PRNT	K	2	EXC	10^{oo}	K	3	x^a	×	
80	(	K	1	÷	K	3	)	arc	cos	=	
90	K	4	CLR	K	4	Ø	:	K	4	PRNT	
1 00	EXC	IFFG	π#	STOP#	LBL	T	PF	PF	P	0	LBL T P → R
10	L	A	R	SPC	T	0	SPC	R	E	C	
20	T	EXC	SFG	RSET	LBL	hyp	PF	#	CD	PF	LBL hyp
30	R	1	STOP	=	K	3	PRNT	Ø	1	CD	
40	STOP	=	K	4	PRNT	EXC	IFFG	LBL	π	PF	LBL π
50	CLR	IFFG	D/R	CONT	P	SPC	T	0	SPC	R	
60	K	3	×	K	4	cos	=	K	1	PF	
70	U	:	K	1	PRNT	K	3	×	K	4	
80	sin	=	K	2	J	V	:	K	2	PRNT	
90	π*	STOP*	LBL	SFG	PF	S	E	T	SPC	F	LBL SFG
2 00	L	A	G	SPC	F	0	R	SPC	D	E	
10	G	C	L	R	SPC	F	L	A	G	SPC	
20	F	0	R	SPC	R	A	D	STOP	RADR	GODP	
30	LBL	IFFG	CD	IFFG	D	E	G	π	CONT	IF=Ø	LBL IFFG
40	R	A	D	CONT	RADR	GODP	LBL	10^{oo}	10^{oo}	=	LBL 10^{oo}
	0	1	2	3	4	5	6	7	8	9	

Replace π STOP with EXC arc for repetitive operation.

* Replace π STOP with EXC hyp for repetitive operation.

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
50	10^{oo}	Ø	Ø	e^x	int	x^a	×	2	-	1	
60	=	×	RADR	GODP	LBL	ENDR	PF	(E)	(N)	(D)	LBL ENDR
70	CD	PF	PF	RSET							
80											
90											
00											
10											
20											
30											
40											
50											
60											
70											
80											
90											
00											
10											
20											
30											
40											
50											
60											
70											
80											
90											
	0	1	2	3	4	5	6	7	8	9	

NOTES

TITLE

Complex Operators: X, ÷, +, and −; Z_1^{Z2}, $\sqrt{Z_1^2 + Z_2^2}$

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

$K_\emptyset$: working
K_1 : x_1, then u
K_2 : y_1, then v
K_3 : x_2
K_4 : y_2
K_5 : working
K_6 : working
K_7 : saves x_1
K_8 : saves y_1
K_9 : logic variable

Labeled Subroutines:

(A) , (Q) , (R)
(W) , (X) , (Y)

×, ÷, +, −,
x^a, $\sqrt{\Sigma x^2}$,
D/R, 10^{oo}, ENDR

PROGRAM DESCRIPTION

This program calculates any one of the standard arithmetic operations:

$$z_1 \times z_2 = z_3 = u + jv \quad ; \quad z_1 \div z_2 = z_3 = u + jv$$

$$z_1 - z_2 = z_3 = u + jv \quad ; \quad z_1 + z_2 = z_3 = u + jv$$

or either of the two-operand functions

$$(z_1)^{(z_2)} = u + jv \quad \text{or} \quad (z_1^2 + z_2^2)^{\frac{1}{2}} = u + jv$$

for any pair of complex variables

$$z_1 = x_1 + jy_1 \quad \text{and} \quad z_2 = x_2 + jy_2$$

In each case the results u and v are stored in K_1 and K_2 in place of the original data x_1 and y_1, respectively. Thus the program is arranged so that these operations may be linked to one another, or to the other Complex Operator programs, in almost any fashion desired.

Thus one can do complicated operations on complex variables such as

$$\left[e^{(z_1 + z_2) \times z_3}\right]^2 \equiv z_4$$ simply by rewriting the expression as

$$z_1 + z_2) \times z_3)e^{z)z^2} = z_4$$ much as one writes standard real-variable operations in the calculator program format. Then the calculation of z_4 becomes a simple process of calling the appropriate programs from the tape cartridge, selecting the desired subroutine, entering data, and keying the necessary operator subroutines, in sequence.

When operations are linked, it is important to remember that each new operation is performed on the cumulative result of all previous operations. Thus linked operations can be performed on any arithmetic sentence that can be written as

$$A \# B = \# C = \# D = \# E = \# \ . \ . \ . = \# N = \text{END RESULT}$$

where the # signifies any of the operations in the series of Complex Operator programs.

Input data and results are printed out. In addition, each operator is printed out following the input data so that a complete record of the complex variables and the operations performed on them is available for verification. When the printer is not used, the results of any operation must be recalled manually from registers K_1 and K_2. The original data x_1 and y_1 are saved in K_7 and K_8 for verification when the printer is not used.

Algorithms:

$$(z_1)^{(z_2)} = (x_1 + jy_1)^{(x_2 + jy_2)} = e^{z_2 \ln z_1} = u + jv$$

$$= e^{(x_2 \ln \sqrt{x_1^2 + y_1^2} - y_2\theta)} \times \left\{ \cos\left(y_2 \ln \sqrt{x_1^2 + y_1^2} + x_2\theta\right) + j\sin\left(y_2 \ln \ldots + x_2\theta\right) \right\} \text{ where } \theta = \arccos \frac{x_1}{\sqrt{x_1^2 + y_1^2}}, \ -\pi \leq \theta \leq \pi$$

$$(z_1^2 + z_2^2)^{\frac{1}{2}} = \pm\sqrt{(x_1 + jy_1)^2 + (x_2 + jy_2)^2} = \pm\left[\sqrt{r}\cos\frac{\theta}{2} + j\left(\sqrt{r}\sin\frac{\theta}{2}\right)\right] = \pm(u + jv)$$

$$r = \left[(x_1^2 - y_1^2 + x_2^2 - y_2^2)^2 + (2x_1y_1 + 2x_2y_2)^2\right]^{\frac{1}{2}}$$

$$\theta = \arccos \frac{(x_1^2 - y_1^2 + x_2^2 - y_2^2)}{r}, \ -\pi \leq \theta \leq \pi$$

Note that the program calculates and prints out only the positive square root.

Reference:

A. Abramowitz and I. A. Stegun (editors), *Handbook of Mathematical Functions, AMS 55*, Dept. of Commerce, Washington, D.C., 1964.

NOTES

EXAMPLES

1. Multiply

$(7 + j3) \times (-27 - j8) =$

$-165 - j137$

2. Divide

$(7 + j3) \div (-27 - j8) =$

$-.2686002522 - j.0315258512$

3. Add

$(7 + j3) + (-27 - j8) =$

$-20 - j5$

4. Subtract

$(7 + j3) - (-27 - j8) =$

$34 + j11$

5. $z_1^{z_2}$

$(1 + j2)^{(2 + j1)} = -1.640101018 + j.2020503986$

6. $(z_1^2 + z_2^2)^{\frac{1}{2}}$

$[(1 + j2)^2 + (2 + j1)^2]^{\frac{1}{2}} =$

$2 + j2$

Note that the symbol # separates the various operations in the printout.

7. Chained operations

$(1 + j2) + (7 + j3)] \times (2 + j1)]$

$- (3 - j4)] \times (4 + j5)] \div (3 + j2)]$

$x^a(2 + j2) =$

-37.45243257

$-j71.84814785$

```
MULT;DIV;ADD;SUB
  Z2
Z1  ;RSSQ(Z1,Z2)
#
X1
              7.
JY1
              3.
*
X2
-            27.
JY2
-             8.
=
U:K1
-           165.
JV:K2
-           137.
#
X1
              7.
JY1
              3.
/
X2
-            27.
JY2
-             8.
=
U:K1
-.2686002522
JV:K2
-.0315258512
#
X1
              7.
JY1
              3.
+
X2
-            27.
JY2
-             8.
=
U:K1
-            20.
JV:K2
-             5.
#
X1
              7.
JY1
              3.
-
X2
-            27.
JY2
-             8.
=
U:K1
             34.
JV:K2
             11.
#
X1
              1.
JY1
              2.
  Z2
Z1
X2
              2.
JY2
              1.
=
U:K1
-1.640101018
JV:K2
 .2020503986
#
X1
              1.
JY1
              2.
RSSQ(Z1,Z2)
X2
              2.
JY2
              1.
=
U:K1
              2.
JV:K2
              2.
```

CHAINED OPERATIONS:

```
#
X1
              1.
JY1
              2.
+
X2
              7.
JY2
              3.
=
U:K1
              8.
JV:K2
              5.
*
X2
              2.
JY2
              1.
=
U:K1
             11.
JV:K2
             18.
-
X2
              3.
JY2
-             4.
=
U:K1
              8.
JV:K2
             22.
*
X2
              4.
JY2
              5.
=
U:K1
-            78.
JV:K2
            128.
/
X2
              3.
JY2
              2.
=
U:K1
 1.692307692
JV:K2
 41.53846154
  Z2
Z1
X2
              2.
JY2
              2.
=
U:K1
-37.45243257
JV:K2
-71.84814785
END
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
A	INITIAL OPERATION:			
1	To print title	EXC (A)	0	Title
2	Initial data entry	EXC D/R	0	#
	x_1	CONT	0	x_1
	y_1	CONT	0	y_1
3	Select operator	EXC $\times$, $\div$, + , - , x^a , or $\sqrt{\Sigma x^2}$	0	operator symbol
4	Second data entry			
	x_2	CONT	0	x_2
	y_2	CONT	π	y_2
5	If printer is not used, recall u and v from K_1 and K_2, respectively.			u v
6	To operate directly on u and v return to Step A-3 (Another Complex Operator program may be called from tape it desired.)			
7	Or to enter a new (x_1, y_1) for another operation return to Step A-2.			
8	To terminate	EXC ENDR		END
B	CHAINED OPERATIONS:			
1	To operate directly on an (x, y) pair already stored in K_1 and K_2 from a previous operation	EXC $\times$, $\div$, + , - , x^a , or $\sqrt{\Sigma x^2}$	0	operator symbol
2	Enter x_2 y_2	CONT CONT	0 π	x_2 y_2
3	Recall results as in A-5			u
4	For the next chained operation return to B-1. (A new Complex Operator program may be called from tape if desired.)			v
5	To terminate	EXC ENDR		END

PROGRAM STEPS

		0	1	2	3	4	5	6	7	8	9	COMMENTS
0	00	LBL	(A)	PF	PF	(M)	(U)	(L)	(T)	(;)	(D)	LBL (A)
	10	(I)	(V)	(;)	(A)	(D)	(D)	(;)	(S)	(U)	(B)	
	20	PF	(SPC)	(SPC)	(Z)	(2)	PF	(Z)	(1)	(SPC)	(SPC)	
	30	(;)	(R)	(S)	(S)	(Q)	(()	(Z)	(1)	(,)	(Z)	
	40	(2)	())	RSET	LBL	D/R	PF	(#)	CLR	PF	(X)	LBL D/R
	50	(1)	STOP	=	K	1	PRNT	(J)	(Y)	(1)	CD	
	60	STOP	=	K	2	PRNT	STOP	LBL	×	PF	(*)	LBL ×
	70	EXC	(W)	CLR	EXC	(X)	LBL	÷	PF	(/)	EXC	LBL ÷
	80	(W)	CLR	1	EXC	(X)	LBL	+	PF	(+)	EXC	LBL +
	90	(W)	CLR	1	EXC	(Y)	LBL	-	PF	(-)	EXC	LBL -
1	00	(W)	CLR	1	+/-	LBL	(Y)	=	K	9	EXC	LBL (Y)
	10	(Q)	K	9	×	K	3	+	K	1	=	
	20	K	1	K	9	×	K	4	+	K	2	
	30	=	K	2	EXC	(R)	LBL	(X)	=	K	9	LBL (X)
	40	EXC	(Q)	K	3	x^2	+	K	4	x^2	=	
	50	x^a	K	9	=	K	Ø	K	9	×	2	
	60	-	1	=	K	9	+/-	×	K	1	×	
	70	K	4	+	K	2	×	K	3	=	÷	
	80	K	Ø	=	K	5	K	1	×	K	3	
	90	+	K	2	×	K	4	×	K	9	=	
2	00	÷	K	Ø	=	K	1	K	5	=	K	
	10	2	LBL	(R)	CLR	PF	(U)	(:)	(K)	(1)	K	LBL (R)
	20	1	PRNT	(J)	(V)	(:)	(K)	(2)	K	2	PRNT	
	30	CD	π#	STOP#	LBL	x^a	PF	(SPC)	(SPC)	(Z)	(2)	LBL x^a
	40	PF	(Z)	(1)	EXC	(Q)	EXC	(W)	K	1	$\sqrt{\Sigma x^2}$	z_1 z_2

The two steps π STOP may be replaced with EXC D/R if automatic repetitive operation is desired.

		0	1	2	3	4	5	6	7	8	9	COMMENTS
	50	K	2	=	K	5	IF=Ø	CLR	=	K	1	
	60	=	K	2	EXC	(R)	CONT	K	2	EXC	10^{oo}	
	70	K	5	x^a	×	(	K	1	÷	K	5	
	80	)	arc	cos	=	K	6	×	K	3	+	
	90	K	4	×	K	5	ln	=	K	Ø	K	
3	00	3	×	K	5	ln	-	K	6	×	K	
	10	4	=	e^x	=	K	6	×	K	Ø	cos	
	20	=	K	1	K	6	×	K	Ø	sin	=	
	30	K	2	EXC	(R)	LBL	$\sqrt{\Sigma x^2}$	PF	(R)	(S)	(S)	LBL $\sqrt{\Sigma x^2}$
	40	(Q)	(	(Z)	(1)	,	(Z)	(2)	)	EXC	(Q)	RSSQ (z_1, z_2)
	50	EXC	(W)	K	1	x^2	-	K	2	x^2	+	
	60	K	3	x^2	-	K	4	x^2	=	K	Ø	
	70	K	1	×	K	2	+	K	3	×	K	
	80	4	)	×	2	=	K	6	$\sqrt{\Sigma x^2}$	K	Ø	
	90	=	K	5	K	6	EXC	10^{oo}	CD	(	K	
4	00	Ø	÷	K	5	)	arc	cos	=	÷	2	
	10	=	K	6	cos	×	K	5	$\sqrt{x}$	=	K	
	20	1	K	6	sin	×	K	5	$\sqrt{x}$	=	K	
	30	2	EXC	(R)	LBL	(Q)	K	1	=	K	7	LBL (Q)
	40	K	2	=	K	8	RADR	GODP	LBL	(W)	PF	LBL (W)
	50	(X)	(2)	CD	STOP	=	K	3	PRNT	(J)	(Y)	
	60	(2)	CD	STOP	=	K	4	PRNT	PF	(=)	RADR	
	70	GODP	LBL	10^{oo}	10^{oo}	=	10^{oo}	Ø	Ø	e^x	int	LBL 10^{oo}
	80	x^a	×	2	-	1	=	×	RADR	GODP	LBL	LBL ENDR
	90	ENDR	PF	(E)	(N)	(D)	CD	PF	PF	RSET		
		0	1	2	3	4	5	6	7	8	9	

NOTES

TITLE

Complex Operators: e^z, ln z, log z; z^2, $\sqrt{z}$, 1/z

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

$K_\emptyset$: working
K_1 : x, then u
K_2 : y, then v
K_3 : R } ln z
K_4 : I }
K_5 : R } log z
K_6 : I }
K_7 : x
K_8 : y
K_9 : logic variable

Labeled Subroutines:
LBL (B) , (Q) , (R) (V) , e^x , ln , log x^2 , $\sqrt{x}$, 1/x , D/R 10^{oo} , ENDR

PROGRAM DESCRIPTION

This program calculates any one of the single-operand functions

$$e^z \text{ , } \ln z \text{ , } \log z \text{ , } z^2 \text{ , } \sqrt{z} \text{ , or } 1/z$$

for any complex variable $z = x + jy$.

In each case the results u and v are stored in K_1 and K_2 in place of the original data x and y, respectively. Thus the program is arranged so that these operations may be linked to one another, or to the other Complex Operator programs, in almost any fashion desired.

Input data and results are printed out. In addition, each operator is printed out following the input data so that a complete record of the complex variables and the operations performed on them is available for verification. When the printer is not used, the results of any operation must be recalled manually from registers K_1 and K_2. The original data x_1 and y_1 are saved in K_7 and K_8 for verification when the printer is not used.

Algorithms:

1. $e^z = e^{(x + jy)} = (e^x \cos y) + j(e^x \sin y) = u + jv$;

2. $\ln z = \ln(x + jy) = \ln r + j\theta = u + jv$

   where $r = + \sqrt{x^2 + y^2}$ and $\theta = \arccos \frac{x}{r}$, $-\pi \leq \theta \leq \pi$;

3. $\log z = (\ln z)/(\ln 10) = u + jv$.

   When either ln z or log z is selected, both are calculated and may be recalled from registers K_3, K_4 or K_5, K_6, respectively.

4. $z^2 = (x + jy)^2 = (x^2 - y^2) + j(2xy) = u + jv$;

5. $\sqrt{z} = \pm \sqrt{x + jy} = \pm (\sqrt{r} \cos \frac{\theta}{2} + j \sqrt{r} \sin \frac{\theta}{2}) = \pm (u + jv)$

   where $r = + \sqrt{x^2 + y^2}$ and $\theta = \arccos \frac{x}{r}$, $-\pi \leq \theta \leq \pi$;

   Note that the program calculates and prints out only the positive root.

6. $\frac{1}{z} = \frac{1}{(x + jy)} = \left(\frac{x}{x^2 + y^2}\right) + j \left(\frac{-y}{x^2 + y^2}\right) = u + jv$

NOTES

EXAMPLES

1. e^{z}

   $e^{(1 + j2)} = -1.131204384 + j2.471726672$

2. ln z

   ln (1 + j2) = 0.8047189562 + j1.107148718

3. log z

   log(1 + j2) = +0.3494850022 + j0.4808285788

4. z^{2}

   $(1 + j2)^{2} = 3. + j4.$

5. $\sqrt{z}$

   $\sqrt{1 + j2} = \pm (1.27201965 + j0.7861513777)$

6. 1/z

   1/(1 + j2) = 0.2 - j0.4

7. Chained operations:

$$\left[\frac{1}{\log \sqrt{e^{(2 + j3)}}}\right]^{2} \equiv$$

$$e^{(2 + j3)}\ ]\ \sqrt{z}\ ]\ \log z\ ]\ \frac{1}{z}\ ]\ z^{2} \equiv$$

-0.6274435634

-j1.505864552

E^{Z}, LN Z, LOG Z

Z^{2}, SQRT Z, 1/Z

```
#

X1
                1.
JV1
                2.

 Z
E

U:K1
-1.131204384
JV:K2
 2.471726672

#

X1
                1.
JV1
                2.

LN Z

U:K1
 .8047189562
JV:K2
 1.107148718

#

X1
                1.
JV1
                2.

LOG Z

J:K1
 .3494850022
JV:K2
 .4808285788

#

X1
                1.
JV1
                2.

 2
Z

U:K1
-               3.
JV:K2
                4.

#

X1
                1.
JV1
                2.

SQRT Z

U:K1
  1.27201965
JV:K2
 .7861513777
```

```
#

X1
                1.
JV1
                2.

1/Z

J:K1
                .2
JV:K2
-               .4

END
```

CHAINED OPERATIONS:

```
#

X1
                2.
JV1
                3.

 Z
E

U:K1
-7.315110095
JV:K2
 1.042743656

SQRT Z

U:K1
 .1922836499
JV:K2
 2.711472496

LOG Z

U:K1
 .4342944819
JV:K2
 .6514417228

1/Z

U:K1
 .7084877209
JV:K2
-1.062731581

 2
Z

J:K1
-.6274435634
JV:K2
-1.505864552

END
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
A	INITIAL OPERATION:			
1	To print title	EXC (B)	0	Title
2	Initial data entry	EXC D/R	0	#
	x_1	CONT	0	x_1
	y_1	CONT	0	y_1
3	Select operator	EXC e^x , ln , log , x^2 , $\sqrt{x}$, or 1/x	π	operator symbol
4	If printer is not used, recall u and v from K_1 and K_2, respectively.			u v
5	To operate directly on u and v return to Step A-3. (Another Complex Operator program may be called from tape if desired.)			
6	Or to enter a new (x_1, y_1) for another operation return to Step A-2.			
7	To terminate	EXC ENDR		END
B	CHAINED OPERATIONS:			
1	To operate directly on an (x, y) pair already stored in K_1 and K_2 from a previous operation	EXC e^x , ln , log , x^2 , $\sqrt{x}$, or 1/x	π	operator symbol
2	Recall results as in A-4			u
3	For the next chained operation return to B-1. (A new Complex Operator program may be called from tape if desired.)			v
4	To terminate	EXC ENDR		END

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
0	LBL	B	PF	PF	SPC	Z	PF	E	SPC	,	LBL B
10	SPC	L	N	SPC	Z	,	SPC	L	O	G	
20	SPC	Z	PF	PF	SPC	2	PF	Z	SPC	,	
30	SPC	S	Q	R	T	SPC	Z	,	SPC	1	
40	/	Z	RSET	LBL	D/R	PF	#	CLR	PF	X	LBL D/R
50	1	STOP	=	K	1	PRNT	J	Y	1	CD	
60	STOP	=	K	2	PRNT	STOP	LBL	e^x	PF	SPC	LBL e^x
70	Z	CLR	E	EXC	Q	K	1	e^x	=	K	
80	3	×	K	2	cos	=	K	1	K	3	
90	×	K	2	sin	=	K	2	EXC	R	LBL	LBL ln
1 00	ln	PF	L	N	SPC	Z	CLR	EXC	V	LBL	LBL log
10	log	PF	L	O	G	SPC	Z	CLR	1	LBL	LBL V
20	V	=	K	9	RADR	=	K	Ø	EXC	Q	
30	K	1	$\sqrt{\Sigma x^2}$	K	2	=	K	3	K	2	
40	EXC	10oo	K	3	x^a	×	(	K	1	÷	
50	K	3	)	arc	cos	=	K	4	K	9	
60	IF<Ø	K	Ø	GODP	CONT	K	4	÷	1	Ø	
70	ln	=	K	6	K	3	ln	=	K	3	
80	÷	1	Ø	ln	=	K	5	K	9	IF=Ø	
90	K	3	=	K	1	K	4	=	K	2	
2 00	EXC	R	CONT	K	5	=	K	1	K	6	
10	=	K	2	LBL	R	CLR	PF	U	:	K	LBL R
20	1	K	1	PRNT	J	V	:	K	2	K	
30	2	PRNT	CD	π#	STOP#	LBL	x^2	PF	SPC	2	LBL x^2
40	CLR	Z	EXC	Q	K	1	×	K	2	×	

The two steps π STOP may be replaced with EXC D/R if automatic repetitive operation is desired.

	0	1	2	3	4	5	6	7	8	9	COMMENTS
50	2	=	K	3	K	1	x^2	−	K	2	
60	x^2	=	K	1	K	3	=	K	2	EXC	
70	(R)	LBL	$\sqrt{x}$	PF	(S)	(Q)	(R)	(T)	(SPC)	(Z)	LBL $\sqrt{x}$
80	CLR	1	+/−	EXC	(V)	K	4	÷	2	=	
90	K	4	cos	×	K	3	$\sqrt{x}$	=	K	1	
3 00	K	3	$\sqrt{x}$	×	K	4	sin	=	K	2	
10	EXC	(R)	LBL	1/x	PF	(1)	(/)	(Z)	CLR	EXC	LBL 1/x
20	(Q)	K	1	x^2	+	K	2	x^2	=	K	
30	3	1/x	×	K	1	=	K	1	CD	−	
40	K	2	)	÷	K	3	=	K	2	EXC	
50	(R)	LBL	(Q)	K	1	=	K	7	K	2	LBL(Q)
60	=	K	8	RADR	GODP	LBL	10^{oo}	10^{oo}	=	10^{oo}	LBL 10^{oo}
70	Ø	Ø	e^x	int	x^a	×	2	−	1	=	
80	×	RADR	GODP	LBL	ENDR	PF	(E)	(N)	(D)	CD	LBL ENDR
90	PF	PF	RSET								
00											
10											
20											
30											
40											
50											
60											
70											
80											
90											
	0	1	2	3	4	5	6	7	8	9	

NOTES

TITLE

Complex Operators: tan z, cos z, sin z; arctan z, arccos z, arcsin z

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

$K_{\emptyset}$: working
K_1 : x, then u
K_2 : y, then v
K_3 : R } cos z
K_4 : I } or working
K_5 : R } sin z
K_6 : I } or working
K_7 : x
K_8 : y
K_9 : logic variable

Labeled Subroutines:

(D) , (E) , (F)
(I) , (M) , (O)
(P) , (Q) , (R)

tan, cos, sin,
D/R, 10^{oo}, ENDR

PROGRAM DESCRIPTION

This program calculates any one of the single-operand functions

tan z, cos z, or sin z,

or calculates the principal value for any one of the functions

Arctan z, Arccos z, or Arcsin z

for any complex operand z = x + jy within the ranges of values given for the particular functions.

In each case the results u and v are stored in K_1 and K_2 in place of the original data x and y, respectively. Thus, the program is arranged so that these operations may easily be linked to one another, or to the other Complex Operator programs, in almost any fashion desired.

Input data and results are printed out. In addition, each operator is printed out following the input data so that a complete record of the complex variables and the operations performed on them is available for verification. When the printer is not used, the results of any operation must re recalled manually from registers K_1 and K_2. The original data x_1 and y_1 are saved in K_7 and K_8 for verification when the printer is not used.

Algorithms:

1. $\cos z = \cos(x + jy) = \cos x \cosh y + j \sin x \sinh y = u + jv$

2. $\sin z = \sin(x + jy) = \sin x \cosh y + j \cos x \sinh y = u + jv$

3. $\tan z = \frac{\sin z}{\cos z} = u + jv$

The algorithms used for the tan, cos, and sin always calculate both cos z and sin z with each operation, regardless of the choice of operators tan, cos, or sin. Thus cos z and sin z may be recalled as follows:

$\cos z$ as $K_3 + jK_4$, and $\sin z$ as $K_5 + jK_6$,

following any tan, cos, or sin operation.

The principal values for the functions Arctan z, Arccos z, and Arcsin z are given by:

1. $$\arctan z = \frac{1}{2}\arctan\left[\frac{2x}{1 - x^2 - y^2}\right] + j\frac{1}{4}\ln\left[\frac{x^2 + (y+1)^2}{x^2 + (y-1)^2}\right] = u + jv$$

   for $z^2 \neq -1$ and where $z = x + jy = t\cos\tau + jt\sin\tau$.

   For $|\tau| = \frac{\pi}{2}$, $t \neq 1$; otherwise for $-\pi \leq \tau \leq \pi$, $0 \leq t \leq 1$.

2. $\arccos z = \pm[\arccos\beta - j\ln(\alpha + (\alpha^2 - 1)^{\frac{1}{2}})] = \pm(u + jv)$, and

3. $\arcsin z = \arcsin\beta + j\ln(\alpha + (\alpha^2 - 1)^{\frac{1}{2}}) = u + jv$,

   where $\alpha = \frac{1}{2}(\sqrt{m} + \sqrt{n})$, $\beta = \frac{1}{2}(\sqrt{m} - \sqrt{n})$ for $m = (x+1)^2 + y^2$ and

   $n = (x-1)^2 + y^2$; and where $z = x + jy = s\cos\sigma + j\,s\sin\sigma$.

   For $\sigma = 0$ or $\pm\pi$, $0 \leq s \leq 1$; otherwise for $-\pi < \sigma < \pi$, $s \geq 0$.

   Note that for arccos z the program calculates and prints out only the positive value.

References:

1. A. Abramowitz and I. A. Stegun (editors), *Handbook of Mathematical Functions, AMS 55*, Dept. of Commerce, Washington, D.C., 1964.

2. E. Jahnke and F. Emde, *Tables of Functions with Formulas and Curves, fourth edition*, Dover Publications, New York, 1945.

NOTES

EXAMPLES

1. tan z

$$\tan (1 + j2) = 3.381282608 \times 10^{-2} + j1.014793616$$

2. cos z

$$\cos (1 + j2) = 2.032723007 - j3.051897799$$

3. sin z

$$\sin (1 + j2) = 3.165778513 + 1.959601041$$

4. arctan z

$$\arctan (1 + j2) = 1.338972522 + j0.4023594781$$

5. arccos z

$$\arccos (1 + j2) = 1.14371774 - j1.528570919$$

6. arcsin z

$$\arcsin (1 + j2) = 0.4270785864 + j1.528570919$$

7. Chained operations

arcsin {cos [arctan (1 + j2)]} ≡

(1 + j2) arctan] cos] arcsin ≡

(1 + j2) EXC (D)) EXC cos) EXC (F) =

0.2318238045 + j0.4023594781

```
TAN, COS, SIN Z,
  AND INVERSES

#

X1
              1.
JV1
              2.

TAN

U:K1
 3.381282608E-02
JV:K2
 1.014793616

#

X1
              1.
JV1
              2.

COS

U:K1
 2.032723007
JV:K2
-3.051897799

#

X1
              1.
JV1
              2.

SIN

U:K1
 3.165778513
JV:K2
 1.959601041

#

X1
              1.
JV1
              2.

ARCTAN

U:K1
 1.338972522
JV:K2
 .4023594781

#

X1
              1.
JV1
              2.

ARCCOS

U:K1
 1.14371774
JV:K2
-1.528570919

#

X1
              1.
JV1
              2.

ARCSIN

U:K1
 .4270785864
JV:K2
 1.528570919

END
```

```
CHAINED
   OPERATIONS:

#

X1
              1.
JV1
              2.

ARCTAN

U:K1
 1.338972522
JV:K2
 .4023594781

COS

U:K1
 .2486028939
JV:K2
-.4022479321

ARCSIN

U:K1
 .2318238045
JV:K2
 .4023594781

END
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
A	INITIAL OPERATION:			
1	To print title	EXC (M)	0	Title
2	Initial data entry	EXC D/R	0	#
	x_1	CONT	0	x_1
	y_1	CONT	0	y_1
3	Select operator	EXC tan, cos, or sin,		operator
		(D) for arctan,		symbol
		(E) for arccos, or		
		(F) for arcsin.	π	
4	If printer is not used, recall u and v from			u
	K_1 and K_2, respectively.			v
5	To operate directly on u and v return to Step A-3. (Another Complex Operator program may be called from tape if desired.)			
6	Or to enter a new (x_1, y_1) for another operation return to Step A-2.			
7	To terminate	EXC ENDR		END
B	CHAINED OPERATIONS:			
1	To operate directly on an (x, y) pair already stored in K_1 and K_2 from a previous operation	EXC tan, cos, sin, or (D), (E), (F) as in A-3.	π	operator symbol
2	Recall results as in A-4			
3	For the next chained operation return to B-1. (A new Complex Operator program may be called from tape if desired.)			
4	To terminate	EXC ENDR		END

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
0	LBL	M	PF	PF	T	A	N	,	SPC	C	LBL M
10	O	S	,	SPC	S	I	N	SPC	Z	,	
20	SPC	SPC	A	N	D	SPC	I	N	V	E	
30	R	S	E	S	RSET	LBL	D/R	PF	#	CLR	LBL D/R
40	PF	X	1	STOP	=	K	1	PRNT	J	Y	
50	1	CD	STOP	=	K	2	PRNT	STOP	LBL	tan	LBL tan
60	PF	T	A	N	CLR	1	+/-	EXC	P	LBL	LBL cos
70	cos	PF	C	O	S	CLR	EXC	P	LBL	sin	LBL sin
80	PF	S	I	N	CLR	1	LBL	P	=	K	LBL P
90	9	EXC	Q	K	1	cos	×	K	2	hyp	
1 00	cos	=	K	3	CLR	-	K	1	sin	×	cosh z to K_3, K_4
10	K	2	hyp	sin	=	K	4	K	1	sin	sinh z to K_5, K_6
20	×	K	2	hyp	cos	=	K	5	K	1	
30	cos	×	K	2	hyp	sin	=	K	6	K	
40	9	IF<Ø	K	3	x^2	+	K	4	x^2	=	
50	K	Ø	1/x	×	(	K	6	×	K	3	
60	-	K	5	×	K	4	=	K	2	K	
70	5	×	K	3	+	K	6	×	K	4	
80	=	÷	K	Ø	=	K	1	EXC	R	CONT	
90	LBL	I	K	9	IF=Ø	K	3	=	K	1	LBL I
2 00	K	4	=	K	2	EXC	R	CONT	K	5	
10	=	K	1	K	6	=	K	2	LBL	R	LBL R
20	CLR	PF	U	:	K	1	K	1	PRNT	J	
30	V	:	K	2	K	2	PRNT	CD	π#	STOP#	
40	LBL	D	PF	A	R	C	T	A	N	CLR	LBL D
	0	1	2	3	4	5	6	7	8	9	

The two steps π STOP may be replaced with EXC D/R if automatic repetitive operation is desired.

	0	1	2	3	4	5	6	7	8	9	COMMENTS
50	1	+/-	EXC	(0)	LBL	(E)	PF	(A)	(R)	(C)	LBL (E)
60	(C)	(O)	(S)	CLR	EXC	(0)	LBL	(F)	PF	(A)	LBL (F)
70	(R)	(C)	(S)	(I)	(N)	CLR	1	LBL	(0)	=	LBL (0)
80	K	9	EXC	(Q)	K	1	x^2	+	K	2	
90	x^2	+	1	=	K	3	K	9	IF≥Ø	K	
3 00	3	-	2	×	K	1	=	$\sqrt{x}$	÷	2	
10	=	K	6	K	3	+	2	×	K	1	
20	=	$\sqrt{x}$	÷	2	+	K	6	=	K	4	
30	-	2	×	K	6	)	Ø	=	K	5	
40	arc	cos	=	K	3	K	5	arc	sin	=	
50	K	5	K	4	x^2	-	1	)	$\sqrt{x}$	+	
60	K	4	=	ln	=	K	6	+/-	=	K	
70	4	EXC	(I)	CONT	K	1	×	2	÷	(	
80	2	-	K	3	=	arc	tan	÷	2	=	
90	K	1	CD	1	-	K	7	$\sqrt{\Sigma x^2}$	K	8	
4 00	)	IF<Ø	K	7	EXC	10^{00}	π	÷	2	+	
10	K	1	=	K	1	CONT	K	3	+	2	
20	×	K	2	=	÷	(	K	3	-	2	
30	×	K	2	=	ln	÷	4	=	K	2	
40	EXC	(R)	LBL	(Q)	K	1	=	K	7	K	LBL (Q)
50	2	=	K	8	RADR	GODP	LBL	10^{00}	10^{00}	=	LBL 10^{00}
60	10^{00}	Ø	Ø	e^x	int	x^a	×	2	-	1	
70	=	×	RADR	GODP	LBL	ENDR	PF	(E)	(N)	(D)	LBL ENDR
80	CD	PF	PF	RSET							
90											
	0	1	2	3	4	5	6	7	8	9	

NOTES

TITLE

Complex Operators: tanh z, cosh z, sinh z; arctanh z, arccosh z, arcsinh z

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

K_0 : working
K_1 : x, then u
K_2 : y, then v
K_3 : R } cosh z
K_4 : I } or working
K_5 : R } sinh z
K_6 : I } or working
K_7 : x
K_8 : y
K_9 : logic variable

Labeled Subroutines:

(G) , (H) , (J)
(K) , (L) , (N)
(R) , (Q) , $_3\Sigma_2$
Σ_1 , Σ_0 , D/R , 10^{00}
ENDR

PROGRAM DESCRIPTION

This program calculates any one of the single-operand hyperbolic functions

tanh z, cosh z, or sinh z,

or the principal value of any one of the inverse hyperbolic functions

Arctanh z, Arccosh z, or Arcsinh z,

for any z = x + jy, within the allowable range for the given function.

In each case the results u and v are stored in K_1 and K_2 in place of the original data x and y, respectively. Thus the program is arranged so that these operations may easily be linked to one another, or to the other Complex Operator programs, in almost any fashion desired.

Input data and results are printed out. In addition, each operator is printed out following the input data so that a complete record of the complex variables and the operations performed on them is available for verification. When the printer is not used, the results of any operation must be recalled manually from registers K_1 and K_2. The original data x_1 and y_1 are saved in K_7 and K_8 for verification when the printer is not used.

Algorithms:

1. $\cosh z = \cosh x \cos y + j \sinh x \sin y = u + jv$

2. $\sinh z = \sinh x \cos y + j \cosh x \sin y = u + jv$

3. $\tanh z = \frac{\sinh z}{\cosh z} = u + jv$

The algorithms used for the tanh, cosh, and sinh always calculate both cosh z and sinh z with each operation, regardless of the choice of operators tanh, cosh, or sinh. Thus cosh z and sinh z may be recalled as follows:

$$\cosh z \text{ as } K_3 + jK_4 \text{ , and } \sinh z \text{ as } K_5 + jK_6 \text{ ,}$$

following any tanh, cosh, or sinh operation.

The program calculates the principal value for Arctanh z from

$$4.\quad \operatorname{arctanh} z = \tfrac{1}{2} \ln\left[\frac{1+z}{1-z}\right] = \tfrac{1}{2}\left[\ln r + j\theta\right] = u + jv \text{ , } -\frac{\pi}{2} \le v \le \frac{\pi}{2} \text{ ,}$$

$$\text{where } r = (a^2 + b^2)^{\frac{1}{2}} \text{ and } \theta = \arccos \frac{a}{r} \times \begin{cases} +1 \text{ for } y \ge 0 \\ 0 \text{ for } r = 0 \\ -1 \text{ for } y < 0 \end{cases} \text{ , for } z^2 \neq 1 \text{ .}$$

$$a = \frac{1 - x^2 - y^2}{(1 - x^2)^2 + y^2} \qquad b = \frac{2y}{(1 - x^2)^2 + y^2}$$

The principal values for Arccosh z and Arcsinh z are obtained from

$$5.\quad \operatorname{arccosh} z = j \arccos z = u + jv \text{ , } 0 \le v \le \pi$$

$$6.\quad \operatorname{arcsinh} z = -j \arcsin jz = u + jv \text{ , } -\frac{\pi}{2} \le v \le \frac{\pi}{2}$$

References:

1. A. Abramowitz and I. A. Stegun (editors), *Handbook of Mathematical Functions, AMS 55*, Dept. of Commerce, Washington, D.C., 1964.

2. A. E. Kennelly, *Tables of Complex Hyperbolic and Circular Functions*, Harvard Un. Press, Cambridge, Mass., 1914.

3. G. A. Korn and T. M. Korn, *Mathematical Handbook for Scientists and Engineers, 2nd ed.*, McGraw-Hill, New York, 1968.

NOTES

EXAMPLES

1. tanh z

   tan h (1 + j1) = 1.083923327
   + j0.2717525853

2. cosh z

   cosh (1 + j1) = 0.8337300251
   + j0.9888977058

3. sinh z

   sinh (1 + j1) = 0.6349639148
   + j1.298457581

4. arctanh z

   arctanh (1 + j1) = 0.4023594781
   + j1.017221968

5. arccosh z

   arccosh (1 + j1) = 1.061275062
   + j0.9045568944

6. arcsinh z

   arcsinh (1 + j1) = 1.061275062
   + j0.6662394324

7. Chained operations:

   arcsinh {cosh [arctanh (1 + j2)]} ≡

   (1 + j2) arctanh) cosh) arcsinh ≡

   (1 + j2) EXC (J)) EXC Σ_1) EXC (L) =

   0.383410822 + j0.1503212048

```
TANH,COSH,SINH Z
& INVERSES

#

X1
              1.
JY1
              1.

TANH

U:K1
 1.083923327
JV:K2
 .2717525853

#

X1
              1.
JY1
              1.

COSH

U:K1
 .8337300251
JV:K2
 .9888977058

#

X1
              1.
JY1
              1.

SINH

U:K1
 .6349639148
JV:K2
 1.298457581

#

X1
              1.
JY1
              1.

ARCTANH

U:K1
 .4023594781
JV:K2
 1.017221968

#

X1
              1.
JY1
              1.

ARCCOSH

J:K1
 1.061275062
JV:K2
 .9045568944

#

X1
              1.
JY1
              1.

ARCSINH

J:K1
 1.061275062
JV:K2
 .6662394324

END
```

```
CHAINED
   OPERATIONS:

#

X1
              1.
JY1
              2.

ARCTANH

U:K1
 .1732867951
JV:K2
 1.178097245

COSH

U:K1
 .3884434935
JV:K2
 .1608985632

ARCSINH

J:K1
  .383410822
JV:K2
 .1503212048

END
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
A	INITIAL OPERATION:			
1	To print title	EXC (N)	0	Title
2	Initial data entry	EXC D/R	0	#
	x_1	CONT	0	x_1
	y_1	CONT	0	y_1
3	Select operator	EXC $_3\Sigma_2$ for tan h, Σ_1 for cosh, Σ_0 for sinh, (J) for arctanh, (K) for arccosh, or (L) for arcsinh.	π	operator symbol
4	If printer is not used, recall u and v from K_1 and K_2, respectively.			u v
5	To operate directly on u and v return to Step A-3. (Another Complex Operator program may be called from tape if desired.)			
6	Or to enter a new (x_1, y_1) for another operation return to Step A-2.			
7	To terminate	EXC ENDR		END
B	CHAINED OPERATIONS:			
1	To operate directly on an (x, y) pair already stored in K_1 and K_2 from a previous operation	EXC $_3\Sigma_2$, Σ_1, Σ_0 (J), (K), or (L) as in A-3.	π	operator symbol
2	Recall results as in A-4			u v
3	For the next chained operation return to B-1. (A new Complex Operator program may be called from tape if desired.)			
4	To terminate	EXC ENDR		END

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
0	LBL	(N)	PF	PF	(T)	(A)	(N)	(H)	(,)	(C)	LBL (N)
10	(O)	(S)	(H)	(,)	(S)	(I)	(N)	(H)	(SPC)	(Z)	
20	(&)	(SPC)	(I)	(N)	(V)	(E)	(R)	(S)	(E)	(S)	
30	RSET	LBL	D/R	PF	(#)	CLR	PF	(X)	(1)	STOP	LBL D/R
40	=	K	1	PRNT	(J)	(Y)	(1)	CD	STOP	=	
50	K	2	PRNT	STOP	LBL	$_3\Sigma_2$	PF	(T)	(A)	(N)	LBL $_3\Sigma_2$: TANH
60	(H)	CLR	1	+/-	EXC	(G)	LBL	Σ_1	PF	(C)	LBL Σ_1 : COSH
70	(O)	(S)	(H)	CLR	EXC	(G)	LBL	Σ_0	PF	(S)	LBL Σ_0 : SINH
80	(I)	(N)	(H)	CLR	1	LBL	(G)	=	K	9	LBL (G)
90	EXC	(Q)	K	1	hyp	cos	×	K	2	cos	
1 00	=	K	3	K	1	hyp	sin	×	K	2	
10	sin	=	K	4	K	1	hyp	sin	×	K	COSH → K_3, K_4
20	2	cos	=	K	5	K	1	hyp	cos	×	SINH → K_5, K_6
30	K	2	sin	=	K	6	K	9	IF<Ø	K	
40	3	x^2	+	K	4	x^2	=	K	Ø	1/x	
50	×	(	K	6	×	K	3	-	K	5	
60	×	K	4	=	K	2	K	5	×	K	
70	3	+	K	6	×	K	4	=	÷	K	
80	Ø	=	K	1	EXC	(R)	CONT	K	9	IF=Ø	
90	K	3	=	K	1	K	4	=	K	2	
2 00	EXC	(R)	CONT	K	5	=	K	1	K	6	
10	=	K	2	LBL	(R)	CLR	PF	(U)	(:)	(K)	LBL (R)
20	(1)	K	1	PRNT	(J)	(V)	(:)	(K)	(2)	K	
30	2	PRNT	CD	π#	STOP#	LBL	(J)	PF	(A)	(R)	LBL (J)
40	(C)	(T)	(A)	(N)	(H)	CLR	1	+/-	EXC	(H)	

	0	1	2	3	4	5	6	7	8	9

The two steps π STOP may be replaced with EXC D/R if automatic repetitive operation is desired.

		0	1	2	3	4	5	6	7	8	9	COMMENTS
	50	LBL	K	PF	A	R	C	C	O	S	H	LBL K
	60	CLR	EXC	H	LBL	L	PF	A	R	C	S	LBL L
	70	I	N	H	CLR	1	LBL	H	=	K	9	LBL H
	80	EXC	Q	K	1	x^2	+	K	2	x^2	=	
	90	K	3	+	1	=	K	Ø	-	2	×	
3	00	K	1	=	K	4	1/x	×	(	1	-	
	10	K	3	=	K	3	K	9	IF<Ø	K	2	
	20	×	2	÷	K	4	=	$\sqrt{\Sigma x^2}$	K	3	=	
	30	K	4	ln	÷	2	=	K	1	K	2	
	40	EXC	10^{oo}	K	4	x^a	×	(	K	3	÷	
	50	K	4	)	arc	cos	÷	2	=	K	2	
	60	EXC	R	CONT	K	9	IF=Ø	K	1	=	K	
	70	2	CONT	K	Ø	-	2	×	K	2	=	
	80	$\sqrt{x}$	=	K	4	x^2	+	4	×	K	2	
	90	=	$\sqrt{x}$	=	K	3	+	K	4	=	÷	
4	00	2	=	K	Ø	x^2	-	1	=	$\sqrt{x}$	+	
	10	K	Ø	=	ln	=	K	Ø	K	9	IF=Ø	
	20	K	Ø	=	K	1	K	3	-	K	4	
	30	=	÷	2	)	Ø	arc	cos	=	K	2	
	40	EXC	R	CONT	K	1	EXC	10^{oo}	K	Ø	=	
	50	K	1	K	4	-	K	3	=	÷	2	
	60	)	Ø	arc	sin	+/-	=	K	2	EXC	R	
	70	LBL	Q	K	1	=	K	7	K	2	=	LBL Q
	80	K	8	RADR	GODP	LBL	10^{oo}	10^{oo}	=	10^{oo}	Ø	LBL 10^{oo}
	90	Ø	e^x	int	x^a	×	2	-	1	=	×	
		0	1	2	3	4	5	6	7	8	9	

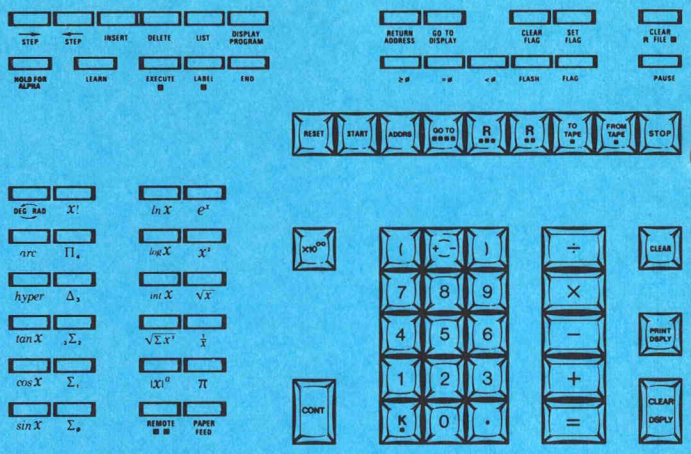
		0	1	2	3	4	5	6	7	8	9	COMMENTS
5	00	RADR	GODP	LBL	ENDR	PF	E	N	D	CD	PF	LBL ENDR
	10	PF	RSET									
	20											
	30											
	40											
	50											
	60											
	70											
	80											
	90											
	00											
	10											
	20											
	30											
	40											
	50											
	60											
	70											
	80											
	90											
	00											
	10											
	20											
	30											
	40											
		0	1	2	3	4	5	6	7	8	9	

SECTION 7 INTEGRATION AND DIFFERENTIAL EQUATIONS

OCTAL CODE	PRINT OUT	KEY	SYMBOL
001	LBL_	LABEL ■	LBL
002	FTP_	FROM TAPE ■	FTP
003	TTP_	TO TAPE ■	TTP
004	EXC_	EXECUTE ■	EXC
005	CFI_	CLEAR R FILE ■	CFI
007	GOTO	GO TO ■■■■	GT
010	R___	R ■■■	Rxxx
011	R__	R ■■	Rxx
040	CLDP	CLEAR DSPLY	CD
041	IFFL	FLASH	IFFL
042	S FG	SET FLAG	SFG
043	STOP	STOP	STOP
044	PRNT	PRINT DSPLY	PRNT
045	CLR	CLEAR	CLR
046	PAUS	PAUSE	PAUS
047	CLFG	CLEAR FLAG	CLFG
050	(	(	(
051	)	)	)
052	*	×	×
053	+	+	+
054	RSET	RESET	RSET
055	-	−	-
056	.	•	.
057	/	÷	÷
060	0	0	ø
061	1	1	1
062	2	2	2
063	3	3	3
064	4	4	4
065	5	5	5
066	6	6	6
067	7	7	7
070	8	8	8
071	9	9	9
072	ADR	ADDRS	ADR
073	STRT	START	STRT
074	IF<0	<ø	IF<ø
075	=	=	=
076	IF>=	≥ø	IF≥ø
077	IF=0	=ø	IF=ø

OCTAL CODE	PRINT OUT	KEY	SYMBOL
100	K_	K ●	K
101	DG/R	DEG RAD	D/R
102	ARC	arc	arc
103	HYP	$hyper$	hyp
104	TAN	$\tan x$	tan
105	COS	$\cos x$	cos
106	SIN	$\sin x$	sin
107	X!	$x!$	x!
110	PI4	Π_4	Π_4
111	DLT3	Δ_3	Δ_3
112	3SM2	${}_3\Sigma_2$	${}_3\Sigma_2$
113	SUM1	Σ_1	Σ_1
114	SUM0	$\Sigma_ø$	Σ_0
115	LN	$\ln x$	ln
116	LOG	$\log x$	log
117	INT	$int\, x$	int
120	RSSQ	$\sqrt{\Sigma x^2}$	$\sqrt{\Sigma x^2}$
121	X↑A	$\|x\|^a$	x^a
122	RM__	REMOTE ■■	RMT
123	E↑X	e^x	e^x
124	X↑2	x^2	x^2
125	SQRT	$\sqrt{x}$	$\sqrt{x}$
126	1/X	$\frac{1}{x}$	1/x
127	PI	π	π
130	PAPR	PAPER FEED	PF
131	*10↑	$\times 10^{oo}$	10^{oo}
132	CONT	CONT	CONT
133	R AD	RETURN ADDRESS	RADR
134	+/-	+/-	+/-
135	GODP	GO TO DISPLAY	GODP
136	IFFG	FLAG	IFFG
137	ENDR	END	ENDR

OCTAL CODE	PRINT OUT	KEY	SYMBOL
201		LABEL ■	LBL
202		FROM TAPE ■	FTP
203		TO TAPE ■	TTP
204		EXECUTE ■	EXC
205		CLEAR R FILE ■	CFI
207		BELL	BELL
210		SPACE ←	SPC←
211		TAB ←	TAB
220		STEP	STP←
221		INSERT	NSRT
222		DELETE	DLT
223		→ STEP	STP→
224		LIST	LIST
225		DISPLAY PROGRAM	DPRG
226		LEARN	LRN
240		SPACE	SPC
241	!	!	!
242	"	"	"
243	#	#	#
244	$	$	$
245	%	%	%
246	&	&	&
247	'	'	'
250	(	(	(
251	)	)	)
252	*	*	*
253	+	+	+
254	,	,	,
255	-	-	-
256	.	.	.
257	/	/	/
260	0	0	ø
261	1	1	1
262	2	2	2
263	3	3	3
264	4	4	4
265	5	5	5
266	6	6	6
267	7	7	7
270	8	8	8
271	9	9	9
272	:	:	:
273	;	;	;
274	<	<	<
275	=	=	=
276	>	>	>
277	?	?	?



OCTAL CODE	PRINT OUT	KEY	SYMBOL
300	@	@	@
301	A	A	A
302	B	B	B
303	C	C	C
304	D	D	D
305	E	E	E
306	F	F	F
307	G	G	G
310	H	H	H
311	I	I	I
312	J	J	J
313	K	K	K
314	L	L	L
315	M	M	M
316	N	N	N
317	O	O	O
320	P	P	P
321	Q	Q	Q
322	R	R	R
323	S	S	S
324	T	T	T
325	U	U	U
326	V	V	V
327	W	W	W
330	X	X	X
331	Y	Y	Y
332	Z	Z	Z
333	[	[	[
334	\	\	\
335	]	]	]
336	↑	↑	↑
337	_	_	_

TITLE

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

$K_{\emptyset}$: m

K_1 : a, then a + ih

K_2 : b

K_3 : h

K_4 : counter

K_5 : summation

K_6 : return address

K_8 : $y_i = g(a + ih)$

K_9 : $I = \int_a^b g(x)dx$

Subroutines used:

LBL Ø, 1, 2, 3, 4, 5, 6, 7, 8

PROGRAM DESCRIPTION

This program includes three separate subroutines for numerical integration (quadrature) by the Trapezoidal Rule, Simpson's 1/3 - Rule and Newton's 3/8 - Rule. Each of these Newton-Cotes (closed) quadrature formulas Q_{nn} is derived by approximating g(x) by interpolating polynomials of order n over each set of n subintervals of width h, and computing the integral of the interpolating polynomial over its interval of definition. These three quadrature Rules are of order Q_{11}, Q_{22}, and Q_{33}, respectively.

A comparison of these algorithms is given in the reference.

Subroutine LBL 1: The Trapezoidal Rule is defined as

$$I = \int_a^b g(x)dx \approx \frac{h}{2}\left[g(a) + 2\sum_{i=1}^{m-1} g(a + ih) + g(b)\right]$$

where the interval from $x_0 = a$ to $x_m = b$ is divided into m equal parts; m must be a positive integer; and then

$$h = \frac{b - a}{m}$$

Subroutine LBL 2: Simpson's Rule is defined as

$$I = \int_a^b g(x)dx \approx \frac{h}{3}\Big\{g(a) + 4g(a + h) + 2g(a + 2h)$$

$$+ 4g(a + 3h) + \ldots + 2g[a + (m - 2)h]$$

$$+ 4g[a + (m - 1)h] + g(b)\Big\}$$

$$= \frac{h}{3}\left\{ g(a) + 4\sum_{i=1,3,5,\ldots}^{m-1} g(a + ih) + 2\sum_{i=2,4,6,\ldots}^{m-2} g(a + ih) + g(b) \right\}$$

where the interval from $x_0 = a$ to $x_m = b$ is divided into m equal parts; m must be an even, positive integer, and then

$$h = \frac{b - a}{m}$$

If an odd or negative value is entered for m, the program automatically converts it to the next higher even, positive integer.

Subroutine LBL 3: Newton's $\frac{3}{8}$ - Rule yields the equation

$$I = \int_a^b g(x)dx$$

$$\approx \frac{3h}{8}\,[g(a) + 3g(a + h) + 3g(a + 2h)$$

$$+ 2g(a + 3h) + 3g(a + 4h) + \ldots$$

$$+ 3g(a + (m - 1)h) + g(a + mh)]$$

where the interval between $x_0 = a$ and $x_m = b$ is divided into m equal parts; m must be a positive integer that is integrally divisible by 3, and then

$$h = \frac{b - a}{m}$$

If a negative or non-factorable m is entered, the program will automatically convert it to the next higher positive integer divisible by 3.

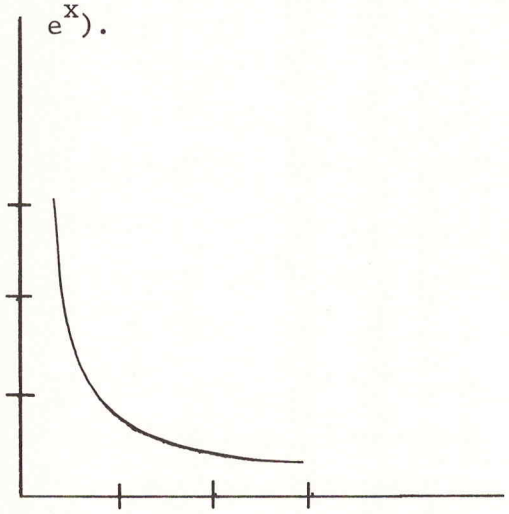
Reference:

T. R. McCalla, *Introduction to Numerical Methods and FORTRAN Programming*, John Wiley and Sons, New York, 1967.

NOTES

EXAMPLES

(1) Given $g(x) = \frac{1}{x}$, find the definite integral for the interval $x_0 = 1$ to $x_m = e$ (the value for e may be easily obtained by keying CD, e^x, e^x).



Program g(x) as $g(K_1) = K_8$:

```
0060   LBL_  F0
0061   0     F0
0062   K_    F0
0063   1     F0
0064   1/X   F0
0065   =     F0
0066   K_    F0
0067   8     F0
0068   R AD  F0
0069   =     F0
0070   K_    F0
0071   6     F0
0072   GODP  F0
```

Let a = 1

b = e

m = 100

```
NUMERICAL
  INTEGRATION

ENTER
A=
             1.
B=
 2.718281828

M=
           100.

FOR TRAP EXC 1
FOR SIMP EXC 2
FOR NEWT EXC 3

H=
 1.718281828E-02

#1

INTEGRAL =

 1.000021274

END
```

```
NUMERICAL
  INTEGRATION

ENTER
A=
             1.
B=
 2.718281828

M=
           100.

FOR TRAP EXC 1
FOR SIMP EXC 2
FOR NEWT EXC 3

M'=
           102.
H=
 1.684590028E-02

#3

INTEGRAL =

 1.000000006

END
```

```
NUMERICAL
  INTEGRATION

ENTER
A=
             1.
B=
 2.718281828

M=
           100.

FOR TRAP EXC 1
FOR SIMP EXC 2
FOR NEWT EXC 3

M'=
           100.
H=
 1.718281828E-02

#2

INTEGRAL =

 1.000000003

END
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1	Press CD 6 Ø GODP LRN LBL Ø Program your function g(x) = y as $g(K_1) = K_8$. You have 140 steps, and you may use K_7 or any Rxx for intermediate storage. Do not alter the value stored in K_1 in programming $g(K_1)$. Finish with the sequence	RADR = K 6 GODP LRN		
2		START	0	Title and data entry instruction
3	a	CONT	0	a
	b	CONT	0	b
	m	CONT	0	m
4	Choose desired algorithm			Execution instruction
	For Trapezoidal Rule	EXC 1	I	m'*
	For Simpson's Rule	EXC 2	I	h
	For Newton's Rule	EXC 3	I	
				#1, #2, or #3 to designate algorithm used
5	To integrate the same g(x) using another of the algorithms return to Step 2.			
6	To integrate a new function return to Step 1.			Integral I : K_9

*Adjusted value of m as necessary to meet the requirements of Simpson's or Newton's Rules.

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
0	EXC	8	LBL	4	K	4	x^2	−	K	4	LBL 4
10	×	K	Ø	)	x^a	+	1	)	×	K	
20	8	+	K	5	=	K	5	×	RADR	GODP	
30	LBL	5	=	K	9	K	3	Σ_1	K	4	LBL 5
40	−	1	=	K	4	IF≥Ø	K	6	−	2	
50	)	GODP	CONT	EXC	7						
60	LBL	Ø									LBL Ø
70				Program your function g(x) = y as							g(x) = y as
80	$g(K_1) = K_8$ beginning with LBL Ø at Step 60. Register K_7										$g(K_1) = K_8$
90	and any of the Rxx registers may be used as intermediate										
1 00	storage registers in programming $g(K_1) = K_8$.										
10											
20											
30											
40											
50	Be sure to finish with the sequence										
60				RADR	=	K	6	GODP			
70	before you reach step 200										
80											
90											
2 00	LBL	1	K	Ø	EXC	6	(#)	(1)	EXC	Ø	LBL 1: TRAP
10	EXC	4	K	3	÷	2	EXC	5	LBL	2	LBL 2: SIMP
20	(M)	(')	(=)	K	Ø	÷	2	)	int	−	
30	K	Ø	÷	2	)	x^a	)	Σ_0	K	Ø	
40	PRNT	EXC	6	(#)	(2)	EXC	Ø	K	4	÷	

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
50	2	)	int	-	K	4	÷	2	)	x^a	
60	+	.	5	)	×	2	×	(	EXC	4	
70	K	3	÷	3	EXC	5	LBL	3	M	'	LBL 3: NEWT
80	=	K	Ø	÷	3	)	int	-	K	Ø	
90	÷	3	)	IF<Ø	+	1	CONT	=	×	4	
3 00	=	int	Σ_0	K	Ø	PRNT	EXC	6	#	3	
10	EXC	Ø	K	4	÷	3	)	int	-	K	
20	4	÷	3	)	x^a	+	1	=	×	(	
30	EXC	4	K	3	×	3	÷	8	EXC	5	
40	LBL	6	=	K	4	1/x	×	(	K	2	LBL 6
50	-	K	1	=	K	3	H	=	PRNT	PF	H =
60	RADR	GODP	LBL	7	PF	I	N	T	E	G	LBL 7
70	R	A	L	SPC	=	K	9	PF	PRNT	PF	Results
80	E	N	D	PF	PF	PF	RSET	LBL	8	PF	LBL 8
90	PF	N	U	M	E	R	I	C	A	L	Title and
4 00	PF	SPC	SPC	I	N	T	E	G	R	A	data entry
10	T	I	O	N	CLR	PF	E	N	T	E	
20	R	=	K	5	A	=	STOP	=	K	1	
30	PRNT	B	=	CD	STOP	=	K	2	PRNT	PF	
40	M	=	CD	STOP	=	x^a	1	=	K	Ø	
50	PRNT	PF	F	O	R	SPC	T	R	A	P	
60	SPC	E	X	C	SPC	1	PF	F	O	R	
70	SPC	S	I	M	P	SPC	E	X	C	SPC	
80	2	PF	F	O	R	SPC	N	E	W	T	
90	SPC	E	X	C	SPC	3	CD	PF	RSET		
	0	1	2	3	4	5	6	7	8	9	

NOTES

TITLE

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

PROGRAM DESCRIPTION

This program is designed to solve ordinary first order differential equations where the derivative may be expressed as a function of x and y, that is,

$$\frac{dy}{dx} = g(x, y).$$

For given initial values x_0, y_0, and the fixed x-interval h, the quartic Runge-Kutta algorithm yields the solution points x_1, y_1 ; x_2, y_2 ; x_3, y_3 ; . . . ; x_n, y_n from the recursion formulas

$$x_{n+1} = x_n + h$$

$$y_{n+1} = y_n + \frac{1}{6}(p + 2q + 2r + s)$$

where

$$p = h \times g(x_n , y_n) ;$$

$$q = h \times g(x_n + \frac{h}{2} , y_n + \frac{p}{2}) ;$$

$$r = h \times g(x_n + \frac{h}{2} , y_n + \frac{q}{2}) ;$$

$$s = h \times g(x_n + h , y_n + r) .$$

The program includes subroutines for two modes of execution.

REGISTERS USED

$K_{\emptyset}$: J
K_1 : x_0, y_1, ...
K_2 : x_0, y_1, ...
K_3 : working
K_4 : working
K_5 : h
K_7 : working
K_8 : K
K_9 : x_n

$R_{\emptyset\emptyset\emptyset}$: Ind. Adr. Reg.
$R_{\emptyset 1}$: p
$R_{\emptyset 2}$: q
$R_{\emptyset 3}$: r
$R_{\emptyset 4}$: s

Subroutines used:
LBL Ø, 1, 2, 3, 4, 5, 6, 7

1. LBL 1, Manual mode: In this mode the user selects J, number of successive points to be calculated; h, the x-interval value; and enters the initial x_0 and y_0. If J = 0 is entered, then the program will calculate one point at a time, and h may be modified by changing the value stored in K_5. To calculate successive points, it is only necessary to press EXC 3. To terminate the program press SET FLAG prior to calculating the last point wanted.

   For any $J \neq 0$, the program will calculate and print the number of points selected.

2. LBL 2, Automatic mode: In this mode the user can select the number of points J to be calculated over an interval from an initial x_0 to a final x_n. He also has the option of selecting a value K which will cause the program to print only every Kth point calculated. Thus for example, for a given equation one can calculate y values for 100 points between x_0 and x_n, that is, use a small h for accuracy, but only print out every 10th point calculated.

   The program contains its own instruction for data entry for either mode. The examples show printouts for the alternative modes of execution.

Reference:

T. R. McCalla, *Introduction to Numerical Methods and FORTRAN Programming*, John Wiley and Sons, New York, 1967.

NOTES

EXAMPLES

Given ydx - xdy = xydx ;

rewrite as

$$\frac{dy}{dx} = g(x, y) = y(1 - x)/x$$

Program g(x, y) as $g(K_3, K_4)$

CD 4 1 4 GODP LRN LBL Ø CLR

1 - K_3) ÷ K_3 × K_4 = K_4 × K_5

= Rxx Rxx Ø Ø RADR GODP LRN

Example 1: $g(K_3, K_4)$

Manual Execution

Let J = 0

h = .2

$x_0 = 1$

$y_0 = 1$

Calculate 5 points

0414	LBL_	F0
0415	0	F0
0416	CLR	F0
0417	1	F0
0418	-	F0
0419	K_	F0
0420	3	F0
0421	)	F0
0422	/	F0
0423	K_	F0
0424	3	F0
0425	*	F0
0426	K_	F0
0427	4	F0
0428	=	F0
0429	K_	F0
0430	4	F0
0431	*	F0
0432	K_	F0
0433	5	F0
0434	=	F0
0435	R__	F0
0436	R__	F0
0437	0	F0
0438	0	F0
0439	R AD	F0
0440	GODP	F0

Example 2:

Automatic Execution

Let J = 100

K = 20

$x_0 = 1$

$y_0 = 1$

$x_n = 2$

Print 5 points;

execution time

∿ ½ minute per

point.

```
1ST-ORD DIFF EQN
 DY/DX = G(X,Y)

FOR MAN EXC 1
FOR AUTO EXC 2

#1
INPUT DATA

DO J POINTS; J=
              0.

H=
              .2

X0=
              1.

Y0=
              1.

XY PAIRS

              1.
              1.

            1.2
 .9824784206

            1.4
 .9384501363

            1.6
  .878100841

            1.8
  .808794338

              2.
 .7357609976

END
```

```
1ST-ORD DIFF EQN
 DY/DX = G(X,Y)

FOR MAN EXC 1
FOR AUTO EXC 2

#2
INPUT DATA

DO J POINTS; J=
            100.

PRINT NTH POINTS
N=
             20.

X0=
              1.

Y0=
              1.

XN=
              2.

H=
             .01

XY PAIRS

              1.
              1.

            1.2
 .9824769037

            1.4
 .9384480645

            1.6
 .8780986178

            1.8
 .8087921355

              2.
 .7357588824

END
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1	To program your function first press			
	CD 4 1 4 GODP LRN LBL Ø CLR D/R*			
	Now enter your function g(x, y) as $g(K_3, K_4)$			
	finishing with the sequence			
	= K_4 × K_5 = Rxx Rxx Ø Ø RADR GODP LRN.			
	*D/R should be omitted unless your function			
	includes trigonometric functions in degrees.			
2		START		Title
				Instructions
3a	For manual execution	EXC 1	0	#1 or #2
	Number of points J	CONT	0	Input Data
	x-interval h	CONT	0	as entered
	x_0	CONT	0	XY Pairs
	y_0	CONT	0	x_0
			or	y_0
	If M = 0, one point will be calculated:		y_i	.
				.
	To calculate the			.
	next point:	EXC 3	y_{i+1}	x_i : K_1
				y_i : K_2
	h may be changed prior to calculating			.
	a new point by:	CD h_{new} = K_5		.
				.
	To calculate the last point			x_n
	wanted:	SFG EXC 3	y_n	y_n
				END
3b	For auto execution	EXC 2	0	See examples
	Number of points J	CONT	0	for detailed
	Print Kth points K	CONT	0	printouts.
	x_0	CONT	0	
	y_0	CONT	0	
	final x_n	CONT	0	

PROGRAM STEPS

		0	1	2	3	4	5	6	7	8	9	COMMENTS
Ø	00	PF	PF	1	S	T	-	O	R	D	SPC	
	10	D	I	F	F	SPC	E	Q	N	PF	SPC	
	20	D	Y	/	D	X	SPC	=	SPC	G	(	
	30	X	,	Y	)	SPC	PF	F	O	R	SPC	
	40	M	A	N	SPC	E	X	C	SPC	1	PF	
	50	F	O	R	SPC	A	U	T	O	SPC	E	
	60	X	C	SPC	2	RSET	LBL	3	Rxx	Ø	Ø	LBL 3
	70	+	1	=	Rxx	Ø	Ø	EXC	Ø	K	1	
	80	+	K	5	÷	2	=	K	3	K	2	
	90	+	Rxx	Rxx	Ø	Ø	÷	2	=	K	4	
1	00	Rxx	Ø	Ø	-	3	)	IF<Ø	EXC	3	CONT	
	10	K	1	+	K	5	=	K	3	K	2	
	20	+	Rxx	Ø	3	=	K	4	Rxx	Ø	Ø	
	30	-	4	)	IF<Ø	EXC	3	CONT	K	3	=	
	40	K	1	Rxx	Ø	2	+	Rxx	Ø	3	)	
	50	×	2	+	Rxx	Ø	1	+	Rxx	Ø	4	
	60	=	÷	6	+	K	2	=	K	2	=	
	70	K	4	CD	=	Rxx	Ø	Ø	+/-	1	Σ_0	
	80	K	Ø	÷	K	8	=	K	7	-	K	
	90	7	int	)	IF=Ø	PF	PF	CLR	K	1	PRNT	
2	00	K	2	PRNT	CONT	K	Ø	-	1	)	IF≥Ø	
	10	EXC	3	CONT	K	Ø	IF=Ø	EXC	4	CONT	IFFG	
	20	EXC	4	CONT	K	2	RSET	LBL	1	PF	#	LBL 1
	30	1	EXC	6	H	=	CD	STOP	=	K	5	
	40	PRNT	EXC	5	EXC	7	EXC	3	LBL	2	PF	LBL 2
		0	1	2	3	4	5	6	7	8	9	

	0	1	2	3	4	5	6	7	8	9	COMMENTS
	#	2	EXC	6	P	R	I	N	T	SPC	
	K	T	H	SPC	P	O	I	N	T	S	
	K	=	CD	STOP	=	K	8	PRNT	EXC	5	
	PF	X	N	=	CD	STOP	=	K	9	PRNT	
	PF	-	K	1	)	÷	K	Ø	=	K	
3	5	H	=	PRNT	EXC	7	EXC	3	LBL	4	LBL 4
	PF	PF	E	N	D	CD	PF	PF	RSET	LBL	LBL 5
	5	PF	CD	=	Rxxx	Ø	Ø	Ø	X	Ø	
	=	STOP	=	K	1	=	K	3	PRNT	Y	
	Ø	=	CD	STOP	=	K	2	=	K	4	
	PRNT	RADR	GODP	LBL	6	I	N	P	U	T	LBL 6
	SPC	D	A	T	A	CD	PF	D	O	SPC	
	J	SPC	P	O	I	N	T	S	'	SPC	
	J	=	=	K	8	STOP	=	K	Ø	PRNT	
	PF	RADR	GODP	LBL	7	PF	PF	X	Y	SPC	LBL 7
4	P	A	I	R	S	K	1	PF	PRNT	K	
	2	PRNT	RADR	GODP	LBL	Ø	CLR	D/R*			LBL Ø
											Your function
											g(x, y)
				Program your function							≡ $g(K_3, K_4)$
	g(x, y) as $g(K_3, K_4)$ finishing with the sequence										
								=	K	4	
	×	K	5	=	Rxx	Rxx	Ø	Ø	RADR	GODP	
	*The D/R step should be omitted unless your function										
	includes trigonometric function arguments in degree										
	measure.										

NOTES

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

$K_{\emptyset}$: h
K_1 : $x_0, \cdots, x_n$
K_2 : $y_0, \cdots, y_n$
K_3 : $y'_0, \cdots, y'_n$
K_4 : $y''_0, \cdots$
K_5 : $y''_{n-1}, \cdots$
K_6 : $y''_{n-2}, \cdots$
K_7 : J
K_8 : K
K_9 : x_n

Subroutines used:
LBL Ø, 1, 2, 3, 4, 5, 6, 7, 8

PROGRAM DESCRIPTION

This program is designed to solve ordinary second order differential equations whose second derivatives may be expressed as a function of x, y, and dy/dx, that is, as

$$\frac{d^2y}{dx^2} = g(x, y, \frac{dy}{dx}) \text{ or } y'' = g(x, y, y')$$

Given the initial values x_0, y_0, y'_0, and the x-interval value h, the Taylor algorithm yields the solutions $y_1, y_2, \cdots, y_n$ and the first derivatives $y'_1, y'_2, \cdots, y'_n$ using the recursion formulas:

$$x_{n+1} = x_n + h$$

$$y_{n+1} = y_n + h \times y'_n + \frac{h^2}{6}(4y''_n - y''_{n-1})$$

$$y'_{n+1} = y'_n + \frac{h}{12}(5y''_{n-2} - 16y''_{n-1} + 23y''_n)$$

where the notation $y' \equiv dy/dx$ and $y'' \equiv d^2y/dx^2$

For initial approximations at n = 0 we first evaluate y''_0 and $y^3{}'_0$ where

$$y''_0 = g(x_0, y_0, y'_0) \text{ and}$$

$$y^3{}'_0 = \frac{d}{dy}[g(x, y, y')] \text{ evaluated at } x_0, y_0, y'_0$$

After entry of these initial values the program automatically calculates the starting values for y''_{-2} and y''_{-1} from

$$y''_{-2} = y''_0 - 2h \times y^3{}'_0 \text{ and } y''_{-1} = y''_0 - h \times y^3{}'_0$$

The program includes subroutines for two modes of execution.

1. LBL 1, Manual mode: In this mode the user selects J, number of successive points to be calculated; h, the x-interval value; and enters the initial x_0, y_0 and y'_0. If J = 0 is entered, then the program will calculate one point at a time, and h may be modified by changing the value stored in $K_{\emptyset}$. To calculate successive points, it is only necessary to press EXC 3. To terminate the program press SET FLAG prior to calculating the last point wanted.

   For any $J \neq 0$, the program will calculate and print the number of points selected.

2. LBL 2, Automatic mode: In this mode the user can select the number of points J to be calculated over an interval from an initial x_0 to a final x_n. He also has the option of selecting a value K which will cause the program to print only every Kth point calculated. Thus for example, for a given equation one can calculate y values for 100 points between x_0 and x_n, that is, use a small h for accuracy, but only print out every 10th point calculated.

   The program contains its own instruction for data entry for either mode. The examples show printouts for the alternative modes of execution.

Reference:

T. R. McCalla, *Introduction to Numerical Methods and FORTRAN Programming*, John Wiley and Sons, New York, 1967.

NOTES

EXAMPLES

Solve the differential equation

$$\frac{d^2y}{dx^2} = \frac{1 - y^2}{10}\frac{dy}{dx} - y$$

Program g(x, y, y') as $g(K_1, K_2, K_3)$ as

CD 4 3 8 GODP LRN LBL Ø CLR
1 - K_2 x^2) ÷ 1 Ø × K_3 - K_2 =
K_4 RADR GODP LRN

Example 1: $g(K_1, K_2, K_3)$

Manual Execution

Let J = 0

h = .1

$x_0 = 0$

$y_0 = 1$

$y'_0 = 0$

$y''_0 = -1$

$y^{3'}_0 = 0$

Calculate 5 points

```
0438   LBL_   F0
0439   0      F0
0440   CLR    F0
0441   1      F0
0442   -      F0
0443   K_     F0
0444   2      F0
0445   X^2    F0
0446   )      F0
0447   /      F0
0448   1      F0
0449   0      F0
0450   *      F0
0451   K_     F0
0452   3      F0
0453   -      F0
0454   K_     F0
0455   2      F0
0456   =      F0
0457   K_     F0
0458   4      F0
0459   R AD   F0
0460   GODP   F0
```

Example 2:

Automatic Execution

Let J = 50

K = 10

$x_0 = 0$

$y_0 = 1$

$y'_0 = 0$

$y''_0 = -1$

$y^{3'}_0 = 0$

$x_n = .5$

```
2ND-ORD DIFF EQN
 Y" = G(X,Y,Y')
FOR MAN EXC 1
FOR AUTO EXC 2
#1
INPUT DATA
DO J POINTS; J=
              0.
X0=
              0.
Y0=
              1.
Y'0=
              0.
Y"0=
-             1.
Y30=
              0.
H=
              .1
XY PAIRS, Y'
              0.
              1.
              0.
**
                .1
            .995
-             .1
**
                .2
 .9800326683
-.1990607854
**
                .3
 .9552462915
- .296037923
**
                .4
 .9208916203
-.3903098345
**
                .5
 .8772782134
-.4811067736
END
```

```
2ND-ORD DIFF EQN
 Y" = G(X,Y,Y')
FOR MAN EXC 1
FOR AUTO EXC 2
#2
INPUT DATA
DO J POINTS; J=
             50.
PRINT NTH POINTS
N=
             10.
X0=
              0.
Y0=
              1.
Y'0=
              0.
Y"0=
-             1.
Y30=
              0.
XN=
              .5
H=
             .01
XY PAIRS, Y'
              0.
              1.
              0.
**
                .1
 .9950040733
-9.983625945E-02
**
                .2
 .9800649071
-.1987090746
**
                .3
 .9553244902
- .295716512
**
                .4
 .9210116933
- .390022768
**
                .5
 .8774358064
-.4808541652
END
```

Note that $y^{3'} = y''(\frac{1 - y^2}{10}) - y'(\frac{5 + y}{5})$ and that

$y''_0 = -1$ and $y^{3'}_0 = 0$ at $x_0 = 0$ $y_0 = 1$ $y' = 0$

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1	To program your function first press CD 4 3 8 GODP LRN LBL Ø CLR D/R* Now enter your function g(x, y, y') as $g(K_1, K_2, K_3)$ finishing with the sequence = K_4 RADR GODP LRN. *D/R should be omitted unless your function includes trigonometric functions in degrees.			
2		START		Title
3a	For manual execution	EXC 1	0	Instructions
	Number of points J	CONT	0	Input data
	x-interval h	CONT	0	as entered
	x_0	CONT	0	XY Pairs
	y_0	CONT	0	Y'
	y'_0	CONT	0	END
	y''_0	CONT	0	
	y^3_0	CONT	0	See examples
			or	for detailed
	If M = 0, one point will be calculated; to calculate the next point:	EXC 3	y_i y_{i+1}	printouts.
	h may be changed prior to calculating a new point by:	CD h_{new} = K_5		
	To calculate the last point wanted:	SFG EXC 3	y_n	
3b	For auto execution	EXC 2	0	
	Number of points J	CONT	0	
	Print Nth points K	CONT	0	
	x_0	CONT	0	
	y_0	CONT	0	
	y'_0	CONT	0	
	y''_0	CONT	0	
	y^3_0	CONT	0	
	final x_n	CONT	0	

PROGRAM STEPS

		0	1	2	3	4	5	6	7	8	9	COMMENTS
0	00	PF	PF	2	N	D	-	O	R	D	SPC	
	10	D	I	F	F	SPC	E	Q	N	PF	SPC	
	20	Y	"	SPC	=	SPC	G	(	X	,	Y	
	30	,	Y	'	)	SPC	PF	F	O	R	SPC	
	40	M	A	N	SPC	E	X	C	SPC	1	PF	
	50	F	O	R	SPC	A	U	T	O	SPC	E	
	60	X	C	SPC	2	RSET	LBL	3	EXC	Ø	K	LBL 3
	70	Ø	Σ_1	K	4	×	4	-	K	5	=	
	80	÷	6	×	K	Ø	x^2	+	K	Ø	×	
	90	K	3	+	K	2	=	K	2	K	6	
1	00	×	5	-	1	6	×	K	5	+	2	
	10	3	×	K	4	=	÷	1	2	×	K	
	20	Ø	+	K	3	=	K	3	K	5	=	
	30	K	6	K	4	=	K	5	K	7	-	
	40	1	=	K	7	÷	K	8	)	int	-	
	50	(	K	7	÷	K	8	)	)	IF=Ø	PF	
	60	*	*	CLR	PF	K	1	PRNT	K	2	PRNT	
	70	PF	K	3	PRNT	CONT	K	7	IF=Ø	EXC	4	
	80	CONT	-	1	)	IF≥Ø	EXC	3	CONT	IFFG	EXC	
	90	4	CONT	K	2	RSET	LBL	1	PF	#	1	LBL 1
2	00	EXC	6	EXC	5	H	=	CD	STOP	=	K	
	10	Ø	PRNT	EXC	8	EXC	7	EXC	3	LBL	2	LBL 2
	20	PF	#	2	EXC	6	P	R	I	N	T	
	30	SPC	K	T	H	SPC	P	O	I	N	T	
	40	S	K	=	CD	STOP	=	K	8	PRNT	PF	

	0	1	2	3	4	5	6	7	8	9	COMMENTS
	50	EXC	5	X	N	=	CD	STOP	=	K	9	
	60	PRNT	-	K	1	)	÷	K	7	=	K	
	70	Ø	H	=	PRNT	EXC	8	EXC	7	EXC	3	
	80	LBL	4	PF	PF	E	N	D	CD	PF	PF	LBL 4
	90	RSET	LBL	5	X	Ø	=	CD	STOP	=	K	LBL 5
3	00	1	PRNT	Y	Ø	=	CD	STOP	=	K	2	
	10	PRNT	Y	'	Ø	=	CD	STOP	=	K	3	
	20	PRNT	Y	"	Ø	=	CD	STOP	=	K	4	
	30	PRNT	Y	3	Ø	=	CD	STOP	=	K	5	
	40	PRNT	RADR	GODP	LBL	6	I	N	P	U	T	LBL 6
	50	SPC	D	A	T	A	CD	PF	D	O	SPC	
	60	J	SPC	P	O	I	N	T	S	,	SPC	
	70	J	=	=	K	8	STOP	=	K	7	PRNT	
	80	PF	RADR	GODP	LBL	7	PF	PF	X	Y	SPC	LBL 7
	90	P	A	I	R	S	;	SPC	Y	'	K	
4	00	1	PF	PRNT	K	2	PRNT	PF	K	3	PRNT	
	10	RADR	GODP	LBL	8	K	4	-	2	×	K	LBL 8
	20	Ø	×	K	5	=	K	6	+	K	Ø	
	30	×	K	5	=	K	5	RADR	GODP	LBL	Ø	LBL Ø
	40	CLR	D/R*		Program your function							Your function
	50	g(x, y, y') as $g(K_1, K_2, K_3)$ finishing with the										$g(x, y, y') \equiv g(K_1, K_2, K_3)$
	60	sequence					=	K	4	RADR	GODP	
	70	*The D/R step should be omitted unless your function										
	80	includes trigonometric function arguments in degree										
	90	measure.										
		0	1	2	3	4	5	6	7	8	9	

NOTES

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

PROGRAM DESCRIPTION

This program is designed to solve higher order differential equations of the form

$$\frac{d^N y}{dx^N} = g\left(x\ ;\ y\ ;\ \frac{dy}{dx}\ ;\ \cdots\ ;\ \frac{d^{N-1} y}{dx^{N-1}}\right)$$

where $N \geq 3$; N is limited only by the number of available storage registers (from R_{11} through R_{xx}) and the practical limits of available program steps and execution time.

Given the initial values x_0, y_0, $\frac{dy_0}{dx}$, ... , $\frac{d^{N-1} y_0}{dx^{N-1}}$, and the x-interval value h, the Euler method yields the solutions

$$y_1\ ,\ \cdots\ ,\ y_n\ ;\ \frac{dy_1}{dx}\ ,\ \cdots\ ;\ \frac{d^{N-1} y_1}{dx}\ ,\ \cdots\ ;$$

using the recursion formulas

$$x_{n+1} = x_n + h \qquad y_{n+1} = y_n + h\,\frac{dy_n}{dx}$$

$$\frac{dy_{n+1}}{dx} = \frac{dy_n}{dx} + h\,\frac{d^2 y_n}{dx^2} \qquad \frac{d^{N-1} y_{n+1}}{dx^{N-1}} = \frac{d^{N-1} y_n}{dx^{N-1}} + h\,\frac{d^N y_n}{dx^N}$$

Notation:

$$\frac{dy}{dx} \equiv y'\ ,\ \frac{d^2 y}{dx^2} \equiv y''\ ,\ \cdots\ ,\ \frac{d^N y}{dx^N} \equiv y^N,$$

REGISTERS USED

$K_\emptyset$: point counter
K_1 : x_0, x_1, ...
K_2 : y_0 , y_1 , ...
K_3 : h, x-increment
K_4 : J, # of points
K_5 : K, Kth point printed
K_6 : working
K_7 : working

$R_{\emptyset\emptyset\emptyset}$: n counter
$R_{\emptyset 1}$: n + 1 counter
$R_{\emptyset 2}$: order N + 10
R_{11} : y'_0, y'_1, ...
R_{12} : y''_0, y''_1, ...
.
.
.
R_{ab} : $y^{(ab - 10)'}$

Subroutines used:
LBL Ø, 1, 2, 3, 4, 5, 6, 7, 9

The program includes subroutines for two modes of execution.

1. LBL 1, Manual mode: In this mode the user selects J, number of successive points to be calculated; and K to print every Kth point calculated; and h, the desired x-increment. If 0 is entered for J and K, then the program will calculate one point at a time, and h may be modified after any point by changing the value stored in K_3. To calculate successive points, it is only necessary to press CONT. To terminate the program press EXC 9 after calculating the last point wanted. For any $J \neq 0$ and $K \neq 0$, the program will calculate and print the number of points selected.

2. LBL 2, Automatic mode: In this mode the user can select the number of points J to be calculated over an interval from an initial x_0 to a final x_J. He again has the option of selecting a value K which will cause the program to print only every Kth point calculated. Thus for example, for a given equation one can calculate y values for 100 points between x_0 and x_J, that is, use a small H for accuracy, but only print out every 10th point calculated.

The program contains its own instruction for data entry for either mode. The examples show printouts for the alternative modes of execution.

In either mode, if SET FLAG is pressed prior to entry of any data point, the new values for the derivatives y' through $y^{(N-1)'}$ will be printed out.

NOTES

EXAMPLES

Solve the differential equation

$$\frac{d^2y}{dx^2} = \frac{1 - y^2}{10}\frac{dy}{dx} - y$$

Program g(x, y, y') as $g(K_1, K_2, R_{11})$:

CD 4 2 5 GODP LRN LBL Ø CLR
1 - K_2 x^2) ÷ 10 × R_{11} - K_2 = $RR_{Ø2}$
RADR GODP LRN

Example 1: $g(K_1, K_2, R_{11})$

Manual Execution,

Let J = 0 , K = 0
N = 2
$x_0 = 0$
$y_0 = 1$
h = .1
$y'_0 = 0$

Calculate 5 points
SET FLAG to print
y' after 4th point.

Example 2:

Automatic Execution,

Let J = 50 , K = 10
N = 2
$x_0 = 0$
$y_0 = 1$
$x_j = .5$
$y'_0 = 0$

```
0425  LBL_  F0
0426  0     F0
0427  CLR   F0
0428  1     F0
0429  -     F0
0430  K_    F0
0431  2     F0
0432  X^2   F0
0433  )     F0
0434  /     F0
0435  1     F0
0436  0     F0
0437  *     F0
0438  R__   F0
0439  1     F0
0440  1     F0
0441  -     F0
0442  K_    F0
0443  2     F0
0444  =     F0
0445  R__   F0
0446  R__   F0
0447  0     F0
0448  2     F0
0449  R/S   F0
0450  GODP  F0
```

```
NTH-ORD DIFF EQN

FOR MAN EXC 1
FOR AUTO EXC 2
#1

DO J POINTS, J=
              0.

PRINT KTH POINTS
K=
              0.

N=
              2.
X0=
              0.
Y0=
              1.
H=
              .1

ENTER Y'0 THRU
  Y(N-1)'0 :
              0.

XY PAIRS
*
              0.
              1.
*
*
              .1
             .99
*
*
              .2
     .97009801
*
*
              .3
  .9404833157
*
*
              .4
  .9014295858
*
-.4814122203
*
              .5
  .8532883638
*

END
```

```
NTH-ORD DIFF EQN

FOR MAN EXC 1
FOR AUTO EXC 2
#2

DO J POINTS, J=
             50.

PRINT KTH POINTS
K=
             10.

N=
              2.
X0=
              0.
Y0=
              1.
XJ=
              .5
H=
             .01

ENTER Y'0 THRU
  Y(N-1)'0 :
              0.

XY PAIRS
*
              0.
              1.
*
*
              .1
    .994504891
*
*
              .2
   .9790713411
*
*
              .3
   .9538459923
*
*
              .4
   .9190621226
*
*
              .5
   .8750331754
*

END
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1	To program your function first press			
	CD 4 2 5 GODP LRN LBL Ø CLR D/R*			
	Now enter your function $g[x, y, y', \ldots, y^{(N-1)'}]$			
	as $g[K_1, K_2, R_{11}, \ldots, R_{(10+N-1)}]$ finishing			
	with the sequence = RR_{02} RADR GODP LRN.			
	*D/R should be omitted unless your function			
	includes trigonometric functions in degrees.			
2		START		Title
3a	For manual execution	EXC 1	0	Instructions
b	For auto execution	EXC 2	0	#1 or #2
4	Number of points, J	CONT	0	Input data
	Print Kth points, K	CONT	0	as entered
	Order; N	CONT	0	XY Pairs
	x_0	CONT	0	*Derivatives
	y_0	CONT	0	END
5a	For manual, h	CONT	0	
b	For auto, x_j	CONT	0	See examples
6	y'_0	CONT	0	for detailed
	⋮	CONT	0	printouts.
		CONT	0	
	$y^{(N-1)'}_0$	CONT	0	
	For manual execution:			
	If J = 0 and K = 0, one point will be calculated;		y_i	
	to calculate the next	CONT	y_{i+1}	
	h may be changed prior to calculating			
	a new point by:	CD h_{new} = K_3		
	After calculating the			
	last point wanted	EXC 9		
	*In either mode, pressing SET FLAG prior to any			
	of the data entries will cause the calculated			
	derivatives to be printed out.			

		0	1	2	3	4	5	6	7	8	9	COMMENTS
0	00	PF	PF	N	T	H	-	0	R	D	SPC	
	10	D	I	F	F	SPC	E	Q	N	PF	F	
	20	0	R	SPC	M	A	N	SPC	E	X	C	
	30	SPC	1	PF	F	0	R	SPC	A	U	T	
	40	0	SPC	E	X	C	SPC	2	RSET	LBL	1	LBL 1
	50	#	1	π	=	K	7	IF<Ø	LBL	2	#	LBL 2
	60	2	π	+/-	=	K	7	CONT	PF	D	0	
	70	SPC	J	SPC	P	0	I	N	T	S	,	
	80	SPC	J	=	CLR	STOP	=	K	4	PRNT	PF	
	90	P	R	I	N	T	SPC	K	T	H	SPC	
1	00	P	0	I	N	T	S	K	=	CD	=	
	10	K	Ø	STOP	=	K	5	PRNT	PF	N	=	
	20	CD	STOP	=	PRNT	+	1	Ø	=	Rxxx	Ø	
	30	Ø	2	X	Ø	=	CD	STOP	=	K	1	
	40	PRNT	Y	Ø	=	CD	STOP	=	K	2	PRNT	
	50	K	7	IF<Ø	X	J	=	CD	STOP	=	PRNT	
	60	-	K	1	)	÷	K	4	=	K	3	
	70	CONT	H	=	K	7	IF≥Ø	CD	STOP	=	CONT	
	80	K	3	PRNT	CD	1	1	=	Rxx	Ø	Ø	
	90	PF	E	N	T	E	R	SPC	Y	'	Ø	
2	00	SPC	T	H	R	U	PF	SPC	SPC	Y	(	
	10	N	-	1	)	'	Ø	SPC	:	LBL	5	LBL 5
	20	Rxx	Ø	Ø	-	Rxx	Ø	2	)	IF≥Ø	EXC	
	30	6	CONT	CD	STOP	=	Rxx	Rxx	Ø	Ø	PRNT	
	40	Rxx	Ø	Ø	+	1	=	Rxx	Ø	Ø	EXC	
		0	1	2	3	4	5	6	7	8	9	

PROGRAM STEPS

	0	1	2	3	4	5	6	7	8	9	COMMENTS
50	5	LBL	6	PF	PF	X	Y	SPC	P	A	LBL 6
60	I	R	S	PF	EXC	7	LBL	3	CD	1	LBL 3
70	Σ_0	CD	1	1	=	Rxx	Ø	Ø	+	1	
80	=	Rxx	Ø	1	EXC	Ø	LBL	4	Rxx	Rxx	LBL 4
90	Ø	Ø	+	K	3	×	Rxx	Rxx	Ø	1	
3 00	=	Rxx	Rxx	Ø	Ø	K	Ø	÷	K	5	
10	=	K	6	-	K	6	int	=	K	6	
20	IF=Ø	CLR	IFFG	Rxx	Rxx	Ø	Ø	PRNT	CONT	Rxx	
30	Ø	1	=	Rxx	Ø	Ø	+	1	=	Rxx	
40	Ø	1	+/-	+	Rxx	Ø	2	)	IF≥Ø	EXC	
50	4	CONT	K	1	+	K	3	=	K	1	
60	K	2	+	K	3	×	Rxx	1	1	=	
70	K	2	K	6	IF=Ø	EXC	7	CONT	K	Ø	
80	-	K	4	)	IF=Ø	EXC	9	CONT	K	4	
90	IF=Ø	K	7	IF≥Ø	K	2	STOP	CONT	EXC	3	
4 00	LBL	7	*	K	1	PF	PRNT	K	2	PRNT	LBL 7
10	PF	*	RADR	GODP	LBL	9	PF	PF	E	N	LBL 9
20	D	CD	PF	PF	RSET	LBL	Ø	CLR	D/R*		LBL Ø
30				Program your function							Your function
40				$g[x, y, y', y'', \ldots, y^{(N-1)'}]$ as							$g[x, y, y', \ldots, y^{(N-1)'}]$
50				$g[K_1, K_2, R_{11}, R_{12}, \ldots, R_{(10+N-1)}]$							
60				finishing with the sequence							$\equiv g[K_1, K_2, R_{11}, R_{12}, \ldots]$
70				=	Rxx	Rxx	Ø	2	RADR	GODP	

*The D/R step should be omitted unless your function includes trigonometric function arguments in degree measure.

NOTES

MINIMUM HARDWARE REQUIRED

TEK 31 (512 Steps and 64 Registers)

REGISTERS USED

$K_{\emptyset}$: t_0, t_1, ...
K_1 : x_0, x_1, ...
K_2 : y_0, y_1, ...
K_3 : h
K_4 : dx/dt, ...
K_5 : dx^2/dt^2, ...
K_6 : a
K_7 : b
K_8 : c
K_9 : d

$R_{\emptyset\emptyset\emptyset}$: point counter
$R_{\emptyset 1}$: J, # of points
$R_{\emptyset 2}$: K, Kth point printed
$K_{\emptyset 3}$: working

Subroutines used:
LBL Ø, 1, 2, 3, 4, 5, 6, 9

PROGRAM DESCRIPTION

This program is designed to solve the simultaneous equations

$$\frac{dx}{dt} + ax + by = M(t)$$

$$\frac{dy}{dt} + cx + dy = N(t)$$

given the constants a, b, c, d, the functions M(t) and N(t), and the initial values t_0, x_0, and y_0. These equations can be reduced to

$$\frac{d^2x}{dt^2} = -(a + d)\frac{dx}{dt} + (bc - ad)x + M'(t)$$

$$+ d \times M(t) - b \times N(t)$$

and $y = \frac{1}{b}\left[M(t) - \frac{dx}{dt} - ax\right]$; where $M'(t) = \frac{d}{dt}M(t)$.

The first of these latter equations can be solved by Euler's method using the recursion formulas

$$t_{n+1} = t_n + h$$

$$x_{n+1} = x_n + h\frac{dx_n}{dt}$$

$$\frac{dx_{n+1}}{dt} = \frac{dx_n}{dt} + h\frac{d^2x_n}{dt^2}$$

after which the solution to the second equation is direct.

Note that the solution requires differentiation of M(t) to get M'(t); choose your notation so that the simpler of the two functions of t in your pair of equations is represented by M(t).

The program includes subroutines for two modes of execution.

1. LBL 1, Manual mode: In this mode the user selects J, number of successive points to be calculated; K to print every Kth point calculated; and h, the desired t-increment. If 0 is entered for J and K, then the program will calculate one point at a time, and h may be modified after any point by changing the value stored in K_3. To calculate successive points, it is only necessary to press CONT. To terminate the program press EXC 9 after calculating the last point wanted. For any $J \neq 0$ and $K \neq 0$, the program will calculate and print the number of points selected.

2. LBL 2, Automatic mode: In this mode the user can select the number of points T to be calculated over an interval from an initial t_0 to a final t_J. He again has the option of selecting a value K which will cause the program to print only every Kth point calculated. Thus for example, for a given equation one can calculate y values for 100 points between t_0 and t_J, that is, use a small h for accuracy, but only print out every 10th point calculated.

The program contains its own instruction for data entry for either mode. The examples show printouts for the alternative modes of execution.

NOTES

Solve the simultaneous differential equations, given the initial conditions:

$$\frac{dx}{dt} + x - y = \cos t \qquad t_0 = 0$$

$$x_0 = 0$$

$$\frac{dy}{dt} - x + y = \cos t \qquad y_0 = 0$$

Now M(t) = cos t, so M'(t) = -sin t

Thus $M(t_0) = \cos t_0 = 1$

$M(K_\emptyset) = K_\emptyset \cos$

$N(K_\emptyset) = K_\emptyset \cos$

$M'(K_\emptyset) = CD - K_\emptyset \sin$

From these data we can program the function as listed.

Example 1:

Manual execution,

Let J = 0, K = 0

Let h = .1

Calculate 5 points,

EXC 9 to end.

Example 2:

Automatic execution,

Let J = 100, K = 20

Let $t_J = .5$

```
0405  LBL_ F0
0406  0    F0
0407  CLR  F0
0408  K_   F0
0409  0    F0
0410  COS  F0
0411  *    F0
0412  K_   F0
0413  9    F0
0414  -    F0
0415  K_   F0
0416  7    F0
0417  *    F0
0418  K_   F0
0419  0    F0
0420  COS  F0
0421  -    F0
0422  K_   F0
0423  0    F0
0424  SIN  F0
0425  =    F0
0426  K_   F0
0427  5    F0
0428  K_   F0
0429  3    F0
0430  SUM0 F0
0431  K_   F0
0432  0    F0
0433  COS  F0
0434  =    F0
0435  K_   F0
0436  2    F0
0437  R AD F0
0438  GODP F0
```

```
2-SIM DIFF EQNS

FOR MAN EXC 1
FOR AUTO EXC 2
#1

DO J PNTS J=
          0.

PRNT KTH PNTS K=
          0.

ENTER DATA

A=
          1.
B=
-         1.
C=
-         1.
D=
          1.
X0=
          0.
Y0=
          0.
T0=
          0.
H=
          .1
M(T0)=
          1.

T; XY PAIRS

          0.
          0.
          0.

          .1
          .1
 .1049958347

          .2
          .2
 .2089509135

          .3
 .2989017491
 .3109256356

          .4
 .3956377867
 .4099803703

          .5
 .4891781445
 .5051888093

END
```

```
2-SIM DIFF EQNS

FOR MAN EXC 1
FOR AUTO EXC 2
#2

DO J PNTS J=
        100.

PRNT KTH PNTS K=
         20.

ENTER DATA

A=
          1.
B=
-         1.
C=
-         1.
D=
          1.
X0=
          0.
Y0=
          0.
T0=
          0.
TJ=
          .5
H=
       .005
M(T0)=
          1.

T; XY PAIRS

          0.
          0.
          0.

          .1
 9.985688131E-02
 .1000841091

          .2
 .1987617131
 .1991725027

          .3
 .2957225019
 .2962788912

          .4
 .3897673652
 .3904361022

          .5
 .4799541163
 .4807058694

END
```

PROGRAM EXECUTION

STEP	ENTER	PRESS	DISPLAY	PRINTOUT
1	To program your function first press CD 4 Ø 5 GODP LRN Ø CLR D/R* . Now enter your function as K_9 × $M(K_Ø)$ − K_7 × $N(K_Ø)$ + $M'(K_Ø)$ = K_5 . Follow with the sequence K_3 Σ_0 $M(K_Ø)$ = K_2 RADR GODP LRN . *D/R should be omitted unless your function includes trigonometric functions in degrees.			
2		START	0	Title
3a	For manual execution	EXC 1	0	Instructions
b	For auto execution	EXC 2	0	#1 or #2
4	Number of points, J	CONT	0	Input data
	Print Kth points, K	CONT	0	as entered
	a	CONT	0	t_0
	b	CONT	0	x_0
	c	CONT	0	y_0
	d	CONT	0	t_1
	x_0	CONT	0	x_1
	y_0	CONT	0	y_1
	t_0	CONT	0	.
5a	For manual, h	CONT	0	.
b	For auto, t_J	CONT	0	. END
	For manual execution:			See examples for detailed printout.
	If J = 0 and K = 0, one point will be calculated;		y_i	
	to calculate the next	CONT	y_{i+1}	
	h may be changed prior to calculating a new point by:	CD h_{new} = K_3		
	After calculating the last point wanted	EXC 9		

PROGRAM STEPS

		0	1	2	3	4	5	6	7	8	9	COMMENTS
0	00	PF	PF	2	-	S	I	M	SPC	D	I	
	10	F	F	SPC	E	Q	N	S	PF	PF	F	
	20	O	R	SPC	M	A	N	SPC	E	X	C	
	30	SPC	1	PF	F	O	R	SPC	A	U	T	
	40	O	SPC	E	X	C	SPC	2	RSET	LBL	1	LBL 1
	50	#	1	EXC	5	H	=	CD	STOP	=	K	
	60	3	PRNT	EXC	6	LBL	2	#	2	EXC	5	LBL 2
	70	T	J	=	CD	STOP	=	PRNT	-	K	Ø	
	80	)	÷	Rxx	Ø	1	=	K	3	H	=	
	90	PRNT	LBL	6	M	(	T	Ø	)	=	CD	LBL 6
1	00	STOP	=	PRNT	-	K	6	×	K	1	-	
	10	K	7	×	K	2	=	K	4	PF	PF	
	20	T	;	SPC	X	Y	SPC	P	A	I	R	
	30	S	PF	EXC	4	LBL	3	EXC	Ø	K	7	LBL 3
	40	×	K	8	-	K	6	×	K	9	)	
	50	×	K	1	+	K	5	-	K	4	×	
	60	(	K	6	+	K	9	=	K	5	K	
	70	3	×	K	4	)	Σ_1	K	4	+	K	
	80	3	×	K	5	=	K	4	K	2	-	
	90	K	4	-	K	6	×	K	1	=	÷	
2	00	K	7	=	K	2	Rxx	Ø	Ø	+	1	
	10	=	Rxx	Ø	Ø	÷	Rxx	Ø	2	=	Rxx	
	20	Ø	3	-	Rxx	Ø	3	int	)	IF=Ø	CLR	
	30	EXC	4	CONT	Rxx	Ø	Ø	-	Rxx	Ø	1	
	40	)	IF=Ø	EXC	9	CONT	Rxx	Ø	1	IF=Ø	K	
		0	1	2	3	4	5	6	7	8	9	

	0	1	2	3	4	5	6	7	8	9	COMMENTS
50	2	STOP	CONT	EXC	3	LBL	4	PF	K	Ø	LBL 4
60	PRNT	K	1	PRNT	K	2	PRNT	PF	RADR	GODP	
70	LBL	5	PF	D	O	SPC	J	SPC	P	N	LBL 5
80	T	S	SPC	J	=	CLR	=	Rxxx	Ø	Ø	
90	Ø	STOP	=	Rxx	Ø	1	PRNT	PF	P	R	
3 00	N	T	SPC	K	T	H	SPC	P	N	T	
10	S	SPC	K	=	CD	STOP	=	Rxx	Ø	2	
20	PRNT	PF	E	N	T	E	R	SPC	D	A	
30	T	A	PF	PF	A	=	CD	STOP	=	K	
40	6	PRNT	B	=	CD	STOP	=	K	7	PRNT	
50	C	=	CD	STOP	=	K	8	PRNT	D	=	
60	CD	STOP	=	K	9	PRNT	X	Ø	=	CD	
70	STOP	=	K	1	PRNT	Y	Ø	=	CD	STOP	
80	=	K	2	PRNT	T	Ø	=	CD	STOP	=	
90	K	Ø	PRNT	RADR	GODP	LBL	9	PF	E	N	LBL 9
4 00	D	CD	PF	PF	RSET	LBL	Ø	CLR	D/R*		LBL Ø
10											Your functions
20				Program your functions as							$M(K_Ø)$
30				$K_9 \times M(K_Ø) - K_7 \times N(K_Ø) + M'(K_Ø) = K_5$							$N(K_Ø)$
40	K	3	Σ_0		$M(K_Ø)$		=	K_2		RADR GODP	$M'(K_Ø)$
50											
60	*The D/R step should be omitted unless your functions										
70	$M(K_Ø)$, $N(K_Ø)$, or $M'(K_Ø)$ include trigonometric function										
80	arguments in degree measure.										
90											
	0	1	2	3	4	5	6	7	8	9	

NOTES

CHANGE INFORMATION

At Tektronix, we continually strive to provide the best possible products by adding improvements as soon as they are developed and tested.

Sometimes, due to printing and shipping requirements, we can't get these changes immediately into printed manuals. Hence, your manual may contain new change information on following pages.

A single change may affect several sections. Sections of the manual are often printed at different times, so some of the information on the change pages may already be in your manual. Since the change information sheets are carried in the manual until ALL changes are permanently entered, some duplication may occur. If no such change pages appear in this section, your manual is correct as printed.